9.2

IBM MQ -Teknik genel bakış

Not

Bu bilgileri ve desteklediği ürünü kullanmadan önce, “Özel notlar” sayfa 297 bölümündeki bilgileri okuyun.

This edition applies to version 9 release 2 of IBM® MQ and to all subsequent releases and modifications until otherwise indicated in new editions.

İçindekiler

IBM MQ Teknik genel bakış

Uygulamalarınızı bağlamak ve bilgi dağıtımını kuruluşunuzda yönetmek için IBM MQ kullanın.

IBM MQ , tutarlı bir uygulama programlama arabirimi kullanarak, programların, bileşenlerin (işlemciler, işletim sistemleri, altsistemler ve iletişim protokolleri) farklı bir ağ üzerinden bir diğeriyle iletişim kurmasını sağlar. Bu arabirimi kullanarak tasarlanmış ve yazılan uygulamalar, ileti kuyruğa alma uygulamaları olarak bilinir.

Use the following subtopics to find out about message queuing and other features provided by IBM MQ.

İlgili kavramlar

IBM MQ' a giriş

İlgili görevler

IBM MQ mimarisinin planlanması

İlgili başvurular

“İleti kuyruklama özelliği ana özellikleri ve avantajları” sayfa 7

Bu bilgiler, ileti kuyruklama bilgilerinin bazı özelliklerini ve avantajlarını vurgular. Bu belge, ileti kuyruklama iletisinin güvenlik ve veri bütünlüğü gibi özellikleri açıklar.

Ürün gereksinimlerinin ve destek bilgilerinin nerede bulunması gerekir

İletinin kuyruğa alınması

IBM MQ ürünleri, programları tutarlı bir uygulama programlama arabirimi kullanarak, bileşenlerin (işlemciler, işletim sistemleri, altsistemler ve iletişim protokolleri) farklı bir ağ üzerinden bir diğeriyle iletişim kurmasını sağlar.

Bu arabirim kullanılarak tasarlanan ve yazılan uygulamalar, *ileti alışverişi* ve *kuyruğa alma* biçemin kullanıldığı için *ileti kuyruklama* uygulamaları olarak bilinir:

- İleti alışverişi, programların birbirini doğrudan aramak yerine, iletilerde diğer verileri göndermesiyle iletişim kurduğunu gösterir.
- Kuyruğa Alma, iletilerin depodaki kuyruklara yerleştirildiğinden, programların birbirinden bağımsız olarak, farklı hızlarda ve zamanlarda, farklı konumlarda ve aralarında mantıksal bir bağlantı olmadan çalıştırılmasına olanak sağlar.

İleti kuyruklama uzun yıllardır veri işlemede kullanıldı. Günümüzde en yaygın olarak elektronik postalarda kullanılır. Kuyruğa alınmadan, uzun mesafeler üzerinden elektronik bir ileti göndermek için rotanın üzerindeki her düğümün iletileri iletmek için kullanılabilmesini ve adresin günlüğe kaydedilmesini gerektirir ve bunları bir ileti göndermeye çalıştığınız gerçeinden bilinçli olarak kaydedilir. Kuyruğa alma sisteminde, iletiler, sistem bunları iletmeye hazır oluncaya kadar ara düğümlerde saklanır. En son varış noktalarında, adresin okunmaya hazır oluncaya kadar elektronik posta kutusunda depolanır.

Öyle bile olsa, günümüzde pek çok karmaşık iş hareketi kuyruksuz olarak işlenmektedir. Büyük bir ağda, sistem, kullanıma hazır bir durumda binlerce bağlantının korunabileceği bir sistem olabilir. Sistemin bir parçasında sorun yaşarsa, sistemin birçok bölümü kullanılamaz duruma gelir.

Programları, programlar için elektronik posta olarak sıralamayı düşünebilirsiniz. İleti kuyruklama ortamında, bir uygulama takımının bir parçasını oluşturan her program, belirli bir isteğe yanıt olarak, iyi tanımlanmış, kendi kendine yeten bir işlev gerçekleştirir. Başka bir programla iletişim kurmak için, programın önceden tanımlanmış bir kuyruğa ileti koyması gerekir. Diğer program iletiyi kuyruktan alır ve iletinin içerdiği istekleri ve bilgileri işler. Yani mesaj quesing, programdan programa iletişim tarzı bir şey.

Kuyruğa Yollama, bir uygulama bunları işlemeye hazır oluncaya kadar iletilerin tutulmakta olduğu mekanizmadır. Kuyruğa Alma işlemi şunları sağlar:

- İletişim kodunu yazmak zorunda kalmadan programlar arasında (her biri farklı ortamlarda çalıştırılabilir) iletişim kurun.

- Bir programın iletileri işleyeceği sırayı seçin.
- İleti sayısı bir eşiği aştığında kuyruğa hizmet vermek üzere birden fazla program için düzenleme yaparak bir sistemde yük dengelemesi.
- Birincil sisteminiz kullanılamaz durumdaysa, diğer bir sistem için, kuyruklara hizmet vermek üzere başka bir sistem düzenleyerek uygulamalarınızın kullanılabilirliğini artırın.

İleti kuyruğu nedir?

Yalnızca kuyruk olarak bilinen bir ileti kuyruğu, iletilerin gönderilebileceği adlandırılmış bir hedeftir. İletiler, kuyruklara hizmet veren programlar tarafından alınıncaya kadar kuyruklarda birikir.

Kuyruklar, bir kuyruk yöneticisi tarafından yönetilir ve yönetilir (bkz. “İleti kuyruklama terminolojisi” sayfa 9). Bir kuyruğun fiziksel yapısı, kuyruk yöneticisinin çalıştırıldığı işletim sistemine bağlıdır. Kuyruk, bir bilgisayarın belleğindeki uçucu bir arabellek alanı ya da kalıcı bir depolama aygıtında (disk gibi) bir veri kümesi olabilir. Kuyrukların fiziksel yönetimi, kuyruk yöneticisinin sorumluluğundadır ve katılımcı uygulama programlarına açık bir şekilde ifade edilmiyor.

Programlar, kuyruklara yalnızca kuyruk yöneticisinin dış hizmetleriyle erişir. Bir kuyruk açabilir, iletiler üzerine iletiler yerleştirebilir, bundan ileti alabilir ve kuyruğu kapatabilirler. Ayrıca, kuyrukların özniteliklerini de ayarlayabilir ve sorgulayabilirler.

İleti kuyruğa alma farklı stilleri

Noktadan noktaya iletişim

Kuyruğa bir ileti yerleştirilir ve bir uygulama bu iletiyi alır.

Noktadan noktaya ileti sisteminde, bir gönderme uygulamasının, bu uygulamaya bir ileti gönderebilmesi için önce alma uygulamasına ilişkin bilgileri bilmesi gerekir. Örneğin, gönderme uygulamasının, bilgilerin gönderileceği kuyruğun adını bilmesi ve kuyruk yöneticisi adını da belirtmesi gerekebilir.

Yayınla/Abone Ol

Bir yayınlama uygulaması tarafından yayınlanan her iletinin bir kopyası, ilgili her uygulamaya teslim edilir. Çok sayıda, bir ya da hiç ilgi çeken uygulama olabilir. Yayınlama/abone olma ilgili bir uygulamaya abone olarak bilinir ve iletiler, abonelik tarafından tanımlanan bir kuyrukta kuyruğa alınır.

Yayınlama/abone olma ileti sistemi, bilgi sağlayıcısını bu bilgilerin tüketicilerinden çözmenize olanak sağlar. Gönderme uygulaması ve alma uygulamasının, gönderilecek ve alınacak bilgiler için birbirleri hakkında çok fazla bilgi sahibi olması gerekmez. Daha fazla bilgi için “Yayınlama/abone olma ileti alışverişi” sayfa 59 başlıklı konuya bakın.

Uygulama tasarımcısı ve geliştiricisine ileti kuyruklama avantajları

IBM MQ , uygulama programlarının, ileti odaklı işleme katılmak için *ileti kuyruklama* kullanmasını sağlar. Uygulama programları, uygun ileti kuyruğa alma yazılım ürünlerini kullanarak farklı platformlar arasında iletişim kurabilir. Örneğin, z/OS uygulamaları IBM MQ for z/OSaracılığıyla iletişim kurabilirler. Uygulamalar, temeldeki iletişimlerin mekaniklerinden korunmuş. İleti kuyruklama avantajlarının diğer bazı avantajları şunlardır:

- Birçok uygulama arasında paylaşabileceğiniz küçük programları kullanarak uygulamalar tasarlayabilirsiniz.
- Bu yapı taşlarını yeniden kullanarak hızlı bir şekilde yeni uygulamalar oluşturabilirsiniz.
- İleti kuyruklama tekniklerini kullanmak için yazılan uygulamalar, kuyruk yöneticilerinin çalışma biçimlerindeki değişikliklerden etkilenmez.
- İletişim protokollerini kullanmanıza gerek yoktur. Kuyruk yöneticisi, sizin için iletişimin tüm yönleriyle ilgilenir.
- İletileri alan programlar, iletilerin gönderileceği sırada çalışmamaya gerek yoktur. İletiler kuyruklarda saklanır.

Tasarımcılar uygulamalarının maliyetini düşürebilir, çünkü geliştirme daha hızlı, daha az geliştirici gerekir ve programlama becerisine ilişkin talepler, ileti kuyruklama özelliği kullanmayan uygulamalar için daha düşüktür.

IBM MQ , uygulamaların çalıştırıldığı her yerde *ileti kuyruğu arabirimi* (ya da MQI) olarak bilinen ortak bir uygulama programlama arabirimini gerçekleştirir. Bu, uygulama programlarını bir platformdan başka bir platformdan diğerine atmanızı kolaylaştırır.

MQI ile ilgili ayrıntılar için bkz. İleti Kuyruğu Arabirimi-Genel Bakış.

İleti kuyruklama özelliği ana özellikleri ve avantajları

Bu bilgiler, ileti kuyruklama bilgilerinin bazı özelliklerini ve avantajlarını vurgular. Bu belge, ileti kuyruklama iletisinin güvenlik ve veri bütünlüğü gibi özellikleri açıklar.

İleti kuyruklama tekniklerini kullanan uygulamaların başlıca özellikleri şunlardır:

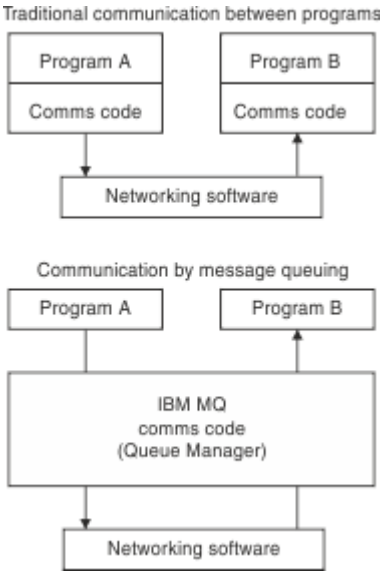
- Programlar arasında doğrudan bağlantı yoktur.
- Programlar arasındaki iletişim zaman bağımsız olabilir.
- Çalışma, kendi kendine yer alan küçük programlar tarafından gerçekleştirilebiliyor.
- İletişim, olaylar tarafından yönlendirilebilir.
- Uygulamalar bir iletiye öncelik atayabilir.
- Güvenlik.
- Veri bütünlüğü.
- Kurtarma desteği.

Programlar arasında doğrudan bağlantı yok

İleti kuyruklama, programdan programa doğrudan iletişim için kullanılan bir tekniktir. Bu, programların birbirleriyle iletişim kurduğu herhangi bir uygulama içinde kullanılabilir. ˜letiŸim, kuyruktan (bir kuyruk yöneticisine ait) iletileri ve kuyruktan iletileri almaya baŸka bir program g " nderen bir program taraflardndan oluŸaca § lar.

Programlar, başka programlar tarafından kuyruğa konulan iletileri alabilirler. Diğer programlar, alma programı ile aynı kuyruk yöneticisine ya da başka bir kuyruk yöneticisine bağlanabilir. Bu başka bir kuyruk yöneticisi başka bir sistemde, farklı bir bilgisayar sisteminde ya da farklı bir iş ya da teşebbüs içinde olabilir.

İleti kuyruklarını kullanarak iletişim kuran programlar arasında fiziksel bağlantı yoktur. Bir program, iletileri kuyruk yöneticisinin sahibi olduğu bir kuyruğa gönderir; başka bir program da kuyruktan ileti alır (bkz. Şekil 1 sayfa 8).



Şekil 1. İleti kuyruklama, geleneksel iletişimle karşılaştırılıyor

elektronik posta ile olduğu gibi, bir işlemin bir parçası olan bireysel mesajlar bir mağazada bir ağ üzerinden-ve-ileriTemel. Düğümler arasında bir bağlantı başarısız olursa, bağlantı geri yükleninceye kadar ileti alıkonur ya da işletmen ya da program iletiyi yeniden yönlendirir.

Bir iletinin kuyruktan kuyruğa taşındığı mekanizma programlardan gizlenir. Bu yüzden programlar daha basitti.

Zamana karşı bağımsız iletişim
Diğer kullanıcıların iş yapabilmesini isteyen programlar, bir isteğin yanıtının beklemesine gerek yoktur. Diğer işleri de yapabilir ve yanıt geldiğinde ya da daha sonra geldiğinde yanıtı işleme alabilirler. Bir ileti alışverişi uygulaması yazarken, program bir ileti gönderdiğinde ya da hedef iletiyi alabildiği zaman bilmeniz gerekir (ya da ilgilenmeyin). İleti kaybedilmez; hedef işlenmeye hazır oluncaya kadar kuyruk yöneticisi tarafından korunur. İleti, bir program tarafından kaldırılıncaya kadar kuyruğun üzerinde kalır. Bu, gönderme ve alma uygulama programlarının kesilen olduğu anlamına gelir; gönderen, alıcının iletiyi almayı kabul etmesini beklemeden işlemeye devam edebilir. Hedef uygulamanın, ileti gönderildiğinde çalıştırılması bile gerekmez. Bu ileti başlatıldıktan sonra iletiyi alabilir.

Küçük programlar
İleti kuyruklama, küçük, kendi kendine bulundurulan programları kullanmanın avantajlarını kullanmanıza olanak sağlar. Bir işin tüm bölümlerini sırayla gerçekleştiren tek, büyük bir program yerine, işi daha küçük, bağımsız programlar üzerinden dağıtabilirsiniz. İstekte bulunan program, her bir programın her birine, işlevlerini gerçekleştirmelerini isteyen iletileri gönderir; her program tamamlandığında, sonuçlar bir ya da daha çok ileti olarak geri gönderilir.

İletilerle yönlendirilen işleme
İletiler bir kuyruğa ulaştığında, *tetikleme*komutunu kullanarak otomatik olarak bir uygulamayı başlatabilirler. Gerekliyse, ileti (ya da iletiler) işlendiğinde uygulamalar durdurulabilir.

Olaya dayalı işleme
Programlar, kuyrukların durumuna göre denetlenebilir. Örneğin, bir ileti kuyruğa varır varmaz bir programın başlatılmasını ayarlayabilir ya da programın başlatılmamasını, örneğin, kuyruğun belirli bir önceliğinin üstünde 10 ileti ya da kuyruğun herhangi bir önceliğinin 10 iletisi olduğunu belirleyebilirsiniz.

İleti önceliği
Bir program, iletiyi bir kuyruğa koyduğunda, bir ileti için öncelik atayabilir. Bu, yeni iletinin eklendiği kuyrukta yer alan konumu belirler.

Programlar, kuyruktan iletilerin kuyrukta bulunduğu sırada ya da belirli bir iletiyi alarak iletileri alabilirler. (Bir program, daha önce gönderdiği bir isteğin yanıtını arıyorsa, belirli bir iletiyi almak isteyebilir.)

Güvenlik
Güvenlik olanakları, bir kuyruk yöneticisi kullandıklarında uygulamaların kimlik doğrulaması da dahil olmak üzere, kuyruk yöneticisine kuyruk gibi kaynakları kullandıklarında yetki denetimleri ve ağ üzerinden seyahat ederken ileti verilerinin şifrelenmesi gibi ve kuyruklarda bulunduğu için sağlanır. Güvenlik hakkında daha fazla bilgi için bkz. Güvenlik-Genel Bakış.

Veri bütünlüğü
Veri bütünlüğü, iş birimleri tarafından sağlanır. İş birimlerinin başlangıç ve bitiş zamanlarının uyumlulaştırılması, her MQGET ya da MQPUT üzerinde bir seçenek olarak desteklenir; bu seçenek, iş biriminin sonuçlarının kesinleştirilmesine ya da geriye işlenmesine olanak sağlar. Sync point support operates either internally or externally to IBM MQ depending on the form of sync point coordination selected for the application.

Kurtarma desteği
Kurtarma işlemi için, tüm kalıcı IBM MQ güncellemelerinin günlüğe kaydedilmesi gerekir. Kurtarma gerekliyse, tüm kalıcı iletiler geri yüklenir, tüm uçuş hareketleri geri alınır ve herhangi bir eşitleme noktası kesinleştirme ve geri alma işlemleri, tutarlılık noktası yöneticisinin denetiminde olağan şekilde işlenir. Kalıcı iletilere ilişkin daha fazla bilgi için bkz. İleti kalıcılığı.

Not: IBM MQ istemcileri ve sunucuları dikkate alındığında, yeni platformlarda ek IBM MQ MQI clients ' i desteklemek için bir sunucu uygulamasını değiştirmeniz gerekmez. Benzer şekilde, IBM MQ MQI client , değişiklik olmadan, ek sunucu tipleriyle işlev görebilirler.

İleti kuyruklama terminolojisi

Bu bilgiler, ileti kuyrukta kullanılan bazı terimlerle ilgili bir kavrayış sağlar.

Bunlar aşağıdakileri içerir:

- İleti
- İleti tanımlayıcısı
- Kuyruk
- kuyruk yöneticisi
- Kanallar
- İleti kanalı aracısı
- Küme
- z/OS Paylaşılan kuyruk
- z/OS Kuyruk paylaşımı grubu
- z/OS Grup içi kuyruğa alma
- IBM MQ MQI client
- Noktadan noktaya iletişim
- Yayınla/abone ol
- Konu
- abonelik

İleti

İleti kuyruklama sırasında, bir ileti, bir program tarafından gönderilen ve başka bir program için hazırlanmış veri toplamlıdır. Bkz. IBM MQ iletileri. İleti tiplerine ilişkin bilgi için İleti tipleribaşlıklı konuya bakın.

İleti tanımlayıcısı

Bir IBM MQ iletisi, denetim bilgileri ve uygulama verilerinden oluşur.

Denetim bilgileri, bir ileti tanımlayıcısı yapısında (MQMD) tanımlanır ve aşağıdaki gibi şeyler içerir:

- İletinin tipi

- İletiye ilişkin tanıtıcı
- İletinin teslim edilmesi için öncelik

Uygulama verilerinin yapısı ve içeriği katılımcı programlar tarafından saptanır, IBM MQtarafından değil.

Kuyruk

İletilerin gönderilebileceği adlandırılmış hedef. İletiler, kuyruklara hizmet veren programlar tarafından alınıncaya kadar kuyruklarda birikir.

Kuyruk yöneticisi

Kuyruk yöneticisi , uygulamalara kuyruğa alma hizmetleri sağlayan bir sistem programıdır.

Programlara ileti yerleştirebilmeleri ve kuyruktan ileti alabilmesi için bir uygulama programlama arabirimi sağlar. Kuyruk yöneticisi, denetimcilerin yeni kuyruklar yaratabilmesi, varolan kuyrukların özelliklerini değiştirebilmeleri ve kuyruk yöneticisinin işlemini denetleyebilmeleri için ek işlevler sağlar.

IBM MQ ileti kuyruklama hizmetlerinin bir sistemde kullanılabilir olması için, bir kuyruk yöneticisi çalışır durumda olmalıdır. Tek bir sistemde çalışan birden çok kuyruk yöneticisi olabilir (örneğin, bir sınama sistemini bir *canlı* sistemden ayırmak için). Bir uygulamaya, her kuyruk yöneticisi bir *bağlantı tanıtıcısı* (*Hconn*) ile tanımlanır.

birçok farklı uygulama aynı anda kuyruk yöneticisi hizmetlerini kullanabilir ve bu uygulamalar tamamen alakasız olabilir. Bir programın kuyruk yöneticisi hizmetlerini kullanması için, o kuyruk yöneticiyle bağlantı kurması gerekir.

Diğer kuyruk yöneticilerine bağlı uygulamalara ileti gönderme uygulamaları için, kuyruk yöneticilerinin kendi aralarında iletişim kurabilmeleri gerekir. IBM MQ , bu tür uygulamalar arasında iletilerin güvenli şekilde sunulmasını sağlamak için bir *depolama ve iletme* iletişim kuralını uygular.

Kanallar

Kanallar, bir kuyruk yöneticisinden diğerine iletişim yolu sağlayan nesnellerdir. Kanallar, iletileri bir kuyruk yöneticisinden diğerine taşımak için kullanılır ve temeldeki iletişim protokollerinden uygulamaları korurlar. Kuyruk yöneticileri aynı sistemde ya da aynı altyapıda ya da farklı platformlarda farklı sistemler üzerinde bulunabilir. Gönderilen iletiler birçok yerden kaynaklanabilir:

- Verileri bir düğümden diğerine aktaran kullanıcı tarafından yazılan uygulama programları.
- PCF komutlarını ya da MQAI olanağını kullanan kullanıcı tarafından yazılmış yönetim uygulamaları.
- IBM MQ Explorer.
- Başka bir kuyruk yöneticisine izleme kodu ekleme olayı iletileri gönderen kuyruk yöneticileri.
- Uzak denetim komutlarını başka bir kuyruk yöneticisine gönderen kuyruk yöneticileri. Örneğin, MQSC komutlarını ya da administrative REST APIkomutunu kullanın.

İleti kanalı aracısı

Bir ileti kanalı aracısı, bir kanalın bir ucundan bir ucuna sahip olur. Bir çift ileti kanalı aracısı, bir gönderme ve bir alma, bir kanalı oluşturan ve bir kuyruk yöneticisinden başka bir kuyruk yöneticisinden ileti taşıma.

Küme

Küme , mantıksal olarak bir şekilde ilişkili olan bir kuyruk yöneticilerinden oluşan bir ağdır.

Kümeleme olmadan dağıtılmış kuyruklama özelliğini kullanan bir IBM MQ ağında, her kuyruk yöneticisi bağımsızdır. Bir kuyruk yöneticisinin başka birine ileti göndermesi gerekiyorsa, uzak kuyruk yöneticisine bir iletim kuyruğu ve bir kanal tanımlamış olmalıdır.

Kümeleri kullanmanın iki farklı nedeni vardır: Sistem yönetimini azaltmak ve kullanılabilirliği ve iş yükü dengelemesini geliştirmek için.

En küçük kümeyi oluşturur kurmaz, basitleştirilmiş sistem yönetiminden yararlanırsınız. Bir kümenin parçası olan kuyruk yöneticileri daha az tanımlara gereksinim duyar ve bu nedenle tanımlarınızda hata yapma riskinin azaltılması riskine neden olur.

Kümelemeye ilişkin ek bilgi için Kümelerbaşlıklı konuya bakın.

z/OS Paylaşılan kuyruk

Paylaşılan kuyruk , bir sistem kuyruğunda bulunan bir ya da daha çok kuyruk yöneticisi tarafından erişilebilen ve iletileri içeren bir yerel kuyruk türüdür. Aynı kuyruk yöneticisi kullanılarak, kuyruğun birden çok uygulama tarafından paylaşıldığı bir kuyruk aynı değil. Bu, yalnızca IBM MQ for z/OSiçin geçerlidir.

z/OS Kuyruk paylaşım grubu

Aynı paylaşılan kuyruk kümesine erişebilen kuyruk yöneticileri, bir *kuyruk paylaşım grubu* (QSG) adı verilen bir gruba erişir. Bunlar, paylaşılan kuyrukları saklayan bir bağlaşım olanağı (CF) ile birbirleriyle iletişim kurarlar. Bu, yalnızca IBM MQ for z/OSiçin geçerlidir. Kuyruk paylaşım gruplarıyla ilgili daha fazla bilgi için bkz. “Paylaşılan kuyruklar ve kuyruk paylaşım grupları” sayfa 160 .

z/OS Grup içi kuyruğa alma

Kuyruk paylaşım grubundaki kuyruk yöneticileri normal kanalları kullanarak iletişim kurabilir ya da kanal tanımlamadan hızlı ileti aktarımı gerçekleştirmenizi sağlayan *grup içi kuyruğa alma* (IGQ) adı verilen bir teknik kullanabilirsiniz. Bu, yalnızca IBM MQ for z/OSiçin geçerlidir.

IBM MQ MQI client

IBM MQ MQI *istemcileri* , IBM MQ' un bağımsız olarak kurulabilen bileşenleridir. Bir MQI istemcisi, diğer platformlardaki bir ya da daha çok Message Queue Interface (MQI) sunucusu ile etkileşimde bulunmak ve kuyruk yöneticilerine bağlanmak için IBM MQ uygulamalarını bir iletişim protokolüyle çalıştırmanızı sağlar.

IBM MQ MQI client bileşenlerinin nasıl kurulacağı ve kullanılacağı ile ilgili ayrıntılı bilgi için aşağıdaki konulara bakın:

- AIX AIXüzerinde bir IBM MQ istemcisi kurulması
- Linux Linux®üzerinde bir IBM MQ istemcisi kurulması
- Windows Windowsüzerinde bir IBM MQ istemcisi kurulması
- IBM i IBM iüzerinde bir IBM MQ istemcisi kurulması

ve Sunucu ile istemci arasındaki bağlantıların yapılandırılması.

Noktadan Noktaya İleti Sistemi

Noktadan noktaya ileti alışverisinde, her ileti bir uygulamadan tek bir uygulamayı tüketmek için hareket eder. İletiler, bir kuyruğa ileti koymak için uygulama üreten uygulama aracılığıyla aktarılır ve alıcı uygulama bu iletileri o kuyruktan alır.

Yayınlama/abone olma ileti alışverişi

Yayınlama/abone olma mesajlarında, yayınlama uygulaması tarafından yayınlanan her iletinin bir kopyası her ilgili uygulamaya teslim edilir. Bir çok kişi olabilir, bir ya da daha çok ilgili uygulama olabilir. Yayınlama/abone olma ilgili bir uygulamaya abone olarak bilinir ve iletiler, abonelik tarafından tanımlanan bir kuyrukta kuyruğa alınır. Daha fazla bilgi için “Yayınlama/abone olma ileti alışverişi” sayfa 59 başlıklı konuya bakın.

Konu

Konu, yayınlama/abone olma iletisinde yayınlanan bilgilerin konusunu açıklayan bir karakter dizgisidir.

Konular, bir yayınlama/abone olma sisteminde iletilerin başarıyla tesliminin anahtınıdır. Her iletide belirli bir hedef adresi kullanmak yerine, bir yayınlayıcı her iletiye bir konu atar. Kuyruk yöneticisi, bu konuya abone olan abonelerin listesiyle eşleşir ve bu abonelerin her birine ileti gönderir.

Abonelik

Bir yayınlama/abone olma uygulaması, belirli konularla ilgili iletilere ilgi kaydedebilir. Bir uygulama bunu yaptığında, bu bir abone olarak bilinir ve abonelik süresi, eşleşen iletilerin nasıl işlenmek üzere kuyruğa alınacağını tanımlar.

Abonelik, abonenin kimliği ve yayınların yerleştirileceği hedef kuyruğun kimliği hakkında bilgi içerir. Ayrıca, bir yayının hedef kuyruğa nasıl yerleştirileceği ile ilgili bilgiler de içerir.

İletiler ve kuyruklar

İletiler ve kuyruklar, ileti kuyruğa alma sisteminin temel bileşenleridir.

Mesaj nedir?

İleti , onu kullanan uygulamalar için anlamlı olan bir bayt dizesidir. İletiler, bir uygulama programından başka bir uygulamaya (ya da aynı uygulamanın farklı bölümleri arasında) bilgi aktarmak için kullanılır. Uygulamalar aynı platformda ya da farklı platformlarda çalıştırılabilir.

Bir IBM MQ iletisi şu şekilde oluşur:

- *Uygulama verileri.*Uygulama verilerinin içeriği ve yapısı, bunu kullanan uygulama programları tarafından tanımlanır.
- *İleti tanımlayıcısı.*İleti tanımlayıcı, iletiyi tanıtır ve ileti tipi ve gönderme uygulaması tarafından iletiye atanan öncelik gibi ek denetim bilgileri içerir.

  İleti tanımlayıcısının biçimi IBM MQtarafından tanımlanır. İleti tanımlayıcısının tam açıklaması için bakınız: MQMD-Message descriptor.
- *İleti özellikleri.* İletiye ilişkin meta veriler. İleti özelliklerinin içeriği, bunları kullanan uygulama programları tarafından tanımlanır. Ek bilgi için İleti özellikleribaşlıklı konuya bakın.

İleti uzunlukları

Bu değeri en fazla 100 MB (1 MB eşittir 1 048 576 bayt) uzunluğuna çıkarabilmeniz için varsayılan ileti uzunluğu üst sınırı 4 MB 'dir. Uygulamada, ileti uzunluğu şu şekilde sınırlanabilir:

- Alma kuyruğu için tanımlanan ileti uzunluğu üst sınırı
- Kuyruk yöneticisi için tanımlanan ileti uzunluğu üst sınırı
- Kuyruk tarafından tanımlanan ileti uzunluğu üst sınırı
- Gönderme ya da alma uygulaması tarafından tanımlanan ileti uzunluğu üst sınırı
- İletiye ilişkin kullanılabilir depolama miktarı

Bir uygulamanın gerektirdiği tüm bilgileri göndermek için birkaç ileti alabilir.

Uygulamalar ileti gönderme ve alma işlemi nasıl yapılır?

Uygulama programları, iletileri **MQI çağrıları**kullanarak gönderir ve alır.

Örneğin, bir iletiyi bir kuyruğa koymak için bir uygulama:

1. Bir MQI `MQOPEN` çağrısı yayınlayarak gerekli kuyruğu açar
2. İletiyi kuyruğa koymak için bir MQI `MQPUT` çağrısı yayınlar.

Başka bir uygulama, bir MQI `MQGET` çağrısı yayınlayarak, iletiyi aynı kuyruktan alabilir

MQI çağrılarına ilişkin ek bilgi için bkz. MQI çağrıları.

Kuyruk nedir?

Kuyruk , iletileri saklamak için kullanılan bir veri yapısıdır.

Her kuyruğun sahibi bir *kuyruk yöneticisi*' dir. Kuyruk yöneticisi, sahip olduğu kuyrukları korumaktan ve aldığı tüm iletilerin uygun kuyruklara saklanmasını sağlamaktan sorumludur. İletiler, uygulama programları tarafından ya da olağan işletmesinin bir parçası olarak kuyruk yöneticisi tarafından kuyruğa konabilir.

Önceden tanımlanmış kuyruklar ve dinamik kuyruklar

Kuyruklar, oluşturulma biçimleriyle karakterize edilebilir:

- **Önceden Tanımlı Kuyruk** , uygun MQSC ya da PCF komutlarını kullanan bir yönetici tarafından oluşturulur.Önceden tanımlanmış kuyruklar kalıcıdır; bu kuyruklar, bunları kullanan uygulamalardan bağımsız olarak varolur ve IBM MQ yeniden başlatılırlar.
- **Dinamik kuyruklar** bir uygulama tarafından yaratıldığında oluşturulur Bir *model kuyruğunun*adını belirten bir MQOPEN isteği yayınlar. Yaratılan kuyruk, model kuyruğu adı verilen bir *şablon kuyruğu tanımlamasına*dayalıdır.MQSC komutunu kullanarak bir model kuyruğu yaratabilirsiniz (QModel TANıMı). Bir model kuyruğunun öznitelikleri (örneğin, üzerinde saklanabilen ileti sayısı üst sınırı), bu kuyruktan yaratılan herhangi bir dinamik kuyruk tarafından devralınır.

  Model kuyrukları, dinamik kuyruğun kalıcı mı, yoksa geçici mi olacağını belirten bir özniteliğe sahiptir. Kalıcı kuyruklar, uygulama ve kuyruk yöneticisini yeniden başlatır; geçici kuyruklar yeniden başlatıldığında kaybedilir.

Kuyruklardan iletilerin alınması

Uygun olarak yetkili uygulamalar, aşağıdaki alma algoritmalarına göre bir kuyruktan ileti alabilir:

- İlk giren ilk çıkış (FIFO).
- İleti tanımlayıcısında tanımlandığı gibi, ileti önceliği. Aynı önceliğe sahip iletiler FIFO esasına dayalı olarak alınır.
- Belirli bir ileti için program isteği.

Kullanılan yöntemi, uygulamadaki MQGET isteği belirler.

IBM MQ nesneler

Kuyruk yöneticileri IBM MQ nesnelerinin özelliklerini tanımlar. Bu özelliklerin değerleri, IBM MQ ' in bu nesneleri işlemesinin yolunu etkiler. You create and manage objects using IBM MQ commands and interfaces. Uygulamalarınızdan, nesneleri denetlemek için Message Queue Interface (MQI) olanağını kullanıyorsunuz. Nesneler, bir programdan adreslendiklerinde bir IBM MQ *nesne tanımlayıcısı* (MQOD) ile tanımlanır.

Nesnelerin yönetimi aşağıdaki görevleri içerir:

- Kuyruk yöneticilerinin başlatılması ve durdurulması.
- Uygulamalar için, başta kuyruklar olmak üzere nesneler yaratılmasına neden olur.
- Nesnelerin özniteliklerinin görüntülenmesi ya da değiştirilmesi.
- Nesneler siliniyor.
- Diğer (uzak) sistemlerdeki kuyruk yöneticilerine iletişim yolları oluşturmak için kanallarla çalışma.
- Genel yönetim sürecini basitleştirmek ve iş yükünü dengelemek için kuyruk yöneticileri için *kümeler* yaratılması.

Dinamik kuyruklar dışında, nesneler üzerinde çalışabilmeniz için kuyruk yöneticisine nesneler tanımlanmalıdır.

Bir nesne denetimi işlemini gerçekleştirmek için bir IBM MQ komutu kullandığınızda, kuyruk yöneticisi, işlemi gerçekleştirmek için gereken yetki düzeyine sahip olup olmadığını denetler. Benzer şekilde, bir

uygulama bir nesneyi açmak için MQOPEN çağrısını kullandığında, kuyruk yöneticisi, uygulamanın o nesneye erişime izin verebilmesi için gerekli düzeyde yetki olduğunu denetler. Çekler, açılmakta olan nesnenin adı üzerinde yapılır.

Aşağıdaki yöntemleri kullanarak nesneleri tanımlayabilir ve yönetebilirsiniz:

- The PCF commands described in Programlanabilir komut biçimleri başvurusu and Yönetim görevlerinin otomatikleştirilmesi
- MQSC komutlarında açıklanan MQSC komutları
- z/OS Operating IBM MQ for z/OS içinde açıklanan IBM MQ for z/OS işlemleri ve denetim panoları
- Linux Windows IBM MQ Explorer (yalnızca Intel sistemleri için Windows ve Linux). Daha fazla bilgi için bakınız: Introduction to MQ Explorer.

Nesneleri aşağıdaki yöntemleri kullanarak da yönetebilirsiniz:

- Bir klavyeden yazılan denetim komutları. Bkz. Denetim komutlarını kullanarak denetim.
- Bir programdaIBM MQ Administration Interface (MQAI) çağrıları vardır. Bkz. IBM MQ Yönetim Arabirimi (MQAI).

z/OS Düzenli olarak kullandığınız IBM MQ for z/OS komutlarının sıraları için, komutları içeren iletileri yaratan ve bu iletileri sistem komut giriş kuyruğuna koyan yönetim programları yazabilirsiniz. Kuyruk yöneticisi, bu kuyrukta yer alan iletileri, komut satırından ya da işlem ve denetim panolarından girilen komutları işleme yöntemiyle aynı şekilde işler. This technique is described in the Writing programs to administer IBM MQ, and demonstrated in the Mail Manager sample application delivered with IBM MQ for z/OS. Bu örneğe ilişkin açıklamalar için Sample Programs for IBM MQ for z/OS başlıklı konuya bakın.

IBM i For sequences of IBM MQ for IBM i commands that you use regularly you can write CL programs. Ek bilgi için Denetim dili (CL) komutlarını kullanarak IBM MQ for IBM i ' in yönetilmesibaşlıklı konuya bakın.

ALW AIX, Linux, and Windowskomutundaki IBM MQ komutlarının sıraları için, bir dosyada tutulan bir dizi komutu çalıştırmak için MQSC olanağını kullanabilirsiniz. Ek bilgi için MQSC komutlarını kullanarak yönetimbaşlıklı konuya bakın.

Nesne tipleri

Yönetim görevlerinin çoğu çeşitli IBM MQ *nesneleri*tiplerini kurcalayanlarla ilgilidir.

IBM MQ nesnelerine adlandırma hakkında bilgi için bkz. “IBM MQ nesnelerini adlandırma” sayfa 33.

Kuyruk yöneticinde yaratılan varsayılan nesnelerle ilgili bilgi için bkz. “Sistem varsayılan nesneleri” sayfa 40.

For information about the different types of IBM MQ objects, see the following subtopics:

İlgili kavramlar

“İletinin kuyruğa alınması” sayfa 5
IBM MQ ürünleri, programları tutarlı bir uygulama programlama arabirimi kullanarak, bileşenlerin (işlemciler, işletim sistemleri, altsistemler ve iletişim protokolleri) farklı bir ağ üzerinden bir diğeriyle iletişim kurmasını sağlar.

“Nesne öznitelikleri” sayfa 39
Bir nesnenin özellikleri, bir nesnenin öznitelikleri tarafından tanımlanır. Bazı kişiler, yalnızca görüntüleyebileceğiniz diğer kişileri de belirleyebilirsiniz.

İlgili başvurular

MQSC komutları

Kuyruklar

IBM MQ kuyruklarına ve kuyruk özniteliklerine giriş.

IBM MQ *kuyruğu* , uygulamaların iletileri koyabileceği ve uygulamaların iletileri alabileceği adlandırılmış bir nesnedir.

İletiler bir kuyruğunda saklanır; bu nedenle, uygulama koyma işlemi iletisine yanıt bekliyorsa, bu yanıtı beklerken başka bir iş yapmak için serbest olur. Uygulamalar, İleti Kuyruğu Arabirimi-Genel Bakışalanında açıklanan, Message Queue Interface (MQI) olanağını kullanarak bir kuyruğa erişir.

Bir iletinin kuyruğa konabilmesi için, kuyruğun önceden yaratılmış olması gerekir. Kuyruğun sahibi bir kuyruk yöneticisi ve kuyruk yöneticisi birçok kuyruklara sahip olabilir. Ancak, her kuyruğun, o kuyruk yöneticisi içinde benzersiz bir adı olmalıdır.

Kuyruk, kuyruk yöneticisi aracılığıyla sürdürür. Çoğu durumda, her kuyruk, kuyruk yöneticisi tarafından fiziksel olarak yönetilir, ancak bu bir uygulama programı için belirgin değildir. IBM MQ for z/OS paylaşılan kuyrukları, kuyruk paylaşım grubunda bulunan herhangi bir kuyruk yöneticisi tarafından yönetilebilir.

Kuyruk yaratmak için IBM MQ komutlarını (MQSC), PCF komutlarını ya da altyapıya özgü arabirimleri kullanabilirsiniz. Örneğin, IBM MQ for z/OS işlemleri ve denetim panoları platforma özeldir.

Uygulamanıza geçici olarak *dinamik olarak* geçici işler için yerel kuyruklar yaratabilirsiniz. Örneğin, *yanıtlama* kuyrukları oluşturabilirsiniz (uygulama sona erdikten sonra gerekli değildir). Daha fazla bilgi için “Dinamik ve Model kuyrukları” sayfa 19 başlıklı konuya bakın.

Bir kuyruğu kullanmadan önce, kuyrukla ne yapmak istediğinizi belirtme işlemi yapmanız gerekir. Örneğin, aşağıdakiler için bir kuyruk açabilirsiniz:

- Yalnızca iletilere göz atma (bunları almama)
- İletiler alınıyor (ve erişimi diğer programlarla paylaşır ya da dışlayıcı erişim ile paylaşır)
- Kuyruğa ileti koyma
- Kuyruğun öznitelikleriyle ilgili hata
- Kuyruğun özniteliklerinin ayarlanması

Bir kuyruğu açtığınızda belirleyebileceğiniz seçeneklerin tam listesi için MQOPEN-Open objectbaşlıklı konuya bakın.

Kuyrukların öznitelikleri

Kuyruk tanımlandığında bir kuyruğun özniteliklerinden bazıları belirtilir ve daha sonra değiştirilemez (örneğin, kuyruğun tipi). Kuyrukların diğer öznitelikleri, değiştirilebilecek olan özniteliklere göre gruplandırılabilir:

- Kuyruğun işlenmesi sırasında kuyruk yöneticisi tarafından (örneğin, bir kuyruğun yürürlükteki derinliği)
- Yalnızca komutlar temelinde (örneğin, kuyruğun metin tanımlaması)
- Uygulamalar temelinde, MQSET çağrısını kullanarak (örneğin, kuyruğa alma işlemlerine izin verilip verilmediğini)

MQINQ çağrısını kullanarak tüm özniteliklere ilişkin değerleri bulabilirsiniz.

Birden çok kuyruk tipi için ortak olan öznitelikler şunlardır:

QName
Kuyruğun adı.

QType
Kuyruğun tipi.

QDesc
Kuyruğun metin tanımlaması.

InhibitGet
Programların kuyruktan ileti almalarına izin verilip verilmeyeceğini belirleyin. Ancak, uzak kuyruklardan hiçbir zaman ileti alamıyor olabilirsiniz.

InhibitPut
Programların kuyruğa ileti koymasına izin verilip verilmeyeceğini belirleyin.

DefPriority
Kuyruğa gönderilen iletiler için varsayılan öncelik değeri.

DefPersistence
Kuyruğa konulan iletiler için varsayılan kalıcılık

Kapsam
Bu kuyruğa ilişkin bir girişin bir ad hizmetinde de bulunup bulunmadığını denetler.

z/OS **Scope** özniteliği z/OSüzerinde desteklenmez.

Bu özniteliklere ilişkin eksiksiz açıklamalar için Kuyruklar için özniteliklerbaşlıklı konuya bakın.

İlgili kavramlar

z/OS Paylaşılan kuyruklar
Paylaşılan kuyruk, yerel bir kuyruğun tipidir. Bir ya da daha çok kuyruk yöneticisi tarafından bir sistem birleşimi (sysplex) içinde olan iletiler bu kuyruğa erişebilir.

"Küme kuyrukları" sayfa 55
Küme kuyruğu, bir küme kuyruk yöneticisi tarafından barındırılan ve kümedeki diğer kuyruk yöneticilerinin kullanımına sunulan bir kuyruktur.

"Ölü-harfli kuyruklar" sayfa 47
Gönderilmeyen iletiler kuyruğu (ya da teslim edilemeyen ileti kuyruğu), iletilerin doğru hedefe yönlendirilememesi durumunda gönderileceği kuyruktır. Her kuyruk yöneticisinin tipik olarak bir ölü-mektup kuyruğu vardır.

İlgili görevler
Uygulama başvurusu geliştirilmesi
İlgili başvurular
MQSC komutları
"Paylaşılan kuyruklar ve küme kuyrukları arasındaki karşılaştırma" sayfa 55
Bu bilgiler, paylaşılan kuyrukları ve küme kuyruklarını karşılaştırmanıza ve sisteminize daha uygun olmaya karar vermenize yardımcı olacak şekilde tasarlanmıştır.

Yerel kuyruklar
İletim, başlatma, dead-letter, komut, varsayılan, kanal ve olay kuyrukları, yerel kuyruk tipleridir.

Programın bağlı olduğu kuyruk yöneticisine aitse, bir program kuyruğa *yerel* olarak bilinir. Yerel kuyruklardan ileti alabilir ve yerel kuyruklara ileti yerleştirebilirsiniz.

Kuyruk tanımlaması nesnesi, kuyruğa konan fiziksel iletilerin yanı sıra, kuyruğun tanımlama bilgilerini de içerir.

Her kuyruk yöneticisi, özel amaçlar için kullandığı bazı yerel kuyruklara sahip olabilir:

İletim kuyrukları

Bir uygulama uzak bir kuyruğa ileti gönderdiğinde, yerel kuyruk yöneticisi iletiyi *iletim kuyruğu*adı verilen özel bir yerel kuyrukta saklar. Uygulamalar, iletileri doğrudan bir iletim kuyruğuna ya da bir uzak kuyruk tanımlamasıyla dolaylı olarak gönderebilirler.

Kuyruk yöneticisi uzak bir kuyruk yöneticisine ileti gönderdiğinde, aşağıdaki sırayı kullanarak iletim kuyruğunu tanımlar:

1. Uzak bir kuyruğun yerel tanımlamasının XMITQ özniteliğe adı geçen iletim kuyruğu.

2. Uzak kuyruk yöneticisiyle aynı adı taşıyan bir iletim kuyruğu. Bu değer, uzak bir kuyruğa ilişkin yerel tanımlamanın XMITQ üzerindeki varsayılan değerdir.
3. Yerel kuyruk yöneticisinin DEFXMITQ özniteliğe adı geçen iletim kuyruğu.

İleti kanalı aracısı , iletim kuyruğuyla ilişkilendirilmiş bir kanal programıdır ve iletiyi sonraki hedefine aktarır. Sonraki hedef, ileti kanalının bağlandığın kuyruk yöneticidir. İletinin son hedefiyle aynı kuyruk yöneticisi olması gerekmez. İleti bir sonraki hedefine teslim edildiğinde, ileti iletim kuyruğundan silinir. İletinin, yolculuğunda son varış noktasına kadar birçok kuyruk yöneticisini geçmesi gerekebilir. Rota boyunca her kuyruk yöneticisinde bir iletim kuyruğu tanımlamanız gerekir. Her bir tutma iletisi bir sonraki hedefe iletilmeyi bekler. Normal iletim kuyruğu, iletilerin farklı varış noktalarına sahip olabilmesine rağmen, sonraki hedefe ilişkin iletileri tutar. Bir küme iletim kuyruğu, birden çok hedef için ileti bulundurur. Her iletinin `correlID` değeri, iletinin bir sonraki hedefine aktarmak üzere yerleştirileceği kanalı tanıtır.

Bir kuyruk yöneticisinde birden çok iletim kuyruğu tanımlayabilirsiniz. Aynı hedef için, her biri farklı bir hizmet sınıfı için kullanılmakta olan birkaç iletim kuyruğu tanımlayabilirsiniz. Örneğin, küçük iletiler ve aynı hedefe giden büyük iletiler için farklı iletim kuyrukları yaratmak isteyebilirsiniz. Daha sonra, büyük iletilerin daha küçük iletileri tutmayabilmesi için iletileri farklı ileti kanallarını kullanarak aktarabilirsiniz. Küme kuyruklarına ya da küme konularına ilişkin tüm iletiler, varsayılan olarak tek bir küme iletim kuyruğu `SYSTEM.CLUSTER.TRANSMIT.QUEUE`üzerine yerleştirilir. Bir seçenek olarak, varsayılan değeri değiştirebilir ve farklı küme kuyruğu yöneticilerine giden ileti trafiğini farklı küme iletim kuyruklarına ayırabilirsiniz. Kuyruk yöneticisi özniteliğini `DEFCLXQ` olarak `CHANNEL`olarak ayarlıyorsanız, her bir küme gönderici kanalı ayrı bir küme iletim kuyruğu oluşturur. Diğer bir seçenek olarak, küme gönderici kanallarına ilişkin küme iletim kuyruklarını el ile tanımlayabilirsiniz.

İletim kuyrukları, ileti göndermek için bir ileti kanalı aracısını tetikleyebilir; bkz. Tetikleyicileri kullanarak IBM MQ uygulamalarını başlatma.

z/OS IBM MQ for z/OS' ta, grup içi kuyruğa alma kullanıyorsanız, iletim kuyruğuna *grup içi kuyruklama aracısı*tarafından hizmet verilmektedir. IBM MQ for z/OSüzerinde grup içi kuyruğa alma kullanılırken, paylaşılan bir iletim kuyruğu kullanılır.

Başlatma kuyrukları

Kullanıma hazırlama kuyruğu , kuyruk yöneticisinin bir uygulama kuyruğunda tetikleme olayı oluştuğunda bir tetikleme iletisi yerleştirdiği yerel bir kuyruktır.

Tetikleme olayı, bir programın kuyruğun işlenmesini başlatması için amaçlanan bir olaydır. Örneğin, 10 'dan fazla ileti varolabilir. Tetikleme çalışmalarıyla ilgili daha fazla bilgi için Tetikleyicileri kullanarak IBM MQ uygulamalarının başlatılmasıbaşlıklı konuya bakın.

Dead-letter (teslim edilmemiş ileti) kuyruğu

Edilmiş harf (teslim edilmemiş ileti) kuyruğu , kuyruk yöneticisinin gönderemediği iletileri koyduğu yerel bir kuyruğdur.

Kuyruk yöneticisi, ileti kuyruğuna bir ileti yerleştirdiğinde, iletiye bir üstbilgi ekler. Üstbilgi bilgileri, kuyruk yöneticisinin iletiyi ölü harf kuyruğuna koymasının nedenlerini içerir. Ayrıca, özgün iletinin hedefi, tarih ve kuyruk yöneticisinin iletiyi ölü harf kuyruğuna yerleştirdiği tarih de yer alır.

Uygulamalar, teslim edemedikleri iletiler için de kuyruğu kullanabilir. Daha fazla bilgi için bkz. Ölülerin harfini kullanma (teslim edilmemiş ileti) kuyruğu.

Sistem komut kuyruğu

Sistem komut kuyruğu , uygun şekilde yetkili uygulamaların IBM MQ komutlarını gönderebileceği bir kuyruktır. Bu kuyruklar, altyapınızda desteklendiği şekilde, kuyruk yöneticisine ilişkin işlemleri yapmak üzere hazır duruma hazır olarak, PCF, MQSC ve CL komutlarını alır.

z/OS On IBM MQ for z/OS the queue is called SYSTEM.COMMAND.INPUT ; on other platforms it is called SYSTEM.ADMIN.COMMAND.QUEUE. Kabul edilen komutlar altyapıya göre değişir. Ayrıntılar için bkz. Programlanabilir komut biçimleri başvurusu.

Sistem varsayılan kuyrukları

sistem varsayılan kuyrukları , sisteminize ilişkin kuyrukların başlangıç tanımlamalarını içerir. Bir kuyruk tanımlaması yarattığınızda, kuyruk yöneticisi tanımlamayı uygun sistem varsayılan kuyruğundan kopyalar. Kuyruk tanımlaması yaratılması, devingen bir kuyruk yaratmaktan farklıdır. Dinamik kuyruğun tanımlaması, dinamik kuyruk için şablon olarak seçtiğiniz model kuyruğuna dayalıdır.

Olay kuyrukları

Olay kuyrukları olay iletilerini tutar. Bu iletiler kuyruk yöneticisi ya da bir kanal tarafından raporlanır.

Uzak kuyruklar

Bir programa, programın bağlı olduğu bir kuyruk yöneticisi farklı bir kuyruk yöneticisine aitse, kuyruk *uzak* olur.

Bir iletişim bağının kurulduğu durumlarda, bir program uzak bir kuyruğa ileti gönderebilir. Bir program, uzak kuyruktan ileti alamayabilir.

Bir uzak kuyruk tanımladığınızda yaratılan kuyruk tanımlaması nesnesi, yalnızca yerel kuyruk yöneticisi için iletinizin gitmesini istediğiniz kuyruğu bulmanız için gereken bilgileri içerir. Bu nesne, *uzak kuyruğun yerel tanımlaması*olarak bilinir. Uzak kuyruğun tüm öznitelikleri, o kuyruk yöneticisine yerel bir kuyruk olduğu için sahip olan kuyruk yöneticisi tarafından tutulur.

Uzak kuyruğu açarken, kuyruğun saptanması için aşağıdakilerden birini belirlemelisiniz:

- Uzak kuyruğu tanımlayan yerel tanımın adı. Bir uygulamanın bakış açısıyla bu, yerel bir kuyruğun açılması ile aynıdır. Bir uygulamanın yerel ya da uzak bir kuyruğun olup olmadığını bilmesi gerekmez.

  IBM idışındaki tüm altyapılarda uzak bir kuyruğun yerel tanımlamasını yaratmak için DEFINE QREMOTE komutunu kullanın.

  IBM i IBM iüzerinde, CRTMQMQ komutunu kullanın.
- Uzak kuyruk yöneticisinin adı ve uzak kuyruk yöneticisi tarafından bilindiği gibi, kuyruğun adı.

"Kuyrukların öznitelikleri" sayfa 15içinde açıklanan ortak özniteliklere ek olarak, uzak kuyruklara ilişkin yerel tanımların üç özniteliği vardır. Bu üç öznitelik şunlardır:

RemoteQName
Kuyruğun sahip olduğu kuyruk yöneticisinin adını bildiği ad.

RemoteQMgrAdı
Sahip olan kuyruk yöneticisinin adı.

XmitQName
İletileri diğer kuyruk yöneticilerine iletirken kullanılan yerel iletim kuyruğunun adı.

Bu özniteliklere ilişkin ek bilgi için Kuyruklar için özniteliklerbaşlıklı konuya bakın.

Bir uzak kuyruğun yerel tanımlaması için MQINQ çağrısını kullanırsanız, kuyruk yöneticisi yalnızca yerel tanımın özniteliklerini döndürür; uzak kuyruk adı, uzak kuyruk yöneticisi adı ve iletim kuyruğu adı, uzak sistemdeki eşleşen yerel kuyruğun öznitelikleri değil.

Ayrıca bkz. İletim kuyrukları.

Diğer Adlar

Diğer ad kuyruğu , başka bir kuyruğa ya da konuya erişmek için kullanabileceğiniz bir IBM MQ nesnesidir. Başka bir deyişle, birden çok program aynı kuyrukla çalışabiliyor ve farklı adlar kullanarak bu programa erişebiliyor.

Temel kuyruk olarak bilinen bir diğer ad adının çözümünden kaynaklanan kuyruk. platformun desteklediği gibi aşağıdaki kuyruk türlerinden herhangi biri olabilir:

- Yerel kuyruk
- Uzak kuyruğun yerel tanımlaması.
- z/OS Yalnızca IBM MQ for z/OSüzerinde kullanılabilir olan bir yerel kuyruk tipi olan paylaşılan bir kuyruk.
- Önceden tanımlanmış bir kuyruk
- Dinamik kuyruk

Diğer ad, bir konuya da çözümleyebilir. Bir uygulama şu anda iletileri bir kuyruğa yerleştiriyorsa, kuyruğun diğer adı konu için bir diğer ad yapılarak bir konuya yayınlanması için bu uygulama yapılabilir. Uygulama kodunda değişiklik yapılması gerekli değil.

Not: Bir diğer ad, aynı kuyruk yöneticisininde başka bir diğer ada doğrudan çözümlenemez.

Diğer ad kuyruklarının bir örneği, bir sistem denetimcisinin temel kuyruk adına (yani, diğer ad çözümleyicilerinin) ve diğer ad kuyruğu adına farklı erişim yetkileri vermeleri için kullanılır. Bu, bir programın ya da kullanıcının diğer ad kuyruğunu kullanma yetkisi verilebileceği, ancak temel kuyruğun kullanılmasına izin verilmediği anlamına gelir.

Diğer bir seçenek olarak, yetki, diğer ad için yapılan işlemleri engellemek üzere ayarlanabilir, ancak temel kuyruk için bunlara izin verir.

Bazı uygulamalarda, diğer ad kuyruklarının kullanılması, sistem denetimcilerinin bir diğer ad kuyruğu nesnesinin tanımlamasını, uygulamanın değiştirilmesine gerek kalmadan kolayca değiştirebilmesini sağlar.

IBM MQ , programlar bu adı kullanmayı denediğinde diğer ad için yetkilendirme denetimlerini yapar. Bu, programın diğer adın çözdüğü ada erişme yetkisine sahip olup olmadığını denetmez. Bu nedenle, bir program diğer ad kuyruğu adına erişim yetkisine sahip olabilir, ancak çözülen kuyruk adı değil.

“Kuyruklar” sayfa 15' ta açıklanan genel kuyruk özniteliklerine ek olarak, diğer ad kuyruklarında bir **BaseQName** özniteliği vardır. Bu ad, diğer ad adının çözümlediği temel kuyruğun adıdır. Bu özniteliğin yerine getirilmesi için bakınız: BaseQName (MQCHAR48).

The *InhibitGet* and **InhibitPut** attributes (see “Kuyruklar” sayfa 15) of alias queues belong to the alias name. Örneğin, ALIAS1 diğer ad kuyruğu adı BASE, temel kuyruk adı BASE 'e çözülürse, ALIAS1 üzerindeki engellemeler yalnızca ALIAS1 ' i etkiler ve BASE engellenmez. Ancak, BASE üzerinde engellemeler de ALIAS1' i etkiler.

DefPriority ve **DefPersistence** öznitelikleri de diğer ad adına aittir. Bu nedenle, örneğin, aynı temel kuyruğun farklı diğer adlarına farklı varsayılan öncelikler atayabilirsiniz. Ayrıca, diğer adları kullanan uygulamaları değiştirmek zorunda kalmadan bu öncelikleri değiştirebilirsiniz.

Dinamik ve Model kuyrukları

Bu bilgiler, dinamik kuyruklara, geçici ve kalıcı dinamik kuyruklara ilişkin özellikler, dinamik kuyrukların kullanımı, dinamik kuyruklar kullanılırken dikkate alınması gereken bazı noktalar ve model kuyrukları için bir kavrayış sağlar.

Bir uygulama programı bir model kuyruğunu açmak için bir MQOPEN çağrısı yayınlarken, kuyruk yöneticisi model kuyrukla aynı özniteliklere sahip bir yerel kuyruğun somut örneğini dinamik olarak yaratır. Kuyruk yöneticisi, model kuyruğunun *DefinitionType* alanının değerine bağlı olarak, geçici ya da kalıcı bir dinamik kuyruk yaratır (Bkz. Dinamik kuyruklar oluşturma).

Geçici dinamik kuyrukların özellikleri

Geçici dinamik kuyruklar aşağıdaki özelliklere sahiptir:

- z/OS Bunlar, bir kuyruk paylaşım grubundaki kuyruk yöneticilerinden erişilebilen, kuyruklar paylaşılamaz.

  Kuyruk paylaşım gruplarının yalnızca IBM MQ for z/OSüzerinde kullanılabilir olduğunu unutmayın.
- Yalnızca kalıcı olmayan iletiler tutuyorlar.
- Onlar kurtarılamaz.

- Bunlar, kuyruk yöneticisi başlatıldığında silinir.
- Kuyruk oluşturan MQOPER çağrısını yayınlayan uygulama kuyruğu kapatır ya da sona erdirildiğinde silinir.
  - Kuyrukda kesinleştirilmiş ileti varsa, bunlar silinir.
  - Kesinleştirilmemiş bir MQGET, MQPUT ya da MQPUT1 çağrıları varsa, bu sırada kuyruğa karşı olağanüstü durum çağrılır; kuyruk, mantıksal olarak silindiği şekilde işaretlenir ve kapatma işleminin bir parçası olarak ya da uygulama sona erdiğinde fiziksel olarak silinir (bu çağrılar kesinleştirildikten sonra).
  - Kuyruk şu anda kullanılıyorsa (yaratma işlemi ya da başka bir uygulama tarafından), kuyruk mantıksal olarak silindiği için işaretlenir ve kuyruğu kullanan son uygulama tarafından kapatıldığında yalnızca fiziksel olarak silinir.
  - Mantıksal olarak silinen bir kuyruğa erişme girişimi (kapatması dışında) başarısız oldu; neden kodu MQRC_Q_DELETED.
  - MQCO_NONE, MQCO_DELETE ve MQCO_DELETE_PURGE, kuyruğu yaratan ilgili MQOPER çağrısına ilişkin MQCLOOSE çağrısında belirtildiğinde, tümü MQCO_NONE olarak kabul edilir.

Kalıcı dinamik kuyrukların özellikleri

Kalıcı dinamik kuyruklar aşağıdaki özelliklere sahiptir:

- Kalıcı ya da kalıcı olmayan iletiler düzenliyorlar.
- Bunlar, sistem arızaları durumunda kurtarılabilir.
- Bunlar, bir uygulama (kuyruğu yaratan MQOPER çağrısını veren kişi olmak zorunda değilse), MQCO_DELETE ya da MQCO_DELETE_PURGE seçeneğini kullanarak kuyruğu başarıyla kapatırken silinir.
  - Kuyrukla ilgili iletiler (kesinleştirilmemiş ya da kesinleştirilmemiş) varsa, MQCO_DELETE seçeneği ile kapama isteği başarısız olur. Kuyrukta kesinleştirilmiş iletiler olsa bile, MQCO_DELETE_PURGE seçeneğiyle kapatma isteği başarılı olur, ancak işlenmemiş MQGET, MQPUT ya da MQPUT1 çağrıları varsa, kuyrukta bekleyen çağrılan iletiler varsa başarısız olur.
  - Silme isteği başarılı olursa, ancak kuyruğun kullanımda olduğu (yaratma ya da başka bir uygulama tarafından), kuyruk mantıksal olarak silindiği şekilde işaretlenir ve kuyruğu kullanan son uygulama tarafından kapatıldığında yalnızca fiziksel olarak silinir.
- Kuyruğun silinmesi için yetkisi olmayan bir uygulama tarafından kapatılırsa bunlar silinmez; ancak, kapatma uygulaması kuyruğu yaratan MQOPER çağrısını yayınlamadıysa, bunlar silinmez. İlgili MQOPEN çağrısını doğrulamak için kullanılan kullanıcı tanıtıcıyla (ya da MQOO_ALTERNATE_USER_AUTHORITY belirtildiyse diğer kullanıcı kimliği) için yetki denetimi gerçekleştirilir.
- Bunlar, normal bir kuyrukken aynı şekilde silinebilir.

Dinamik kuyrukların kullanımı

Aşağıdakiler için dinamik kuyrukları kullanabilirsiniz:

- Uygulama sona erdirildikten sonra kuyruk gerektirmeyen uygulamalar saklanmalıdır.
- Başka bir uygulama tarafından işlenecek iletilere yanıt gerektiren uygulamalar. Bu tür uygulamalar, bir model kuyruğu açarak dinamik olarak bir yanıtlama kuyruğu oluşturabilir. Örneğin, bir istemci uygulaması aşağıdaki gibi olabilir:
  1. Dinamik bir kuyruk yaratır.
  2. İstek iletisinin ileti tanımlayıcı yapısının **ReplyToQ** alanında adını belirtin.
  3. İsteği, bir sunucu tarafından işlenmekte olan bir kuyruğa yerlesin.

Daha sonra, yanıt iletisini yanıtlama kuyruğuna yerleştirebilir. Son olarak, istemci yanıtı işleyebildi ve silme seçeneğiyle yanıtlama kuyruğunu kapatma işlemi sona erebilir.

Dinamik kuyrukları kullanırken dikkat edilmesi gereken noktalar

Dinamik kuyrukları kullanırken aşağıdaki noktaları göz önünde bulundurun:

- Bir istemci-sunucu modelinde, her istemci kendi dinamik yanıtlama kuyruğunu yaratmalı ve kullanmalıdır. Dinamik bir yanıtlama kuyruğu birden çok istemci arasında paylaşılıyorsa, kuyrukta kesinleştirilmemiş etkinlik olduğu için ya da kuyruk başka bir istemci tarafından kullanımda olduğu için, yanıt kuyruğun silinmesi gecikebilir. Buna ek olarak, kuyruk mantıksal olarak silindiği için ve sonraki API istekleri için (MQCLOSE dışında) erişilemez olarak işaretlenir.
- Uygulama ortamınız, uygulamalar arasında dinamik kuyrukların paylaşılmasını gerektiriyorsa, kuyruğa ilişkin tüm etkinlik kesinleştirildiğinde, kuyruğun yalnızca kapalı (silme seçeneğiyle birlikte) kapatıldığından emin olun. Bu, son kullanıcıya göre olmalıdır. Bu, kuyruğun silinmesinin geciktirilmemesini sağlar ve mantıksal olarak silindiği için kuyruğun erişilemez olduğu dönemi en aza indirir.

Model kuyrukları

Model kuyruğu , dinamik bir kuyruk yaratırken kullandığınız bir kuyruk tanımlaması şablonundan biri.

Bir IBM MQ programından dinamik olarak yerel bir kuyruk yaratabilirsiniz; kuyruk özniteliklerine ilişkin şablon olarak kullanmak istediğiniz model kuyruğunu adlandırır. Bu noktada, yeni kuyruğun bazı özniteliklerini değiştirebilirsiniz. Ancak, **DefinitionType**değiştiremezsiniz. Örneğin, kalıcı bir kuyruk gerektiriyorsa, tanım tipi kalıcı olarak ayarlanmış bir model kuyruğu seçin. Bazı etkileşimli uygulamalar, yanıtları işledikten sonra bu kuyrukların bakımını yapmalarına gerek olmadığı için, sorgularına yanıt vermek için dinamik kuyruklar kullanabilir.

MQOPED çağrısının *nesne tanımlayıcısında* (MQOD) bir model kuyruğunun adını belirtmenizi sağlar. Kuyruk yöneticisi, model kuyruğunun özniteliklerini kullanarak, sizin için devingen olarak bir kuyruk yaratır.

Dinamik kuyruk için bir ad (tam olarak) ya da bir adın sapını (örneğin, ABC) belirleyebilir ve kuyruk yöneticisinin buna benzersiz bir rol eklemesine izin verebilir ya da kuyruk yöneticisinin sizin için tam benzersiz bir ad atamasına izin verirsiniz. Kuyruk yöneticisi adı atarsa, bu adı MQOD yapısına koyar.

Bir MQPUT1 çağrısını doğrudan bir model kuyruğuna gönderemezsiniz; ancak, bir model kuyruğu açarak yaratılmış dinamik kuyruğa MQPUT1 komutunu yayınlayabilirsiniz.

Bir model kuyruğuna karşı MQSET ve MQINQ komutu verilemez. Devingen olarak yaratılan kuyrukla karşılaştırmalı olarak, izleyen MQINQ ve MQSET çağrılarında MQOO_SORGULAMAK ya da MQOO_SET sonuçlarıyla bir model kuyruğu açılması.

Bir model kuyruğunun öznitelikleri, yerel kuyruklardan oluşan bir altkümedir. Fuller tanımlaması için Kuyruklar için özniteliklerbaşlıklı konuya bakın.

Kuyruk tanımlama yolları

IBM MQ komutlarını (MQSC), PCF komutlarını kullanarak ya da altyapıya özgü arabirimleri kullanarak bir kuyruk yaratabilirsiniz.

MQSC DEFE komutunu ya da PCF Create Queue komutunu kullanarak, kuyrukları IBM MQ olarak tanımlayabilirsiniz. Komutlar, kuyruğun tipini ve özniteliklerini belirtir. Örneğin, bir yerel kuyruk nesnesi, MQI çağrılarında kuyruğa gönderme yapan uygulamalar olduğunda ne olacağını belirten özniteliklere sahiptir. Öznitelik örnekleri şunlardır:

- Uygulamaların kuyruktan ileti alıp alamayacağı (GET etkin)
- Uygulamaların kuyruğa ileti koyup koyamayacağı (PUT etkin)
- Kuyruğa erişimin tek bir uygulamaya dışlayıcı olup olmadığı ya da uygulamalar arasında paylaşıldığı
- Kuyrukta aynı anda saklanabilen ileti sayısı üst sınırı (kuyruk derinliği üst sınırı)
- Kuyruğa konabilecek iletilerin uzunluk üst sınırı

Ayrıca, kuyruklar tanımlamak için kullanabileceğiniz, platforma özgü çeşitli arabirimler de vardır. Daha fazla bilgi için *İlgili bilgiler*başlıklı konuya bakın.

İlgili kavramlar

MQSC komutlarını kullanan yönetim

IBM MQ Console' deki kuyruklarla çalışma

İlgili görevler

PCF komutlarını kullanarak yönetim otomatikleştiriliyor

MQ Gezgini ile kuyruk yöneticisi ve nesne yaratılması ve yapılandırılması

CL komutlarını kullanarak IBM MQ for IBM i ' in yönetilmesi

IBM MQ for z/OSkomutlarına komut verilmesi

Queues used for specific purposes by IBM MQ

IBM MQ , işlemi ile ilgili belirli amaçlar için bazı yerel kuyrukları kullanır.

Bu kuyrukları, IBM MQ kullanabilmesi için önce tanımlamanız gerekir.

Başlatma kuyrukları

Başlatma kuyrukları, tetikleme sırasında kullanılan kuyruklardır. Bir kuyruk yöneticisi, bir tetikleme olayı ortaya çıktığında bir tetikleme iletisini bir başlatma kuyruğuna koyar. Tetikleme olayı, kuyruk yöneticisi tarafından saptanan koşulların mantıksal bir birleşimidir. Örneğin, bir kuyruktaki ileti sayısı önceden tanımlı bir derinliğe ulaştığında tetikleme olayı oluşturulabilir. Bu olay, kuyruk yöneticisinin belirtilen bir başlatma kuyruğuna tetikleyici ileti koymasına neden olur. Bu tetikleyici iletisi, bir başlatma kuyruğunu izleyen özel bir uygulama olan *tetikleme izleme programı*tarafından alınır. Tetikleyici izleyicisi, tetikleme iletisinde belirtilen uygulama programını başlatır.

Bir kuyruk yöneticisi tetiklemeyi kullanacaksa, o kuyruk yöneticisi için en az bir kullanıma hazırlama kuyruğu tanımlanmalıdır. Bkz. Tetikleme için nesneleri yönetme, runmqtrmve Tetikleyicileri kullanarak IBM MQ uygulamalarını başlatma

İletim kuyrukları

İletim kuyrukları, uzak kuyruk yöneticisine yazmakta olan iletileri geçici olarak saklayan kuyruklardır. Yerel kuyruk yöneticisinin iletileri doğrudan göndereceği her uzak kuyruk yöneticisi için en az bir iletim kuyruğu tanımlamanız gerekir. Bu kuyruklar uzak yönetimde de kullanılır; bkz. Yerel kuyruk yöneticisinden uzak denetim. Dağıtılmış kuyrukta iletim kuyruklarının kullanımıyla ilgili bilgi için bkz. IBM MQ dağıtılmış kuyruklama teknikleri.

Her kuyruk yöneticisinin varsayılan iletim kuyruğu olabilir. Bir kümenin parçası olmayan bir kuyruk yöneticisi, bir iletiyi uzak bir kuyruğa yerleştirirse, varsayılan işlem varsayılan iletim kuyruğunu kullanmaktadır. Hedef kuyruk yöneticisiyle aynı adı taşıyan bir iletim kuyruğu varsa, ileti bu iletim kuyruğuna yerleştirilir. Bir kuyruk yöneticisi diğer adı tanımlaması varsa, **RQMNAME** değiştirgesi hedef kuyruk yöneticiyle eşleşir ve **XMITQ** değiştirgesi belirtilirse, ileti **XMITQ**tarafından adlandırılan iletim kuyruğuna yerleştirilir. Bir **XMITQ** parametresi yoksa, ileti iletide adı belirtilen yerel kuyruğa yerleştirilir.

Küme iletim kuyrukları

Bir kümedeki her kuyruk yöneticisinin, `SYSTEM.CLUSTER.TRANSMIT.QUEUE`adlı bir küme iletim kuyruğu ve bir model kümesi iletim kuyruğu vardır. `SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE`. Bu kuyruklara ilişkin tanımlamalar, bir kuyruk yöneticisi tanımladığınızda varsayılan olarak yaratılır. Kuyruk yöneticisi özniteliği **DEFCLXQ**, `CHANNEL`olarak ayarlandıysa, yaratılan her bir küme gönderici kanalı için otomatik olarak kalıcı bir dinamik küme iletim kuyruğu yaratılır. Kuyruklara `SYSTEM.CLUSTER.TRANSMIT.` *ChannelName*adı verilir. Küme iletim kuyruklarını el ile de tanımlayabilirsiniz.

Kümenin bir parçası olan bir kuyruk yöneticisi, bu kuyruklardan birine, aynı kümedeki diğer kuyruk yöneticilerine ileti gönderir.

Ad çözme sırasında, bir küme iletim kuyruğu, varsayılan iletim kuyruğundan önceliklidir ve belirli bir küme iletim kuyruğu, `SYSTEM.CLUSTER.TRANSMIT.QUEUE`' a göre öncelik kazanır.

Ölü-harfli kuyruklar

Ölü-mektup (teslim edilemeyen ileti) kuyruğu, doğru yerlere yönlendirilemeyen iletilerin saklandığı bir kuyruğdur. Örneğin, hedef kuyruk dolu olduğunda bir ileti yöneltilemez. Verilen ölü-harf kuyruğu `SYSTEM.DEAD.LETTER.QUEUE`olarak adlandırılır.

Dağıtılmış kuyruklama için, ilgili her kuyruk yöneticisininde bir dead-letter kuyruğu tanımlayın.

Komut kuyrukları
The command queue, SYSTEM.ADMIN.COMMAND.QUEUE, is a local queue to which suitably authorized applications can send MQSC commands for processing. Bu komutlar, daha sonra komut sunucusu adı verilen bir IBM MQ bileşeni tarafından alınır. Komut sunucusu, komutların geçerliliğini denetler, kuyruk yöneticisi tarafından işlenmek üzere geçerli olanları geçirir ve uygun yanıtlama kuyruğuna yanıt döndürür.

Kuyruk yöneticisi yaratıldığında, her kuyruk yöneticisi için otomatik olarak bir komut kuyruğu yaratılır.

Yanıtlama-Kuyruklar
Bir uygulama, bir istek iletisi gönderdiğinde, iletiyi alan uygulama, gönderme uygulamasına bir yanıt iletisi gönderebilir. Bu ileti, genellikle gönderme uygulamasına yerel bir kuyruk olan yanıt kuyruğu adı verilen bir kuyruğa gönderilir. Yanıtlama kuyruğu adı, gönderme uygulaması tarafından ileti tanımlayıcısının bir parçası olarak belirtilir.

Olay kuyrukları
İzleme kodu ekleme olayları, kuyruk yöneticilerini MQI uygulamalarından bağımsız olarak izlemek için kullanılabilir.

Bir izleme kodu ekleme olayı ortaya çıktığında, kuyruk yöneticisi bir olay kuyruğuna bir olay iletisi yerleştirir. Bu ileti daha sonra bir izleme uygulaması tarafından okunabilir; bu durum, bir yöneticiyi bilgilendirebilir ya da olay bir sorunu gösterirse, bazı iyileştirici işlemler başlatabilir.

Not: Tetikleme olayları, özel işlemden geçirme olaylarından farklıdır. Tetikleme olayları aynı koşullardan kaynaklanmaz ve olay iletileri oluşturmaz.

Özel işlemden geçirme olaylarıyla ilgili daha fazla bilgi için Instrumentation eventsbaşlıklı konuya bakın.

IBM MQ kuyruk yöneticileri

Uygulamalara sundukları *kuyruk yöneticilerine* ve kuyruğa alma hizmetlerine giriş.

Bir programın kuyruk yöneticisinin hizmetlerini kullanabilmesi için, bir programın kuyruk yöneticiyle bağlantısı olması gerekir. Bir program bu bağlantıyı belirtik olarak (MQCONN ya da MQCONNX çağrısını kullanarak) ya da bağlantıyı örtük olarak (programın çalıştığı platforma ve ortama bağlıdır) yapabilir.

Kuyruk yöneticileri, uygulamalara kuyruğa alma hizmetleri sağlar ve kendilerine ait olan kuyrukları yönetir. Kuyruk yöneticisi aşağıdaki işlemleri sağlar:

- Nesne öznitelikleri alınan komutlara göre değiştirilir.
- Tetikleme olayları ya da özel işlemden geçirme olayları gibi özel olaylar, uygun koşullar karşılandığında oluşturulur.
- Messages are put on the correct queue, as requested by the application making the MQPUT call. Bu işlem yapılamazsa, uygulama bilgilendirilir ve uygun bir neden kodu verilir.

Her kuyruk tek bir kuyruk yöneticisine ait olup, o kuyruk yöneticisine bir *yerel kuyruk* olduğu söylenmektedir. Bir uygulamanın bağlı olduğu kuyruk yöneticisinin o uygulama için *yerel kuyruk yöneticisi* olduğu söylenmektedir. Uygulama için, yerel kuyruk yöneticisine ait olan kuyruklar yerel kuyruklardır.

Uzak kuyruk , başka bir kuyruk yöneticisine ait olan bir kuyruğdur. *uzak kuyruk yöneticisi* , yerel kuyruk yöneticisinden başka bir kuyruk yöneticidir. Uzak bir kuyruk yöneticisi uzak bir makinede ağ üzerinde bulunabilir ya da yerel kuyruk yöneticiyle aynı makinede varolabilir. IBM MQ , aynı makineden birden çok kuyruk yöneticisini destekler.

Bazı MQI çağrılarında bir kuyruk yöneticisi nesnesi kullanılabilir. For example, you can inquire about the attributes of the queue manager object using the MQI call MQINQ.

Kuyruk yöneticilerinin öznitelikleri

Her kuyruk yöneticisiyle ilişkilendirilmiş, özelliklerini tanımlayan bir öznitelikler (ya da özellikler) kümesidir.Bir kuyruk yöneticisinin özniteliklerinden bazıları, yaratıldığında onarılır; IBM MQ komutlarını

kullanarak diğer kişileri değiştirebilirsiniz. MQINQ çağrısını kullanarak, Transport Layer Security (TLS) şifrelemesi için kullanılanlar dışında, tüm özniteliklerin değerlerini sorgulayabilirsiniz.

Sabit öznitelikler şunlardır:

- Kuyruk yöneticisinin adı
- Kuyruk yöneticisinin çalıştırdığı altyapı (örneğin, Windows)
- Kuyruk yöneticisinin desteklediği sistem denetimi komutlarının düzeyi
- Kuyruk yöneticisi tarafından işlenen iletilere atayabileceğiniz öncelik sayısı üst sınırı
- Programların IBM MQ komutlarını gönderebileceği kuyruğun adı
- Kuyruk yöneticisinin işleyebileceği ileti uzunluğu üst sınırı z/OS (yalnızca IBM MQ for z/OS içinde düzeltilebilir)
- Kuyruk yöneticisinin, programlar yerleştirip ileti aldıklarında uyumluluk göstermesini destekleyip desteklemediği

Değiştirilebilir öznitelikler şunlardır:

- Kuyruk yöneticisine ilişkin bir metin tanımlaması
- Kuyruk yöneticisinin, MQI çağrılarını işlediğinde karakter dizgileri için kullandığı karakter kümesi tanıtıcısı
- Kuyruk yöneticisinin tetikleyici ileti sayısını kısıtlamak için kullandığı zaman aralığı
- z/OS Kuyruk yöneticisinin süresi dolan iletiler için hangi sıklıkla kuyrukların taranacağını belirlemek için kullandığı zaman aralığı (yalnızca IBM MQ for z/OS)
- Kuyruk yöneticisinin adı (teslim edilmemiş ileti) kuyruğun adı
- Kuyruk yöneticisinin varsayılan iletim kuyruğunun adı
- Herhangi bir bağlantı için açık tutamaç sayısı üst sınırı
- Olay raporlamasının çeşitli kategorilerinin etkinleştirilmesi ve devre dışı bırakılması
- Bir iş birimi içinde kesinleştirilmemiş ileti sayısı üst sınırı

Kuyruk yöneticileri ve iş yükü yönetimi

Aynı kuyruk için birden çok tanımlaması olan bir kuyruk yöneticisi kümesi ayarlayabilirsiniz (örneğin, kümedeki kuyruk yöneticileri birbirinin klonlarını olabilir). Belirli bir kuyruğa ilişkin iletiler, kuyruğun somut örneğini barındıran herhangi bir kuyruk yöneticisi tarafından işlenebilir. İş yükü yönetimi algoritması, iletiyi hangi kuyruk yöneticisinin işleyeceğine karar verir ve böylece kuyruk yöneticilerinizle arasındaki iş yükünü yayılır; ek bilgi için Küme iş yükü yönetimi algoritmasına bakın.

Süreç tanımlamaları

Süreç tanımlaması nesneleri, uygulamanın, kuyruk yöneticisi tarafından kullanılacak uygulama özniteliklerini tanımlayarak, işletmen müdahalesini gerekmeden başlatılmasına olanak sağlar.

Süreç tanımlaması nesnesi, bir IBM MQ kuyruk yöneticisinde tetikleme olayına yanıt olarak başlayan bir uygulamayı tanımlar. Süreç tanımlaması öznitelikleri, uygulama tanıtıcısını, uygulama tipini ve uygulamaya özgü verileri içerir. Daha fazla bilgi için, “Queues used for specific purposes by IBM MQ” sayfa 22içindeki *Initiation queuer* başlıklı konuya bakın.

Bir uygulamanın işletmen müdahalelerine gerek kalmadan başlatılmasına izin vermek için, Starting IBM MQ applications using triggersiçinde açıklandığı gibi, uygulamanın öznitelikleri kuyruk yöneticisinde bilinmelidir. Bu öznitelikler, bir *süreç tanımlaması nesnesinde*tanımlanır.

Nesne yaratıldığında **ProcessName** özniteliği sabittir. Ancak, IBM MQ komutlarını kullanarak diğer öznitelikleri değiştirebilirsiniz. z/OS Diğer bir seçenek olarak, z/OSişletim sistemi üzerinde IBM MQ for z/OS işlemleri ve denetim panolarını kullanabilirsiniz.

MQINQ-Sorgula nesne özniteliklerinikullanarak, tüm özniteliklerin değerlerini sorgulayabilirsiniz.

İlgili başvurular
Süreç tanımlamalarına ilişkin öznitelikler

Ad listeleri

ad listesi , küme adları, kuyruk adları ya da kimlik doğrulama bilgileri nesne adlarının listesini içeren bir IBM MQ nesnesidir. Bir kümede, kuyruk yöneticisinin havuzları elinde tuttuğu kümelerin listesini tanımlamak için kullanılabilir.

Ad listesi, diğer IBM MQ nesnelerinin listesini içeren bir IBM MQ nesnesidir. Genellikle ad listeleri, tetikleme izleme programları gibi uygulamalar tarafından, bir grup kuyrukları tanımlamak için kullanılırlar. Ad listesi kullanmanın yararı, uygulamalardan bağımsız olarak korunmasıdır; bunu kullanan uygulamaların herhangi birini durdurmadan güncellenebilir. Ayrıca, bir uygulama başarısız olursa, ad listesi etkilenmez ve diğer uygulamalar bu uygulamayı kullanmaya devam edebilir.

Ad listeleri ayrıca, birden çok IBM MQ nesnesi tarafından başvurulan kümelerin bir listesini korumak için kuyruk yöneticisi kümeleriyle de kullanılır.

MQSC komutlarını kullanarak ad listelerini tanımlayabilir ve değiştirebilirsiniz. z/OS Diğer bir seçenek olarak, z/OSüzerinde IBM MQ for z/OS işlemleri ve denetim panolarını kullanabilirsiniz.

Programlar, bu ad listelerinde hangi kuyrukların içerileceğini öğrenmek için MQI ' yi kullanabilir. İsim listelerinin kuruluşu, uygulama tasarımcısının ve sistem yöneticisinin sorumluluğundadır.

İlgili başvurular
Ad listelerine ilişkin öznitelikler
AD LISTESINI TANı

Kimlik doğrulama bilgileri nesneleri

Kimlik doğrulama bilgileri nesnesi, sertifika iptal denetimini gerçekleştirmek için gereken tanımlamaları sağlar.

The queue manager authentication information object forms part of IBM MQ support for Transport Layer Security (TLS). İptal edilen sertifikaların denetlenmesi için gerekli tanımlamalar sağlar. Sertifikasyon yetkileri, artık güvenilmeyecek sertifikaların geri alınmasına neden olur.

Bir kimlik doğrulama bilgileri nesnesi tanımlamak için **DEFINE AUTHINFO** MQSC komutunu kullanabilirsiniz. Kimlik doğrulama bilgileri nesnelerinin öznitelikleriyle ilgili daha fazla bilgi için bkz. **DEFINE AUTHINFO**.

Bir kimlik doğrulama bilgileri nesnesiyle aşağıdaki IBM MQ denetim komutlarını kullanabilirsiniz:

- **setmqaut** (yetki ver ya da yetkiyi iptal et)
- **dspmqaut** (görüntü nesnesi yetkilendirmesi)
- **dmpmqaut** (döküm yetkileri)
- **rcrmqobj** (nesneyi yeniden yarat)
- **rcdmqimg** (kayıt ortamı görüntüsü)
- **dspmqfls** (dosya adlarını görüntüle)

TLS ' ye genel bakış ve kimlik doğrulama bilgileri nesnelerinin kullanımı için bkz. IBM MQiçinde TLS güvenlik iletişim kuralları .

İlgili kavramlar
Transport Layer Security (TLS) kavramları

İletişim bilgileri nesneleri

IBM MQ Multicast, düşük gecikme süresi, yüksek fanlı, güvenilir çoklu yayın ileti sistemi sunar. Çoklu yayın iletimi kullanmak için bir iletişim bilgisi (COMMINFO) nesnesi gereklidir.

Bir COMMINFO nesnesi, çok hedefli iletimle ilişkili öznitelikleri içeren bir IBM MQ nesnesidir. Bu özniteliklere ilişkin ek bilgi için DEFINE COMMINFObaşlıklı konuya bakın. Bir COMMINFO nesnesi yaratma hakkında daha fazla bilgi için bkz. Çok noktaya yayın ile çalışmaya başlama.

İlgili kavramlar

"IBM MQ Çok Yaylı" sayfa 104
IBM MQ Multicast, düşük gecikme süresi, yüksek fan dışında, güvenilir çoklu yayın ileti sistemi sunar.

Kanallar

Kanal , dağıtılmış kuyruk yöneticileri tarafından IBM MQ MQI client ile bir IBM MQ sunucusu arasında ya da iki IBM MQ sunucusu arasında kullanılan, mantıksal bir iletişim bağlantısıdır.

Kanallar, bir kuyruk yöneticisinden diğerine iletişim yolu sağlayan nesnellerdir. Kanallar, iletileri bir kuyruk yöneticisinden diğerine taşımak için kullanılır ve temeldeki iletişim protokollerinden uygulamaları korurlar. Kuyruk yöneticileri aynı sistemde ya da aynı altyapıda ya da farklı platformlarda farklı sistemler üzerinde bulunabilir. Gönderilen iletiler birçok yerden kaynaklanabilir:

- Verileri bir düğümden diğerine aktaran kullanıcı tarafından yazılan uygulama programları.
- PCF komutlarını ya da MQAI olanağını kullanan kullanıcı tarafından yazılmış yönetim uygulamaları.
- IBM MQ Explorer.
- Başka bir kuyruk yöneticisine izleme kodu ekleme olayı iletileri gönderen kuyruk yöneticileri.
- Uzak denetim komutlarını başka bir kuyruk yöneticisine gönderen kuyruk yöneticileri. Örneğin, MQSC komutlarını ya da administrative REST APIkomutunu kullanın.

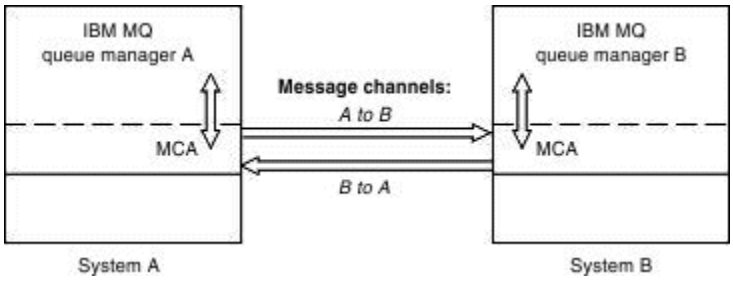
Bir kanalda iki tanım vardır: bağlantının her iki ucunda bir tane vardır. Kuyruk yöneticilerinin birbiriyle iletişim kurmaları için, kuyruk yöneticisinde ileti göndermek için bir kanal nesnesi tanımlamalı ve bunları alacak kuyruk yöneticisinde başka, tamamlayıcı bir kanal nesnesi tanımlamalısınız. Bağlantının her ucunda aynı *kanal adı* kullanılmalı ve kullanılan *kanal tipi* uyumlu olmalıdır.

IBM MQ' ta üç kanal kategorisi vardır; bu kategoriler içinde farklı kanal tipleri vardır:

- Tek yönlü olan ileti kanalları ve bir kuyruk yöneticisinden başka bir kuyruk yöneticisinden ileti aktarır.
- İki yönlü olan MQI kanalları ve bir IBM MQ MQI client ' dan bir kuyruk yöneticisine yönelik MQI çağrıları ve bir kuyruk yöneticisinden IBM MQ istemcisine verilen yanıtlar.
- İki yönlü olan ve bir AMQP istemcisini sunucu makinesinde kuyruk yöneticisine bağlamak için kullanılan MQP kanalları. IBM MQ , AMQP çağrılarını ve yanıtlarını AMQP uygulamaları ile kuyruk yöneticileri arasında aktarmak için AMQP kanallarını kullanır.

İleti kanalları

İleti kanalının amacı, iletilerin bir kuyruk yöneticisinden diğerine aktarılabilmesidir. İstemci sunucusu ortamı ileti kanallarını gerektirmiyor.



Şekil 2. İki kuyruk yöneticisi arasındaki ileti kanalları

İleti kanalı tek yönlü bir bağlantıdır. Uzak kuyruk yöneticisinin yerel bir kuyruk yöneticisi tarafından gönderilen iletilere yanıt vermesini istiyorsanız, yanıtları yerel kuyruk yöneticisine göndermek için ikinci bir kanal ayarlamanız gerekir.

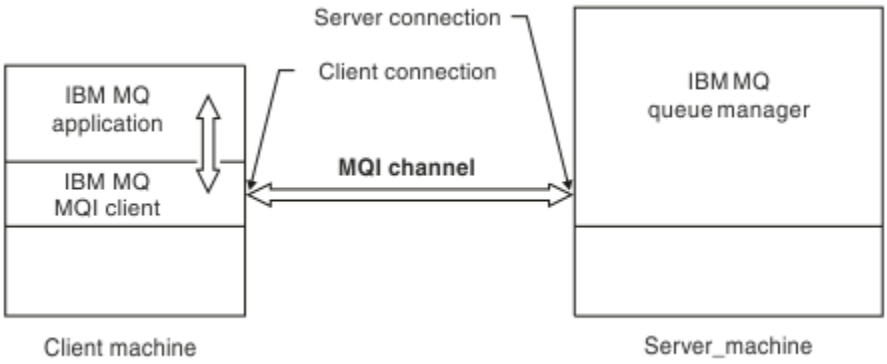
Bir ileti kanalı, iki kuyruk yöneticisini *ileti kanalı aracıları* (MCA ' lar) kullanarak bağlar. Bir kanalın her ucunda bir ileti kanalı aracısı vardır. Bir MCA ' nın birden çok iş parçacığını kullanarak ileti aktarmasına izin verebilirsiniz. Bu işlem *Pipelining*olarak bilinir. Pipelining, MCA ' nın iletileri daha verimli bir şekilde aktarmasını, kanal performansını artırmasını sağlar. Pipelining ile ilgili daha fazla bilgi için Kanalların öznitelikleribaşlıklı konuya bakın.

Kanallar hakkında daha fazla bilgi için bkz. Kanal-çıkış aramaları ve veri yapılarıve “Dağıtılmış kuyruklama bileşenleri” sayfa 44.

MQI kanalları

Message Queue Interface (MQI) kanalı, bir IBM MQ MQI client uygulamasını bir sunucu makinesinde bir kuyruk yöneticisine bağlar ve bir IBM MQ MQI client uygulamasından MQCONN ya da MQCONNX çağrısı yayınlarken kurulur.

Bu, iki yönlü bir bağlantıdır ve yalnızca, ileti verilerini içeren MQPUT çağrıları ve ileti verilerinin döndürülmesi sonucu MQGET çağrıları da dahil olmak üzere, MQI çağrılarının ve yanıtların aktarılması için kullanılır. Kanal tanımlamalarının yaratılıp kullanılmasının farklı yolları vardır (bkz. MQI kanallarının tanımlanması).



Şekil 3. Bir MQI kanalına istemci bağlantısı ve sunucu bağlantısı

z/OS Bir MQI kanalı, bir istemciyi tek bir kuyruk yöneticisine ya da kuyruk paylaşım grubunun bir parçası olan bir kuyruk yöneticisine bağlamak için kullanılabilir (bkz. İstemcinin kuyruk paylaşım grubuna bağlanması).

MQI kanalı tanımlamaları için iki kanal tipi vardır. İki yönlü MQI kanalını tanımlarlar.

İstemci bağlantı kanalı
Bu tip, IBM MQ MQI clientiçindir.

Sunucu bağlantısı kanalı
Bu tip, kuyruk yöneticisini çalıştıran sunucu içindir; IBM MQ MQI client ortamında çalışan IBM MQ uygulaması iletişim kuralıdır.

AMQP kanalları

Multi

Tek bir AMQP kanalı tipi vardır.

Bir AMQP ileti sistemi uygulamasını bir kuyruk yöneticisiyle bağlamak için, uygulamanın IBM MQ uygulamalarıyla ileti değiştirmesini sağlayan bir kanalı kullanıyorsunuz. Bir AMQP kanalı, MQ Lightkullanarak bir uygulama geliştirmenizi ve daha sonra, IBM MQtarafından sağlanan kurumsal düzeydeki olanaklardan yararlanarak bunu bir kurumsal uygulama olarak devreye almanıza olanak sağlar.

Kanal tanımları

Her kanal tipine ilişkin açıklamalar için bkz. “Kanal tanımları” sayfa 28 .

İlgili kavramlar

“Dağıtılmış kuyruğa alma ve kümeler” sayfa 40
Dağıtılmış kuyruklama, bir kuyruk yöneticisinden başka bir kuyruk yöneticisinden ileti gönderme anlamına gelir. Alma kuyruk yöneticisi aynı makinede ya da başka bir yerde olabilir; yakınlarda ya da dünyanın diğer tarafında olabilir. Yerel kuyruk yöneticiyle aynı platformda çalışıyor olabilir ya da IBM MQtarafından desteklenen platformların herhangi birinde olabilir. You can manually define all the connections in a distributed queuing environment, or you can create a cluster and let IBM MQ define much of the connection detail for you.

İleti Kuyruğu Arabirimi-Genel Bakış

İlgili görevler

Uzak IBM MQ nesnelerini yönetme

MQI kanalları durduruluyor

İlgili başvurular

Kanal-çıkış çağrıları ve veri yapıları

“İletişim” sayfa 31
IBM MQ MQI clients , sunucuyla iletişim kurmak için MQI kanallarını kullanın.

Kanal tanımları

IBM MQ ' un kullandığı farklı tipteki ileti kanallarını ve MQI kanallarını tanımlayan tablolar.

İleti kanallarından söz edilirken, sözcük kanalı genellikle bir kanal tanımlamasının eşanlamlısı olarak kullanılır. Genellikle tek bir ucu olan, iki ucu ya da kanal tanımlaması olan, tam bir kanaldan mı bahsediyoruz, bu bağlamda durum nettir.

İleti kanalları

İleti kanalı tanımlamaları aşağıdaki tiplerden biri olabilir:

İleti kanalı tanımlaması tipi	Tanım
Gönderen	Gönderen kanalı, kuyruk yöneticisinin diğer kuyruk yöneticilerine ileti göndermek için kullandığı bir ileti kanalsıdır. Gönderici kanalı kullanarak ileti göndermek için, diğer kuyruk yöneticisiyle de, gönderen kanalıyla aynı adı taşıyan bir alıcı kanalı yaratmalısınız. Bir "geri çağırma" mekanizması uyguluyorsanız, gönderen kanallarını istek kanallarıyla da kullanabilirsiniz.
Sunucu	Sunucu kanalı, kuyruk yöneticisinin diğer kuyruk yöneticilerine ileti göndermek için kullandığı bir ileti kanalsıdır. Bir sunucu kanalını kullanarak ileti göndermek için, diğer kuyruk yöneticisininde de, sunucu kanalıyla aynı adı taşıyan bir alıcı kanalı yaratmalısınız. Ayrıca, sunucu kanallarını istekçi kanallarıyla da kullanabilirsiniz. Bu durumda, kanalın diğer ucundaki istekçi kanalı tanımlaması, sunucu kanalı tanımlamasını başlatmaya devam eder. Sunucu istekçiye ileti gönderir. Sunucu, iş ortağı kanalının bağlantı adını bildiği sürece iletişimi de başlatabiliyor.

İleti kanalı tanımlaması tipi	Tanım
Alıcı	Alıcı kanalı, kuyruk yöneticisinin diğer kuyruk yöneticilerinden gelen iletileri almak için kullandığı bir ileti kanalsıdır. Bir alıcı kanalını kullanarak ileti almak için, diğer kuyruk yöneticisiyle, bir gönderici ya da bu alıcı kanalıyla aynı adı taşıyan bir sunucu kanalı yaratmalısınız.
İsteyen	İstekte Bulunan kanalı, kuyruk yöneticisinin diğer kuyruk yöneticilerinden gelen iletileri almak için kullandığı bir ileti kanalsıdır. Bir Requester kanalı, uzak uçta tanımlanan iş ortağı kanalının başlamasını isteyebilir. Ortak kanal bir Sunucu kanalıysa, Sunucu kanalı başlatma isteğini kabul eder ve Sunucu kanalı tanımında belirlenen iletim kuyruğundan Requester kanalına ileti göndermeye başlar. İş ortağı kanalı bir Gönderen kanalıysa, Gönderen kanalı başlatma isteğini kabul eder, ancak daha sonra İstek İstekçisi ile bağlantıyı kapatır. Sonra Gönderen kanalı başlatılır, iş ortağı İstekçisi kanalıyla bir oturum kararlaştırılır ve Gönderen kanalı tanımında belirlenen iletim kuyruğundan ileti göndermeye başlar. Bu ikinci durum esasen, Requester kanalının, Gönderen kanalını geri aramasını istemesi durumunda bir geri arama mekanizması sağlar.
Küme-gönderen	Kümeli gönderen (CLUSSDR) kanal tanımlaması, bir küme kuyruk yöneticisinin küme bilgilerini tam havuzlardan birine gönderebileceği bir kanalın gönderileceği bir kanal sonunu tanımlar. Küme gönderici kanalı, kuyruk yöneticisinin durumuna ilişkin değişiklikleri (örneğin, kuyruğun eklenmesi ya da kaldırılması gibi) bilgilendirmek için kullanılır. İletileri iletmek için de kullanılır. Tam havuz kuyruğu yöneticilerinin, birbirlerine işaret eden küme gönderici kanalları vardır. Bunlar, küme durumu değişikliklerini birbiriyle iletişim kurmak için kullanırlar. Bir kuyruk yöneticisinin CLUSSDR kanal tanımlama noktasının gösterdiği tam havuz önem düzeyi küçük önem derecesidir. İlk iletişim sorumlusu yapıldıktan sonra, kuyruk yöneticisinin her havuza küme bilgilerini ve her kuyruk yöneticisine ileti gönderebilmesi için, daha fazla küme kuyruk yöneticisi nesneleri otomatik olarak tanımlanmıştır.
Küme-alıcı	Bir küme-alıcı (CLUSRCVR) kanal tanımlaması, küme kuyruk yöneticisinin kümedeki diğer kuyruk yöneticilerinden ileti alabileceği bir kanalın giriş sonunu tanımlar. Bir küme-alıcı kanalı, havuza ilişkin kümeyle ilgili bilgileri de taşıyabilir. Kuyruk alıcı kanalı tanımlanarak, kuyruk yöneticisi diğer küme kuyruğu yöneticilerine ileti almak için uygun olduğunu gösterir. Her küme kuyruk yöneticisi için en az bir küme alıcı kanalına ihtiyacınız vardır.

Her kanal için, her iki ucu da tanımlamanız gerekir. Böylece, kanalın her bir ucu için bir kanal tanımınız olur. Kanalın iki ucu uyumlu tipte olmalıdır.

Kanal tanımlarının aşağıdaki birleşimlerini elde edebilirsiniz:

- Gönderen-Alıcı
- Sunucu-Receiver
- İstekte Bulunanın-Sunucu
- İsteyenin-Gönderen (geri çağrı)
- Küme-gönderen-Küme-alıcı

İleti kanalı aracıları

Yarattığınız her kanal tanımlaması, belirli bir kuyruk yöneticisine ait olur. Bir kuyruk yöneticisi, aynı ya da farklı tipte birkaç kanal olabilir. Kanalın her bir ucunda bir program olan Message Channel Agent (MCA) bulunur. Kanalın bir ucunda, çağıran MCA ileti iletim kuyruğundan ileti alır ve bunları kanaldan gönderir. Kanalın diğer ucunda ise yanıt veren MCA, iletileri alır ve uzak kuyruk yöneticisine teslim eder.

Çağıran MCA, bir gönderen, sunucu ya da istekçi kanalıyla ilişkilendirilebilir. Yanıt veren MCA, herhangi bir ileti kanalı tipiyle ilişkilendirilebilir.

IBM MQ , bir bağlantının iki ucunda aşağıdaki kanal tipi birleşimlerini destekler:

Çağıran		**İleti akışının yönü**	**Yanıt Veren**	
Kanal tipi	**Dinleyici gerekli mi?**		**Dinleyici gerekli mi?**	**Kanal tipi**
Gönderen	Hayır	Arayan Kişiye Yanıt Veren Kişi	Evet	Alıcı
Sunucu	Hayır	Arayan Kişiye Yanıt Veren Kişi	Evet	Alıcı
Sunucu	Hayır	Arayan Kişiye Yanıt Veren Kişi	Evet	İsteyen
İsteyen	Hayır	Caller 'a yanıt veren kişi	Evet	Sunucu
İsteyen	Evet	Caller 'a yanıt veren kişi	Evet	Gönderen

MQI kanalları

MQI kanalları aşağıdaki tiplerden biri olabilir:

MQI kanalı tipi	**Tanım**
Sunucu bağlantısı	Sunucu bağlantısı kanalı, bir IBM MQ istemcisini IBM MQ sunucusuna bağlamak için kullanılan iki yönlü bir MQI kanalı. Sunucu bağlantı kanalı, kanalın sunucu sondur.
İstemci bağlantısı	A client connection channel is a bidirectional MQI channel that is used to connect an IBM MQ client to an IBM MQ server. IBM MQ Explorer , uzak kuyruk yöneticilerine bağlanmak için istemci bağlantılarını da kullanır. İstemci bağlantı kanalı, kanalın istemci bitişidir. Bir istemci-bağlantı kanalı yarattığınızda, kuyruk yöneticisini barındıran bilgisayarda bir dosya yaratılır. İstemci-bağlantı kütüğünü IBM MQ istemci bilgisayarına kopyalamanız gerekir.

İletişim

IBM MQ MQI clients , sunucuyla iletişim kurmak için MQI kanallarını kullanın.

Bağlantının hem IBM MQ MQI client hem de sunucu uçlarında bir kanal tanımlaması yaratılmalıdır. Kanal tanımlamalarının nasıl yaratılacağı, MQI kanallarının tanımlanmasıbaşlıklı konu ile açıklanmıştır.

Olası iletim protokolleri aşağıdaki çizelgede gösterilmiştir:

Çizelge 1. MQI kanallarına ilişkin iletim protokolleri

İstemci altyapısı	LU 6.2	TCP/IP	NetBIOS	SPX
IBM i IBM i		Evet		
Linux AIX AIX and Linux sistemleri	Evet [1]	Evet		
Windows Windows	Evet	Evet	Evet	Evet

Not:

1. Linux LU6.2 aşağıdaki altyapılarda desteklenmez:
   - Linux (POWER platformu)
   - Linux (x86-64 altyapısı)
   - Linux (zSeries s390x platformu)

İletim protokolleri- IBM MQ MQI client ve sunucu altyapılarının birleşimi shows the possible combinations of IBM MQ MQI client and server platforms, using these transmission protocols.

IBM MQ MQI client üzerindeki bir IBM MQ uygulaması, kuyruk yöneticisinin yerel olduğu gibi, tüm MQI çağrılarını aynı şekilde kullanabilir. **MQCONN** ya da **MQCONNX** , IBM MQ uygulamasını seçilen kuyruk yöneticisiyle ilişkilendirir ve bir *bağlantı tanıtıcısı*yaratır. Bu bağlantı tanıtıcısını kullanan diğer aramalar, bağlı kuyruk yöneticisi tarafından işlenir. IBM MQ MQI client iletişimi, istemci ile sunucu arasında, bağlantı bağımsız ve zamana bağımsız olan, kuyruk yöneticileri arasındaki iletişimin karşıtlığı arasında etkin bir bağlantı gerektirir.

İletim protokolü, kanal tanımlaması kullanılarak belirtilir ve uygulamayı etkilemez. Örneğin, bir Windows uygulaması, TCP/IP üzerinden bir kuyruk yöneticisine ve NetBIOSüzerinden başka bir kuyruk yöneticiye bağlanabilir.

Performansa dikkat

Kullandığınız iletim protokolü, IBM MQ istemci ve sunucu sisteminin başarımını olumsuz etkileyebilir. Yavaş bir telefon hattı üzerinden çevirmeli destek için, IBM MQ kanal sıkıştırma özelliğini kullanmanız önerilir.

İstemci bağlantı kanalları

İstemci bağlantı kanalları , IBM MQ MQI client ' dan kuyruk yöneticisine bir iletişim yolu sağlayan nesnellerdir.

İstemci bağlantı kanalları, bir kuyruk yöneticisi ile bir istemci arasında ileti taşımak için dağıtımlı kuyruğa alma sırasında kullanılır. Temeldeki iletişim protokollerinden uygulamaları korurlar. İstemci, kuyruk yöneticisinden aynı ya da farklı bir altyapıda var olabilir.

İlgili görevler

Sunucu ve istemci arasındaki bağlantıların yapılandırılması

z/OS Depolama sınıfları

Bir depolama sınıfı, bir ya da daha çok kuyrukları bir sayfa kümesine eşler.

Bu, söz konusu kuyruğa ilişkin iletilerin o sayfa kümesinde saklanma (arabelleğe alma konusu) olduğu anlamına gelir.

Depolama sınıfları yalnızca IBM MQ for z/OSüzerinde desteklenir.

Depolama sınıflarıyla ilgili daha fazla bilgi için Planlama on z/OSbaşlıklı konuya bakın.

Dinleyiciler

Dinleyiciler , diğer kuyruk yöneticilerinden ya da istemci uygulamalarından gelen ağ isteklerini kabul eden ve ilişkili kanalların başlatıldığını kabul eden işlemlerdir.

Dinleyici işlemleri , **runmqlsr** denetim komutu kullanılarak başlatılabilir.

Dinleyici nesneleri , dinleyici işlemlerinin başlangıç ve durdurulmasını kuyruk yöneticisi kapsamı içinden yönetmenize olanak sağlayan IBM MQ nesnelidir. Bir dinleyici nesnesine ilişkin öznitelikleri tanımlayarak aşağıdakileri yapın:

- Dinleyici işlemini yapılandırın.
- Dinleyici işleminin kuyruk yöneticisi başladığında ve durduğunda otomatik olarak başlatılıp başlatılmayacağını ve duramayacağını belirtir.

Önemli: z/OS Dinleyici nesneleri IBM MQ for z/OSüzerinde desteklenmez. For more information about how IBM MQ for z/OS implements listening, by using the channel initiator, see “z/OSüzerindeki kanal başlatıcısı” sayfa 156.

İlgili başvurular

runmqlsr (çalıştırma dinleyicisi)

Hizmetler

Hizmet nesneleri, bir kuyruk yöneticisi başlatıldığında ya da durduğunda çalıştırılacak programları tanımlama biçimidir.

Programlar aşağıdaki tiplerden biri olabilir:

Sunucular
Sunucu , SERVTYPE parametresine sahip sunucu olarak belirtilen bir hizmet nesnesidir. Sunucu hizmeti nesnesi, belirlenen bir kuyruk yöneticisi başlatıldığında yürütülecek olan bir programın tanımlamasıdır. Bir sunucu işleminin tek bir eşgörünümü koşutzamanlı olarak yürütülebiliyor. Çalışma sırasında, bir sunucu işleminin durumu, MQSC komutu kullanılarak izlenebilir, DISPLAY SVSTATUS. Genellikle sunucu hizmeti nesneleri, ölü harf işleyicileri ya da tetikleme izleme programları gibi programların tanımlarıdır, ancak çalıştırılabilecek programlar IBM MQile birlikte sağlanan programlarla sınırlı değildir. Ayrıca, bir sunucu hizmeti nesnesi, belirtilen kuyruk yöneticisi programı sona erdirmek için sona erdirildiğinde çalıştırılacak bir komut da içerecek şekilde tanımlanabilir.

Komutlar
Komut , COMMAND olarak belirtilen SERVTYPE parametresine sahip bir hizmet nesnesidir. Bir komut hizmeti nesnesi, belirtilen bir kuyruk yöneticisi başlatıldığında ya da durdurulduğunda yürütülecek olan bir programın tanımlamasıdır. Bir komut işleminin birden çok eşgörünümü koşutzamanlı olarak yürütülebiliyor. Komut hizmeti nesneleri, sunucu yürütüldükten sonra, kuyruk yöneticisi programı izlemeyince, sunucu hizmet nesnelerinden farklı olur. Genellikle komut hizmeti nesneleri, kısa ömürlü olan programların tanımlarıdır ve bir ya da daha çok görev başlatma gibi belirli bir görevi gerçekleştirir.

Önemli: z/OS Hizmet nesneleri IBM MQ for z/OSüzerinde desteklenmez.

İlgili kavramlar

Hizmetlerle çalışma

Konu nesneleri

Konu nesnesi , konulara belirli, varsayılan olmayan öznitelikler atamanıza olanak sağlayan bir IBM MQ nesnesidir.

konu , belirli bir *konu dizesi*' ye abone olan bir uygulama yayınlama ya da abone olma yoluyla tanımlanır. Bir konu dizgisi, ileriye eğik çizgi karakteri (/) ile ayırarak bir konu sıradüzeni belirtebilir. Bu, bir *konu ağacı*tarafından görselleştirilebilir. Örneğin, bir uygulama `/Sport/American Football` ve `/Sport/Soccer`konu dizgilerinde yayınlanırsa, iki alt düğümü (`American Football`, ve `Soccer`) `Sport` üst düğümüne sahip bir konu ağacı yaratılır.

Konular, özniteliklerini, konu ağacında bulunan ilk üst denetim düğümünden devralır. Belirli bir konu ağacında herhangi bir denetim konusu düğümü yoksa, tüm konular özniteliklerini temel konu nesnesinden (SYSTEM.BASE.TOPIC.

Konu nesnesinin TOPICSTR özniteisinde o düğümün konu dizgisini belirterek, bir konu ağacındaki herhangi bir düğümde bir konu nesnesi yaratabilirsiniz. Denetim konusu düğümü için başka öznitelikler de tanımlayabilirsiniz. Bu özniteliklerle ilgili daha fazla bilgi için bakınız: The MQSC commansya da Automating admins using PCF commans. Her konu nesnesi varsayılan olarak, özniteliklerini en yakın üst denetim konusu düğümünden edinir.

Konu nesneleri, uygulama geliştiricilerinden tam konu ağacını gizlemek için de kullanılabilir. If a topic object named `FOOTBALL.US` is created for the topic `/Sport/American Football`, an application can publish or subscribe to the object named `FOOTBALL.US` instead of the string `/Sport/American Football` with the same result.

Bir konu nesnesindeki bir konu dizgisinde bir #, +,/ya da * karakteri girerseniz, karakter dizgi içinde normal bir karakter olarak işlenir ve bir konu nesnesiyle ilişkili konu dizgisinin bir parçası olarak kabul edilir.

Konu nesnelerine ilişkin daha fazla bilgi için bkz. “Yayınlama/abone olma ileti alışverişi” sayfa 59.

IBM MQ nesnelerini adlandırma

IBM MQ nesneleri için kabul edilen adlandırma kuralı, nesneye bağlıdır. The name of the machines and the user IDs that you use with IBM MQ are also subject to some naming restrictions.

Bir kuyruk yöneticisinin her yönetim ortamı adı ile bilinir. Bu ad, birbiriyle bağlantılı kuyruk yöneticileri ağı içinde benzersiz olmalıdır; böylece, bir kuyruk yöneticisi, herhangi bir iletinin gönderildiği hedef kuyruk yöneticisini belirsiz bir şekilde tanımlayabilir.

Diğer nesne tipleri için, her nesnenin kendisiyle ilişkilendirilmiş bir adı vardır ve bu ad bu adla anılabilir. Bu adlar, bir kuyruk yöneticisi ve nesne tipi içinde benzersiz olmalıdır. Örneğin, aynı adı taşıyan bir kuyruğunuz ve bir işleminiz olabilir, ancak aynı adı taşıyan iki kuyruğun olamaz.

IBM MQiçinde, en çok 20 karakterden oluşan *kanal* özel durumu ile en çok 48 karakter olabilir. IBM MQ nesnelerine adlandırma hakkında daha fazla bilgi için bkz. “IBM MQ nesnelerinin adlandırılmasına ilişkin kurallar” sayfa 34.

The name of the machines and the user IDs that you use with IBM MQ are also subject to some naming restrictions:

- Makine adının boşluk içermediğinden emin olun. IBM MQ , boşluk içeren makine adlarını desteklemez. IBM MQ ' u böyle bir makineye kursanız, kuyruk yöneticisi yaratamazsınız.
- IBM MQ yetkileri için, kullanıcı kimliklerinin ve grupların adlarının 20 karakterden uzun olmaması gerekir (boşluk kullanılamaz).
- Windows İstemci, @ karakterini içeren bir kullanıcı kimliği altında çalışıyorsa, IBM MQ for Windows sunucusu, IBM MQ MQI client bağlantısını desteklemez; örneğin, abc@d.

İlgili kavramlar

“IBM MQ dosya adları” sayfa 37
Her IBM MQ kuyruk yöneticisi, kuyruğu, süreç tanımlaması, ad listesi, kanal, istemci bağlantısı kanalı, dinleyici, hizmet ve kimlik doğrulama bilgileri nesnesi bir dosya tarafından temsil edilir. Nesne adlarının

geçerli dosya adları olması gerektiğinden, kuyruk yöneticisi nesne adını gerektiği yerde geçerli bir dosya adına dönüştürür.

İlgili başvurular

"IBM MQ nesnelerinin adlandırılmasına ilişkin kurallar" sayfa 34

IBM MQ nesne adlarında uzunluk üst sınırı vardır ve büyük/küçük harf duyarlıdır. Her nesne tipi için tüm karakterler desteklenmez ve adların benzersizliğine ilişkin birçok nesne kural içerir.

IBM MQ nesnelerinin adlandırılmasına ilişkin kurallar

IBM MQ nesne adlarında uzunluk üst sınırı vardır ve büyük/küçük harf duyarlıdır. Her nesne tipi için tüm karakterler desteklenmez ve adların benzersizliğine ilişkin birçok nesne kural içerir.

Birçok farklı IBM MQ nesnesi tipi vardır ve her tipteki nesneler ayrı nesne ad alanlarında varolduğu için aynı ada sahip olabilir: Örneğin, yerel bir kuyruk ve bir gönderen kanalı aynı ada sahip olabilir. Ancak, bir nesne aynı ad alanındaki başka bir nesnenle aynı adı olamaz: Örneğin, yerel bir kuyruğun adı bir model kuyruğundan aynı olamaz ve bir gönderen kanalı bir alıcı kanalıyla aynı ada sahip olamaz.

Aşağıdaki IBM MQ nesneleri ayrı nesne ad alanlarında bulunur:

- Kimlik doğrulama bilgileri
- Kanal
- İstemci kanalı
- Dinleyici
- Ad Listesi
- Süreç
- Kuyruk
- Hizmet
- Depolama sınıfı
- Abonelik
- Konu

Nesne adlarına ilişkin karakter uzunluğu

Genel olarak, IBM MQ nesne adları en çok 48 karakter uzunluğunda olabilir. Bu kural aşağıdaki nesneler için geçerlidir:

- Kimlik doğrulama bilgileri
- Küme
- Dinleyici
- Ad Listesi
- Süreç tanımlaması
- Kuyruk
- Kuyruk yöneticisi
- Hizmet
- Abonelik
- Konu

Kısıtlamalar vardır:

1. z/OS z/OS sistemlerinde, kuyruk yöneticileri en çok 4 karakter olmalıdır ve yalnızca büyük harfli karakterler ve sayısal karakterler içmelidir.
2. Kanal nesnesi adlarının ve istemci bağlantı kanalı adlarının uzunluk üst sınırı 20 karakterdir. Kanallarla ilgili ek bilgi için Kanalların tanımlanması başlıklı konuya bakın.

3. Konu dizgileri en fazla 10240 byte olabilir. Tüm IBM MQ nesne adları büyük/küçük harfe duyarlıdır.
4. Abonelik adları en fazla 10240 byte olabilir ve boşluk içerebilir.
5. Depolama sınıfı adlarının uzunluk üst sınırı 8 karakterdir.
6. CF yapısı adlarının uzunluk üst sınırı 12 karakterdir.

Nesne adlarındaki karakterler

IBM MQ nesne adları için geçerli karakterler şunlardır:

Karakterler	Kısıtlamalar
Büyük A-Z	• Yok
Küçük harf a-z	• MQSC komut dosyalarında, küçük harfli karakterler tek tırnak işareti içine alınmalıdır. Bu, küçük harflerin büyük harfle bükülmesini önler. • EBCDIC Katakana kullanan sistemler, nesne adlarında küçük harf a-z karakterleri kullanamaz. • z/OS z/OS sistemlerinde küçük harfli karakterler kullanılırken bazı kısıtlamalar olabilir; örneğin, kuyruk yöneticisi adları küçük harfli karakterler içeremez. • IBM i CL komutlarını kullanırken IBM i sistemlerinde, küçük harf içeren adlar tek tırnak işareti içine alınmalıdır. Bu, küçük harflerin büyük harfle bükülmesini önler.
Sayısal 0-9	• Yok
Nokta (.)	• Yok
Alt çizgi (_)	• Multi Yok • z/OS Avoid using names with leading or trailing underscores because they cannot be handled by the IBM MQ for z/OS operations and control panels.
Eğik Çizgi (/)	• Windows Windows sistemlerinde, kuyruk yöneticisi adının ilk karakteri eğik çizgi olamaz. • IBM i CL komutlarını kullanırken IBM i sistemlerinde, eğik çizgi içeren adların tek tırnak işareti içine alınması gerekir. • z/OS Yok

Karakterler	Kısıtlamalar
Yüzde işareti (%)	• ALW Yok • z/OS If you are using RACF as the external security manager for IBM MQ for z/OS, do not use % in object names because the names are not included in security checks when RACF generic profiles are used. • IBM i CL komutlarını kullanırken IBM i sistemlerinde, yüzde imi içeren adların tek tırnak işareti içine alınması gerekir.

Ayrıca, nesne adlarındaki karakterlerle ilgili bazı genel kurallar da vardır:

1. Baştaki ya da gömülü boşluklara izin verilmez.
2. Ulusal dil karakterlerine izin verilmez.
3. Alan uzunluğundan az olan herhangi bir ad, boşlukla sağa doğru doldurulabilir. Kuyruk yöneticisi tarafından döndürülen tüm kısa adlar, her zaman boşluklarla sağa yaslanır.

Kuyruk adları

Bir kuyruğun adı iki bölümden oluşan bir addır:

- Kuyruk yöneticisinin adı
- Kuyruk yöneticisi tarafından bilindiği için, kuyruğun yerel adı

Kuyruk adının her bölümü 48 karakter uzunluğunda olabilir.

Yerel bir kuyruğa gönderme yapmak için, kuyruk yöneticisinin adını (boş karakterlerle değiştirerek ya da baştaki boş değerli bir karakteri kullanarak) atlayabilirsiniz. Ancak, IBM MQ tarafından bir programa döndürülen tüm kuyruk adları, kuyruk yöneticisinin adını içerir.

z/OS Kuyruk paylaşım grubundaki herhangi bir kuyruk yöneticisi tarafından erişilebilen, paylaşılan bir kuyruk, aynı kuyruk paylaşım grubundaki paylaşılmayan yerel kuyruklarla aynı adı sahip olamaz. Bu kısıtlama, bir uygulamanın yerel bir kuyruk açmayı amaçladığında, bir uygulamanın paylaşılan bir kuyruğu yanlışlıkla açmasını önler ya da tam tersi şekilde bir uygulama kuyruğu açar. Paylaşılan kuyruklar ve kuyruk paylaşım grupları yalnızca IBM MQ for z/OSüzerinde kullanılabilir.

Uzak bir kuyruğa gönderme yapmak için, bir programın kuyruk yöneticisinin adını tam kuyruk adına eklemesi ya da uzak kuyruğun yerel tanımlaması olması gerekir.

Bir uygulama kuyruk adı kullandığında, bu ad yerel bir kuyruğun adı (ya da bir diğeradı) ya da uzak bir kuyruğun yerel tanımlamasının adı olabilir; ancak, kuyruktan ileti almak için gerekmedikçe (kuyruk yerel olması gerektiğinde) uygulamanın bilmesi gerekmez. Uygulama kuyruk nesnesini açtığında, MQOPEN çağrısı, sonraki işlemleri gerçekleştirmek üzere hangi kuyruğun gerçekleştirileceğini belirlemek için bir ad çözme işlevi gerçekleştirir. Bunun önemi, uygulamanın, belirli kuyruklar üzerinde, bir kuyruk yöneticisi ağındaki belirli konumlarda tanımlanmakta olan yerleşik bir bağımlılığının olmamasıdır. Bu nedenle, bir sistem denetimcisi ağdaki kuyrukları yeniden bulursa ve tanımlarını değiştirirse, bu kuyrukları kullanan uygulamaların değiştirilmesi gerekmez.

Ayrılmış nesne adları

SYSTEM. ile başlayan nesne adları, kuyruk yöneticisi tarafından tanımlanan nesneler için ayrılmıştır. Bu nesne tanımlamalarını kuruluşunuza uyacak şekilde değiştirmek için **Alter**, **Define**ve **Replace** komutlarını kullanabilirsiniz. IBM MQ için tanımlanan adlar, Kuyruk adlarıiçinde tam olarak listelenir.

z/OS IBM MQ for z/OS' ta, CSQSYSAPPL bağlantı olanağı uygulama yapısı adı ayrılmış olur.

İlgili kavramlar

AIX, Linux, and Windowsüzerindeki kuruluş adı

IBM MQ dosya adları

Her IBM MQ kuyruk yöneticisi, kuyruğu, süreç tanımlaması, ad listesi, kanal, istemci bağlantısı kanalı, dinleyici, hizmet ve kimlik doğrulama bilgileri nesnesi bir dosya tarafından temsil edilir. Nesne adlarının geçerli dosya adları olması gerektiğinden, kuyruk yöneticisi nesne adını gerektiği yerde geçerli bir dosya adına dönüştürür.

Kuyruk yöneticisi dizini için varsayılan yol aşağıdaki gibidir:

- IBM MQ yapılandırma bilgilerinde tanımlanan bir önek:
  - Linux AIX AIX and Linux'ta varsayılan önek `/var/mqm`' dir. Bu, mqs.ini yapılandırma dosyasının `DefaultPrefix` stanza içinde yapılandırılır.
  - Windows Windows 32 bit sistemlerinde varsayılan önek `C:\Program Files (x86) \IBM\WebSphere MQ`olur. Windows 64 bit sistemlerde varsayılan önek `C:\Program Files\IBM\MQ`' dir. Hem 32 bit, hem de 64 bit kuruluşlar için, veri dizinleri `C:\ProgramData\IBM\MQ`' e kurulur. Bu, mqs.ini yapılandırma dosyasının `DefaultPrefix` stanza içinde yapılandırılır.

  Kullanılabilir olduğunda, önek IBM MQ Explorer 'da IBM MQ özellikler sayfası kullanılarak değiştirilebilir, tersi durumda `mqs.ini` yapılandırma dosyasını el ile düzenleyin.
- Kuyruk yöneticisi adı geçerli bir dizin adı olarak dönüştürüldü. Örneğin, kuyruk yöneticisi:

  ```
  queue.manager
  ```

  şu şekilde temsil edilir:

  ```
  queue!manager
  ```

  Bu işleme *ad dönüşümü*adı verilir.

IBM MQ' ta, bir kuyruk yöneticisinde en çok 48 karakterden oluşan bir ad verebilirsiniz.

Örneğin, bir kuyruk yöneticisi adını verebilirdiniz:

```
QUEUE.MANAGER.ACCOUNTING.SERVICES
```

Ancak, her kuyruk yöneticisi bir dosya tarafından temsil edilir ve bir dosya adının uzunluk üst sınırında ve adda kullanılabilen karakterlerle ilgili sınırlamalar vardır. Sonuç olarak, nesneleri temsil eden dosyaların adları dosya sisteminin gereklerini karşılamak için otomatik olarak dönüştürülür.

Kuyruk yöneticisi adının dönüşümünü yöneten kurallar şunlardır:

1. Tek tek karakterleri dönüştür:
   - -Evet. !
   - Başlangıç/to &
2. Ad hala geçerli değilse:
   a. Sekiz karaktere kadar kes
   b. Üç karakterlik bir sayısal sonek ekler

For example, assuming the default prefix and a queue manager with the name `queue.manager`:

- Windows Windows üzerinde NTFS ya da FAT32ile birlikte kuyruk yöneticisi adı şöyle olur:

```
C:\Program Files\IBM\MQ\qmgrs\queue!manager
```

- Windows On Windows with FAT, the queue manager name becomes:

```
C:\Program Files\IBM\MQ\qmgrs\queue!ma
```

- Linux AIX AIX and Linux' ta kuyruk yöneticisi adı şöyle olur:

```
/var/mqm/qmgrs/queue!manager
```

Dönüştürme algoritması, yalnızca büyük ve küçük harfe duyarlı olmayan dosya sistemlerine göre farklılık gösteren adlar arasında da ayırt edilir.

Nesne adı dönüşümü

Nesne adlarının geçerli dosya sistemi adları olması gerekmez. Nesne adlarınızı dönüştürmeniz gerekebilir. Kullanılan yöntem, kuyruk yöneticisi adlarından farklıdır; her makinede yalnızca birkaç kuyruk yöneticisi adı olmasına rağmen, her kuyruk yöneticisi için çok sayıda başka nesne olabilir. Dosya sisteminde kuyruklar, süreç tanımlamaları, ad listeleri, kanallar, istemci bağlantı kanalları, dinleyiciler, hizmetler ve kimlik doğrulama bilgileri nesneleri gösterilir.

Dönüştürme işlemi tarafından yeni bir ad oluşturulduğunda, özgün nesne adıyla basit bir ilişki yoktur. Gerçek ve dönüştürülen nesne adları arasında dönüştürme yapmak için **dspmqfls** komutunu kullanabilirsiniz.

İlgili başvurular
dspmqfls (dosya adlarını görüntüle)

IBM i IBM iüzerindeki nesne adları

Kuyruk yöneticisinin, benzersiz bir adı olan bir kuyruk yöneticisi kitaplığı var. Kuyruk yöneticisi adlarının ve nesne adlarının IBM i Integrated File System ' nin (IFS) gereksinimlerini karşılamak için dönüştürülmesi gerekebilir.

Bir kuyruk yöneticisi yaratıldığında IBM MQ , bir kuyruk yöneticisi kitaplığını bu kitaplık ile ilişkilendirir. Bu kuyruk yöneticisi kitaplığına benzersiz bir ad, en çok 10 karakter uzunluğunda, büyük ölçüde kullanıcı tanımlı kuyruk yöneticisi adına dayalı bir ad verilir. Hem kuyruk yöneticisi hem de kuyruk yöneticisi kitaplığı, `/QIBM/UserData/mqm`önekine sahip kuyruk yöneticisi adına dayalı bir dizine yerleştirilecek. Aşağıda bir kuyruk yöneticisi, kuyruk yöneticisi kitaplığı ve dizin örneği verilmiştir:

Kuyruk yöneticisi adı	Turuncu
Kuyruk yöneticisi kitaplığı adı	QMORANGE
Dizin	`/QIBM/UserData/mqm/ORANGE`

Tüm kuyruk yöneticisi adları ve kuyruk yöneticisi kitaplık adları, `/QIBM/UserData/mqm/mqs.ini`dosyasındaki stanzas adlarına yazılır.

IBM MQ IFS dizinleri ve dosyaları

The IBM i Integrated File System (IFS) is used extensively by IBM MQ to store data. IFS ' ye ilişkin ek bilgi için *Integrated File System Introduction*başlıklı konuya bakın.

Her IBM MQ nesnesi (örneğin, bir kanal ya da kuyruk yöneticisi) bir dosya tarafından temsil edilir. Nesne adlarının geçerli dosya adları olması gerektiğinden, kuyruk yöneticisi nesne adını gerektiği yerde geçerli bir dosya adına dönüştürür.

Kuyruk yöneticisi dizinine giden yol aşağıdaki gibi biçimiyle oluşturulur:

- Kuyruk yöneticisi yapılanış kütüğünde tanımlı olan bir önek (`qm.ini`). Varsayılan önek `/QIBM/UserData/mqm`' dir.
- Hazır bilgi, `qmgrs`.
- Kuyruk yöneticisi adı, geçerli bir dizin adına dönüştürülen, kodlanmış bir kuyruk yöneticisi adı. For example, the queue manager `queue/manager` is represented by `queue&manager`.

Bu işleme ad dönüştürme adı verilir.

IFS kuyruk yöneticisi adı dönüşümü

IBM MQ' ta, bir kuyruk yöneticisinde en çok 48 karakterden oluşan bir ad verebilirsiniz.

Örneğin, bir kuyruk yöneticisini `QUEUE/MANAGER/ACCOUNTING/SERVICE`Solarak adlayabilirsiniz. Her kuyruk yöneticisi için bir kitaplığın yaratıldığı şekilde, her kuyruk yöneticisi de bir dosya tarafından temsil edilir. EBCDIC ' deki varyant kod noktaları nedeniyle, adda kullanılabilecek karakterlerle ilgili sınırlamalar vardır. Sonuç olarak, nesneleri temsil eden IFS dosyalarının adları, dosya sisteminin gereklerini karşılamak için otomatik olarak dönüştürülür.

Using the example of a queue manager with the name `queue/manager`, transforming the character / to &, and assuming the default prefix, the queue manager name in IBM MQ for IBM i becomes `/QIBM/UserData/mqm/qmgrs/queue&manager`.

Nesne adı dönüşümü

Nesne adlarının geçerli dosya sistemi adları olması gerekmez, bu nedenle nesne adlarının dönüştürülmesi gerekebilir. Kuyruk yöneticisi adları için kullanılan yöntem, kuyruk yöneticisi adlarından farklıdır; ancak, her makine için birkaç kuyruk yöneticisi adı olmasına rağmen, her kuyruk yöneticisi için çok sayıda başka nesne olabilir. Dosya sisteminde yalnızca süreç tanımlamaları, kuyruklar ve ad listeleri gösterilir; kanallar bu önemli noktalardan etkilenmez.

Dönüştürme işlemi tarafından yeni bir ad oluşturulduğunda, özgün nesne adıyla basit bir ilişki yoktur. You can use the DSPMQMOBJN command to view the transformed names for IBM MQ objects.

Nesne öznitelikleri

Bir nesnenin özellikleri, bir nesnenin öznitelikleri tarafından tanımlanır. Bazı kişiler, yalnızca görüntüleyebileceğiniz diğer kişileri de belirleyebilirsiniz.

For example, the maximum message length that a queue can accommodate is defined by its **MaxMsgLength** attribute; you can specify this attribute when you create a queue. **DefinitionType** özniteliği, kuyruğun nasıl yaratıldığını belirtir; yalnızca bu özniteliği görüntüleyebilirsiniz.

IBM MQ' ta, bir özniteliğe gönderme yapmak için iki yol vardır:

- Örneğin, PCF adını (örneğin, **MaxMsgLength**) kullanarak.
- MQSC komut adını kullanarak, örneğin, MAXMSGL.

z/OS Kuyruk paylaşım grupları

Queue managers that can access the same set of shared queues form a group called a *kuyruk paylaşım grubu* (QSG), and they communicate with each other using a coupling facility (CF) that stores the shared queues.

Paylaşılan kuyruk, bir kuyruk paylaşım grubunda yer alan bir ya da daha fazla kuyruk yöneticisi tarafından erişilebilen iletileri içeren bir yerel kuyruk türüdür. Aynı kuyruk yöneticisi kullanılarak, kuyruğun birden çok uygulama tarafından paylaşıldığı bir kuyruk aynı değil.

Kuyruk paylaşım grupları en çok dört karakterden oluşan bir ada sahiptir. Adın ağınızda benzersiz olması ve kuyruk yöneticisi adlarından farklı olması gerekir.

Kuyruk paylaşım grupları tam olarak nesneler değildir, ancak kolaylık sağlamak amacıyla burada belirtilir.

Önemli: Paylaşılan kuyruklar ve kuyruk paylaşım grupları yalnızca IBM MQ for z/OSüzerinde desteklenir.

İlgili kavramlar

“Paylaşılan kuyruklar ve kuyruk paylaşım grupları” sayfa 160
You can use shared queues and queue sharing groups, to implement high availability of IBM MQ resources. Paylaşılan kuyruklar ve kuyruk paylaşım grupları, z/OS altyapısındaki IBM MQ for z/OS için benzersiz işlevlerdir.

Sistem varsayılan nesneleri

Sistem varsayılan nesneleri , bir kuyruk yöneticisi yaratıldığında otomatik olarak yaratılan bir nesne tanımlamaları kümesidir.

Bu nesne tanımlarından herhangi birini, kuruluşunuzdaki uygulamalarda kullanmak üzere kopyalayabilir ve değiştirebilirsiniz.

Varsayılan nesne adları kök SYSTEM ' e sahiptir; örneğin, varsayılan yerel kuyruk SYSTEM.DEFAULT.LOCAL.QUEUE(Kuyruk) ve varsayılan alıcı kanalı SYSTEM.DEF.RECEIVER Bu nesneleri yeniden adlandıramazsınız; bu adlara ilişkin varsayılan nesneler gereklidir.

Bir nesne tanımladığınızda, belirtmediğiniz öznitelikler belirtik olarak uygun varsayılan nesneden kopyalanır. Örneğin, yerel bir kuyruk tanımlıyorsanız, belirtmediğiniz öznitelikler varsayılan kuyruk SYSTEM.DEFAULT.LOCAL.QUEUE.

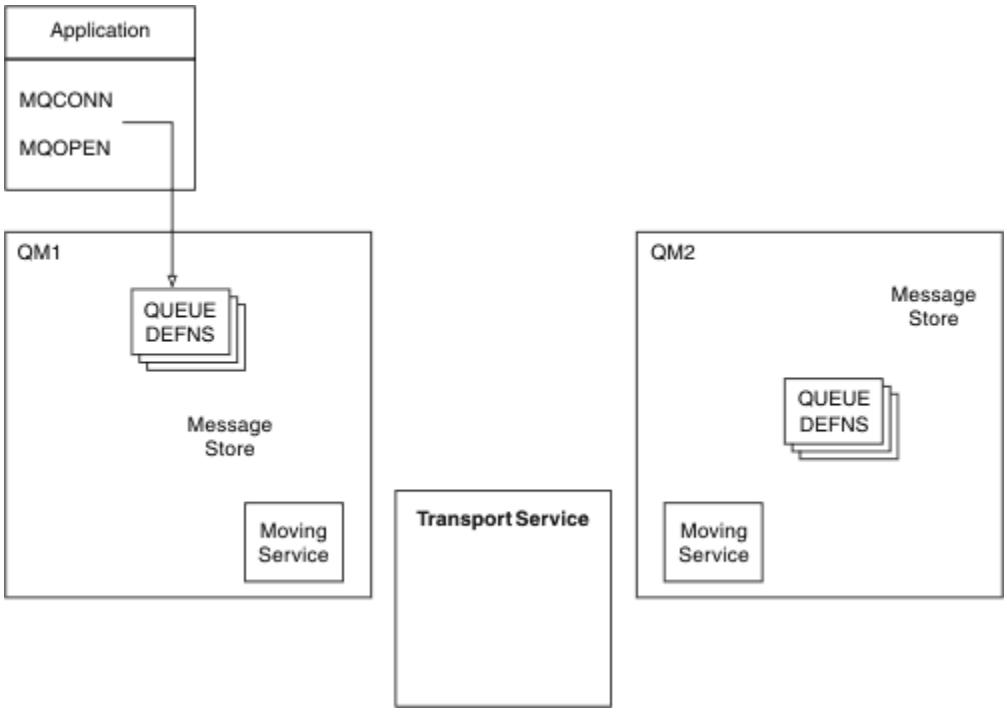
İlgili başvurular

Sistem ve varsayılan nesneler

Dağıtılmış kuyruğa alma ve kümeler

Dağıtılmış kuyruklama, bir kuyruk yöneticisinden başka bir kuyruk yöneticisinden ileti gönderme anlamına gelir. Alma kuyruk yöneticisi aynı makinede ya da başka bir yerde olabilir; yakınlarda ya da dünyanın diğer tarafında olabilir. Yerel kuyruk yöneticiyle aynı platformda çalışıyor olabilir ya da IBM MQtarafından desteklenen platformların herhangi birinde olabilir. You can manually define all the connections in a distributed queuing environment, or you can create a cluster and let IBM MQ define much of the connection detail for you.

Dağıtılmış kuyruklama



Şekil 4. Dağıtılmış kuyruğa alma bileşenlerine genel bakış

Önceki şekildeki:

- Bir uygulama, bir kuyruk yöneticisine bağlanmak için MQCONN çağrısını kullanır. Daha sonra, uygulama, kuyruktan ileti koyabilmesi için bir kuyruğu açmak üzere MQOPER çağrısını kullanır.
- Her kuyruk yöneticisinin, her bir kuyruğu için bir tanımlaması vardır. *Yerel kuyruklar* (bu kuyruk yöneticisi tarafından barındırılan) ve *uzak kuyrukların* tanımlarının (diğer kuyruk yöneticilerinin barındırdığı) tanımlarına sahip olabilir.
- İletiler uzak bir kuyruğa yollanırsa, yerel kuyruk yöneticisi bunları bir *iletim kuyruğunda*tutar ve bunlar uzak kuyruk yöneticisine iletilinceye kadar, bir ileti depolarında saklanır.
- Her kuyruk yöneticisi, kuyruk yöneticisinin diğer kuyruk yöneticileriyle iletişim kurmak için kullandığı, *taşıma hizmeti*olarak bilinen iletişim yazılımını içerir.
- *transport service* , kuyruk yöneticilerinden bağımsızdır ve aşağıdakilerden herhangi biri olabilir (altyapıya bağlı olarak):
  - Systems Network Architecture Advanced Program-to Program Communication (SNA APPC)
  - İletim Denetimi İletişim Kurası/Internet Protocol (TCP/IP)
  - Ağ Temel Giriş/Çıkış Sistemi (NetBIOS)
  - Sıralı Paket Değişimi (SPX)

İleti göndermek için gereken bileşenler

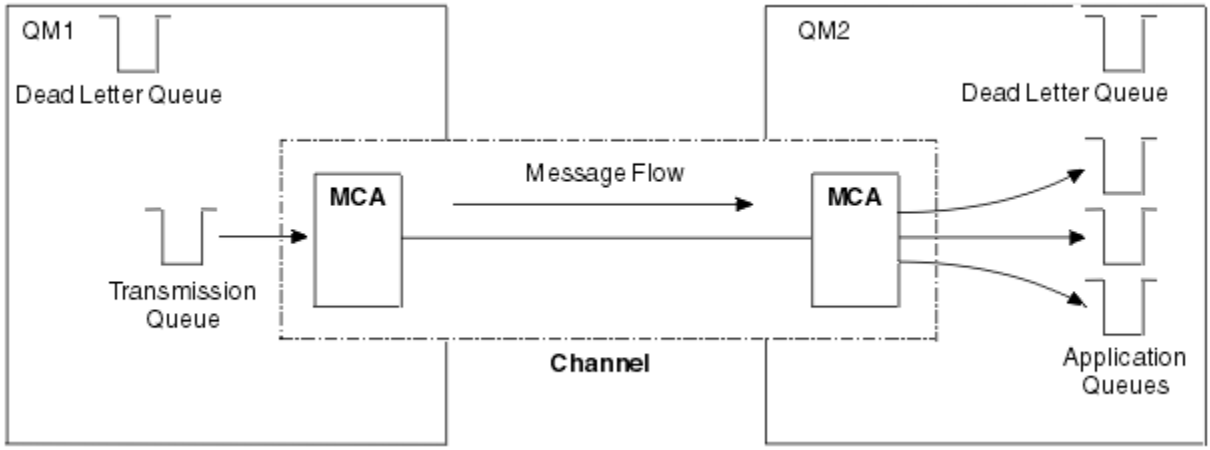
Uzak kuyruk yöneticisine bir ileti gönderilecekse, yerel kuyruk yöneticisinin bir *iletim kuyruğu* ve bir *kanal*için tanımlamalara gerek vardır. Kanal, iki kuyruk yöneticisi arasında tek yönlü bir iletişim bağlantısıdır. Uzak kuyruk yöneticisinde herhangi bir kuyruğa yollanmak üzere gönderilen iletileri taşıyabilir.

Bir kanalın her bir ucu ayrı bir tanımlamaya sahiptir; örneğin, gönderme sonu ya da alma sonu gibi. Basit bir kanal, yerel kuyruk yöneticisinde *gönderen* kanal tanımlamasından ve uzak kuyruk yöneticisinde bir *alıcı* kanal tanımlamasından oluşur. Bu iki tanım aynı ada sahip olmalı ve birlikte bir kanal oluştururlar.

İletileri gönderme ve alma işlemlerini işleyen yazılım, *Message Channel Agent* (MCA) adı verilir. Bir kanalın her ucunda bir *ileti kanalı aracısı* (MCA) vardır.

Her kuyruk yöneticisinin bir *ölü-mektup kuyruğu* olmalıdır (*teslim edilmemiş ileti kuyruğu* olarak da bilinir). Hedefe teslim edilemezlerse, iletiler bu kuyruğa konmaya devam eder.

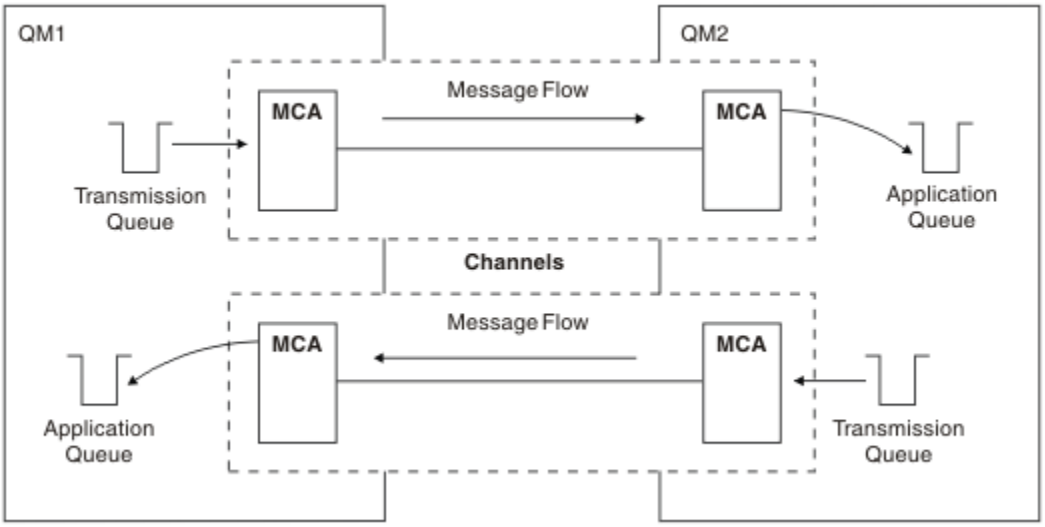
Aşağıdaki şekil, kuyruk yöneticileri, iletim kuyrukları, kanallar ve MCA ' lar arasındaki ilişkiyi göstermektedir:



Şekil 5. İletilerin gönderilmesi

İleti döndürmek için gereken bileşenler

Uygulamanız uzak kuyruk yöneticisinden iletilerin döndürülemesini gerektiriyorsa, aşağıdaki şekilde gösterildiği gibi, kuyruk yöneticileri arasında ters yönde çalışabilmek için başka bir kanal tanımlamanız gerekir:



Şekil 6. İletileri her iki yönde gönderme

Kümeler

Dağıtılmış bir kuyruklama ortamındaki tüm bağlantıları el ile tanımlamak yerine, bir küme içindeki kuyruk yöneticilerini gruplayabilirsiniz. Bunu gerçekleştirdiğinizde, kuyruk yöneticileri, anasistemin diğer kuyruk yöneticilerine, belirtik kanal tanımlamalarına, uzak kuyruk tanımlamalarına ya da her hedefe ilişkin iletim kuyruklarına gereksinim duymadan, kümedeki diğer kuyruk yöneticilerine kullanılabilmesini sağlayabilirler. Bir kümedeki her kuyruk yöneticisinin, iletileri kümedeki diğer bir kuyruk yöneticisine ileten tek bir iletim kuyruğu vardır. Her kuyruk yöneticisi için, yalnızca bir küme alıcı kanalı ve bir kümeli gönderici kanalı tanımlamanız gerekir; ek kanallar da küme tarafından otomatik olarak yönetilir.

Bir IBM MQ istemcisi, diğer bir kuyruk yöneticisine bağlanabildiği gibi, bir kümenin parçası olan bir kuyruk yöneticisine bağlanabilirler. El ile yapılandırılmış dağıtılmış kuyruğa alma işlemi olarak, herhangi bir kuyruk yöneticisinde kuyruğa ileti yerleştirmek için MQPUT çağrısını kullanıyorsunuz. Yerel kuyruktan ileti almak için MQGET çağrısını kullanıyorsunuz.

Kümeleri destekleyen platformlarda bulunan kuyruk yöneticileri, bir kümenin parçası olmak zorunda değildir. Dağıtılmış kuyruğa alma işlevini, kümeleri kullanarak ya da kullanmak yerine el ile yapılandırmayı sürdürebilirsiniz.

Kümelerin kullanılmasının yararları

Kümeleme, iki temel avantaj sağlar:

- Kümeler, genellikle kanallar, iletim kuyrukları ve konfigürasyonu tanımlanacak uzak kuyruklar için birçok nesne tanımlaması gerektiren IBM MQ ağlarının yönetimini basitleştirir. Bu durum, özellikle çok sayıda kuyruk yöneticisinin birbiriyle bağlantılı olması gereken büyük, potansiyel olarak değişen ağlarda da geçerlidir. Bu mimari, özellikle yapılandırılması ve etkin bir şekilde sürdürülmesi güçtür.
- Kümeler, ileti trafiğindeki iş yükünü kümedeki kuyruklar ve kuyruk yöneticilerine dağıtmak için kullanılabilir. Bu tür bir dağıtım, tek bir kuyruğun ileti iş yükünün, birden çok kuyruk yöneticisi üzerinde bulunan o kuyruğun eşdeğer eşgörünümlerine dağıtılmasını sağlar. İş yükünün bu dağılımı, sistem hatalarına karşı daha fazla esneklik sağlamak ve özellikle bir sistemdeki etkin ileti akışlarının ölçekleme performansını artırmak için kullanılabilir. Böyle bir ortamda, dağıtılmış kuyrukların her bir eşgörünümü, iletileri işleyen uygulamaları tüketir. Daha fazla bilgi için İş yükü yönetimi için kümeleri kullanmabaşlıklı konuya bakın.

Kümenin bir kümede nasıl yönlendirilir

Bir kümeyi, vicdani sistem yöneticisi tarafından sağlanan kuyruk yöneticilerinin ağı olarak düşünebilirsiniz. Bir küme kuyruğu tanımladığınızda, sistem denetimcisi diğer kuyruk yöneticilerindeki gerektiğinde otomatik olarak karşılık gelen uzak kuyruk tanımlarını yaratır.

You do not need to make transmission queue definitions because IBM MQ provides a transmission queue on each queue manager in the cluster. Bu tek iletim kuyruğu, iletilerin başka bir kuyruk yöneticisine ileti taşımak için kullanılabilir. Tek bir iletim kuyruğunu kullanmak üzere sınırlanmanız yoktur. Kuyruk yöneticisi, bir kümedeki her kuyruk yöneticisine giden iletileri ayırmak için birden çok iletim kuyruğu kullanabilir. Tipik olarak, kuyruk yöneticisi tek bir küme iletim kuyruğu kullanır. You can change the queue manager attribute DEFCLXQ, so that a queue manager uses a different cluster transmission queue for each queue manager in a cluster. Küme iletim kuyruklarını el ile de tanımlayabilirsiniz.

Bir kümeye katılan tüm kuyruk yöneticileri bu şekilde çalışmayı kabul eder. kendileri ve ev sahibi oldukları kuyruklar hakkında bilgi gönderip, küba ' nın diğer üyeleri hakkında bilgi alıyorlar.

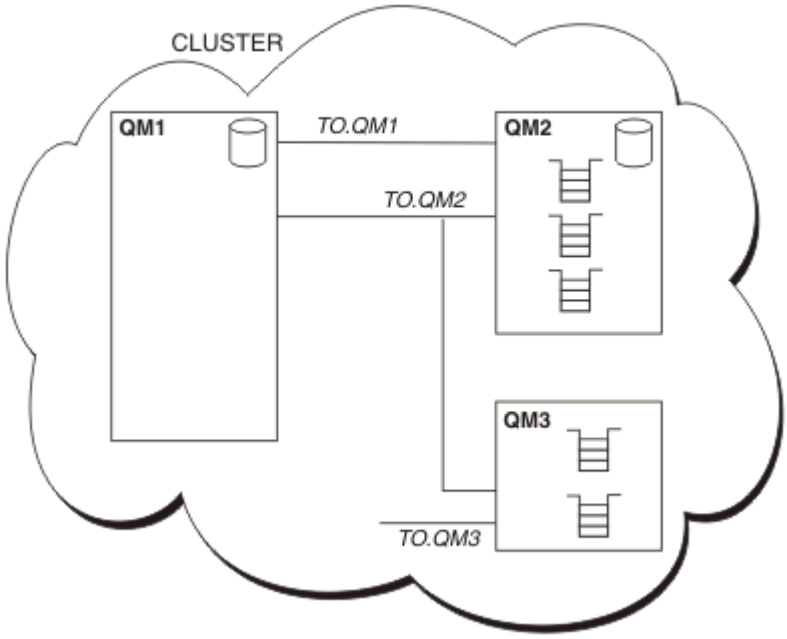
Bir kuyruk yöneticisi kullanılamaz duruma geldiğinde hiçbir bilginin kaybolmadığından emin olmak için, kümede iki kuyruk yöneticisi belirttiğinizde *tam havuzlar*olarak işlev görmenizi sağlar. Bu kuyruk yöneticileri, kümedeki tüm kuyruk yöneticilerine ve kuyruklara ilişkin eksiksiz bilgi kümesi içerir. Kümedeki diğer tüm kuyruk yöneticileri yalnızca, iletileri değiş tokuş ettikleri kuyruk yöneticilerine ve kuyruklara ilişkin bilgileri saklar. Bu kuyruk yöneticileri *kısmi havuzlar*olarak bilinir. Daha fazla bilgi için “Küme havuzu” sayfa 53 başlıklı konuya bakın.

Bir kümenin parçası olmak için, kuyruk yöneticisinin iki kanalı olmalıdır; bir küme gönderici kanalı ve bir küme alıcı kanalı olmalıdır:

- Küme-gönderici kanalı, gönderici kanalı gibi bir iletişim kanalıdır. Bir kuyruk yöneticisi üzerinde, zaten kümenin üyesi olan bir tam havuza bağlanmak için, bir küme gönderici kanalı el ile oluşturmanız gerekir.
- Küme-alıcı kanalı, alıcı kanalı gibi bir iletişim kanalıdır. Bir küme-alıcı kanalı el ile yaratmanız gerekir. Kanal, kuyruk yöneticisinin küme iletişimlerini almasını sağlayan düzenek olarak işlev görür.

Bu kuyruk yöneticisi ile kümenin diğer üyeleri arasında iletişim için gerekli olan diğer tüm kanallar otomatik olarak yaratılır.

Aşağıdaki şekil, KÜME adı verilen bir kümenin bileşenlerini göstermektedir:



Şekil 7. Bir kuyruk yöneticisi kümesi

- KüME, üç kuyruk yöneticisi, QM1, QM2ve QM3içerir.
- QM1 ve QM2 , kümedeki kuyruk yöneticilerine ve kuyruklara ilişkin bilgilerin tam havuzlarını içerir.
- QM2 ve QM3 bazı küme kuyrukları (yani, kümedeki başka bir kuyruk yöneticisi tarafından erişilebilir olan kuyruklar) içerir.
- Her kuyruk yöneticisinin, ileti alabileceği TO.qmgr adlı bir küme alıcı kanalına sahiptir.
- Her kuyruk yöneticisinin, havuz kuyruğu yöneticilerinden birine bilgi gönderebileceği bir küme gönderici kanalı da vardır.
- QM1 and QM3 send to the repository at QM2 and QM2 sends to the repository at QM1.

Dağıtılmış kuyruklama bileşenleri

dağıtık queuing ' in bileşenleri ileti kanalları, mesaj kanalı ajanları, iletim kuyrukları, kanal başlatıcıları ve dinleyicileri ve kanal çıkış programları. Bir ileti kanalının her bir ucunun tanımı, çeşitli tiplerden biri olabilir.

İleti kanalları, iletileri bir kuyruk yöneticisinden diğerine taşıyan kanallardır. İleti kanallarını MQI kanallarıyla karıştırmayın. İki tip MQI kanalı, sunucu bağlantısı (SVRCONN) ve istemci-bağlantısı (CLNTCONN) vardır. Ek bilgi için Kanallarbaşlıklı konuya bakın.

Bir ileti kanalının her bir ucunun tanımı aşağıdaki tiplerden biri olabilir:

- Gönderen (SDR)
- Günlük Nesnesi (RCVR)
- Sunucu (SVR)
- İstek Sahibi (RQSTR)
- Küme gönderen (CLUSSDR)
- Küme alıcısı (CLUSRCVR)

Bir ileti kanalı, bir uçta tanımlanan bu tiplerden biri ve diğer uçta uyumlu bir tip kullanılarak tanımlanır. Olası birleşimler şunlardır:

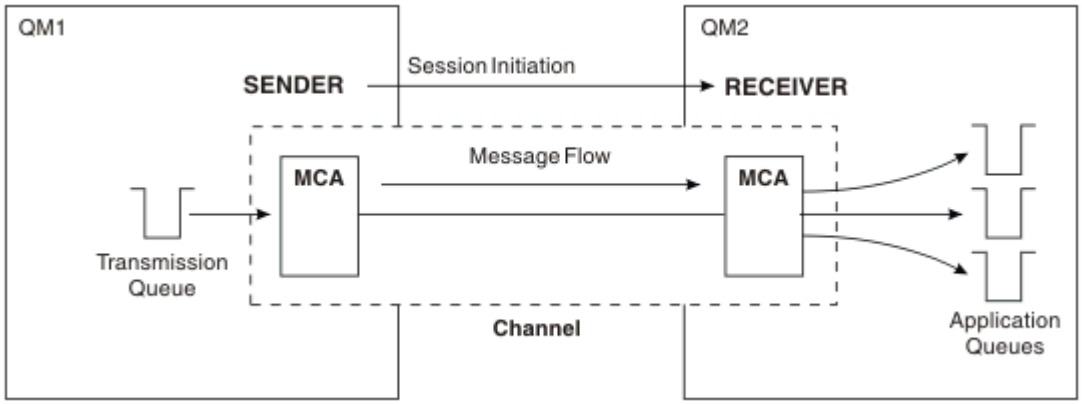
- Gönderen-alıcı
- İstekte Bulunan-sunucu

- İsteyenin-gönderen (geri çağırma)
- Sunucu Alıcısı
- Küme gönderen-küme alıcısı

Gönderici alıcı kanalı yaratılmasına ilişkin ayrıntılı yönergeler için Kanalların tanımlanmasıbaşlıklı konu yer alır. Gönderen alıcı kanallarını ayarlamak için gereken parametrelere ilişkin örnekler için, altyapınız için geçerli olan Örnek yapılandırma bilgileri başlıklı konuya bakın. Herhangi bir tipte kanal tanımlamak için gereken parametreler için bkz. KANAL TANıMLAMA.

Gönderen-alıcı kanalları

Bir sistemde gönderici, diğer sisteme ileti gönderebilmesi için kanalı başlatır. Gönderen, alıcıya kanalın diğer ucunda başlamasını ister. Gönderen, iletim kuyruğundan alıcıya ileti gönderir. Alıcı, iletileri hedef kuyruğa koyar. Şekil 8 sayfa 45 bunu gösterir.

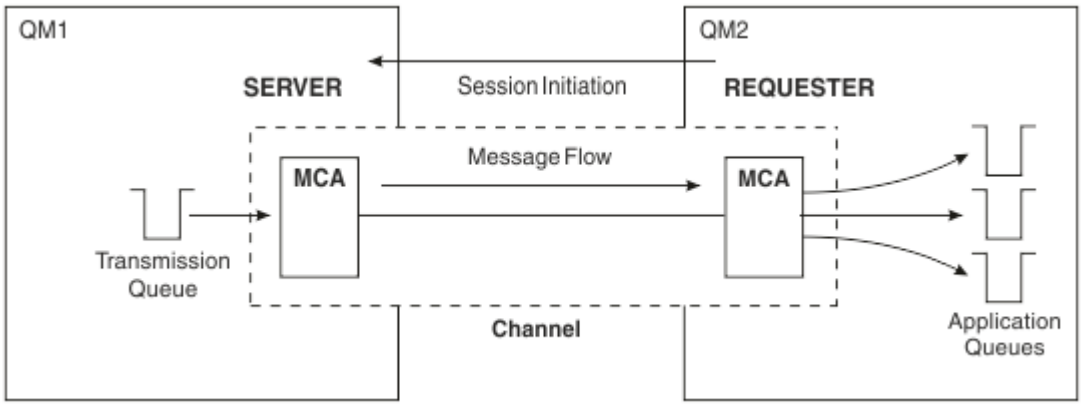


Şekil 8. Gönderici-alıcı kanalı

İstekçi-sunucu kanalları

Bir sistemdeki istekte bulunan kişi, diğer sistemden ileti alabilmesi için kanala başlar. İstekte bulunanın, sunucunun diğer ucundaki sunucuyu başlatmasını ister. Sunucu, ileti istekçisine, kanal tanımında tanımlanan iletim kuyruğundan ileti gönderir.

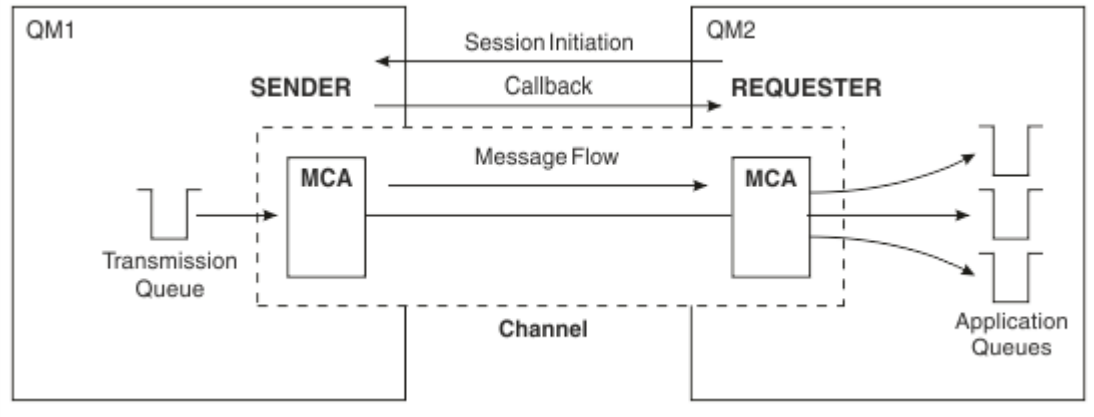
Bir sunucu kanalı, iletişimi başlatabilir ve istekte bulunan bir istekçiye ileti gönderebilir. Bu, yalnızca kanal tanımlamasında belirtilen ortağın bağlantı adına sahip sunucu kanalları olan *tam olarak nitelenmiş* sunucular için geçerlidir. Tam olarak nitelenmiş bir sunucu, bir istekçiyle başlatılabilir ya da bir istekçiyle iletişim başlatabilir.



Şekil 9. İstekte bulunan-sunucu kanalı

İsteyenin-gönderici kanalları

İsteği sunan kişi kanalı başlatır ve gönderici aramayı sonlandırır. Daha sonra, gönderici, iletişim kanalını kanal tanımındaki bilgilere göre yeniden başlatır (*geri çağırma* olarak bilinir). İleti, ileti kuyruğundan istekte bulunana ileti gönderir.



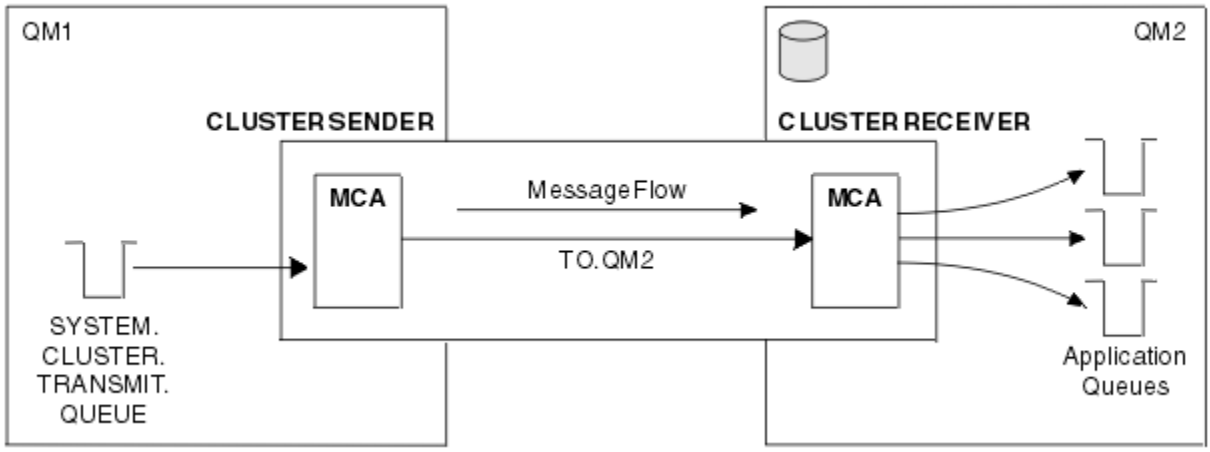
Şekil 10. İstekte bulunanın-gönderen kanalı

Sunucu alıcı kanalları

Bu, gönderici alıcı gibidir, ancak yalnızca *tam nitelikli* sunucuları için geçerlidir; bu, kanal tanımlamasında belirtilen ortağın bağlantı adına sahip sunucu kanallarından olur. Kanal başlatma, bağlantının sunucu sonunda başlatılmalıdır. Bu şekilde, Şekil 8 sayfa 45içindeki şekil de gösterilmektedir.

Küme-gönderici kanalları

Bir kümede, her kuyruk yöneticisinin, küme bilgilerini tam havuz kuyruğu yöneticilerinden birine gönderebileceği bir küme gönderici kanalı vardır. Kuyruk yöneticileri, diğer kuyruk yöneticilerine, küme gönderen kanallarındaki diğer kuyruk yöneticilerine de ileti gönderebilirler.



Şekil 11. Bir küme-gönderici kanalı

Küme-alıcı kanalları

Bir kümede, her kuyruk yöneticisinin, küme ile ilgili iletileri ve bilgileri alabileceği bir kümeleme alıcısı kanalı vardır. Bu şekilde, Şekil 11 sayfa 46içindeki şekil de gösterilmektedir.

Ölü-harfli kuyruklar

Gönderilmeyen iletiler kuyruğu (ya da teslim edilemeyen ileti kuyruğu), iletilerin doğru hedefe yönlendirilememesi durumunda gönderileceği kuyruktır. Her kuyruk yöneticisinin tipik olarak bir ölü-mektup kuyruğu vardır.

Bazen *teslim edilemeyen ileti kuyruğu*olarak adlandırılan bir *ölü-mektup kuyruğu* (DLQ), hedef kuyruklarına teslim edilemeyen iletiler için bir tutma kuyruğunudur; örneğin, kuyruk var olmadığı için ya da dolu olduğu için. Veri dönüştürme hataları için, bir kanalın gönderme sonunda da ölme-mektup kuyrukları kullanılır. Bir ağdaki her kuyruk yöneticisinin genellikle bir yerel kuyruğu vardır. Bu nedenle, doğru hedeflerine teslim edilemeyen iletiler daha sonra geri alınabilmesi için saklanabilirler.

İletiler, kuyruk yöneticilerine, ileti kanalı aracılarına (MCA 'lar) ve uygulamalara göre DLQ' ya yerleştirilebilir. All messages on the DLQ must be prefixed with a *dead-letter üstbilgisi* structure, MQDLH. MQDLH yapısının *Reason* (Neden) alanı, iletinin DLQ ' da neden olduğunu belirten bir neden kodu içerir.

Genellikle her kuyruk yöneticisi için bir dead-letter kuyruğu tanımlamanız gerekir. Bunu yapmazsanız ve MCA bir ileti koyamazsa, ileti iletim kuyruğunda bırakılır ve kanal durdurulur. Ayrıca, hızlı, kalıcı olmayan iletiler de varsa (bkz. Hızlı, kalıcı olmayan iletiler) teslim edilemiyor ve hedef sistemde hiçbir ölü-mektup kuyruğu yok, bu iletiler atılır.

Ancak, ölü harf kuyrukları kullanılması, iletilerin teslim edildiği sırayı etkileyebilir ve bunları kullanmamayı tercih edebilirsiniz.

Uzak kuyruk tanımlamaları

Uzak kuyruk tanımları, başka bir kuyruk yöneticisinin sahip olduğu kuyruklara ilişkin tanımlardır.

Uygulamalar iletileri yalnızca yerel kuyruklardan alabilirken, iletileri yerel kuyruklara ya da uzak kuyruklara yerleştirebilir. Bu nedenle, yerel kuyruklarının her biri için bir tanım olduğu gibi, bir kuyruk yöneticisinin *uzak kuyruk tanımlamaları*olabilir. Uzak kuyruk tanımlamalarının yararı, uzak kuyruğun ya da uzak kuyruk yöneticisinin adını ya da iletim kuyruğunun adını belirtmeden, bir uygulamanın uzak kuyruğa bir ileti yerleştirebilmesini sağlar. Uzak kuyruk tanımları size konum bağımsızlığı sağlar.

Daha sonra açıklanan uzak kuyruk tanımları için başka kullanımlar da vardır.

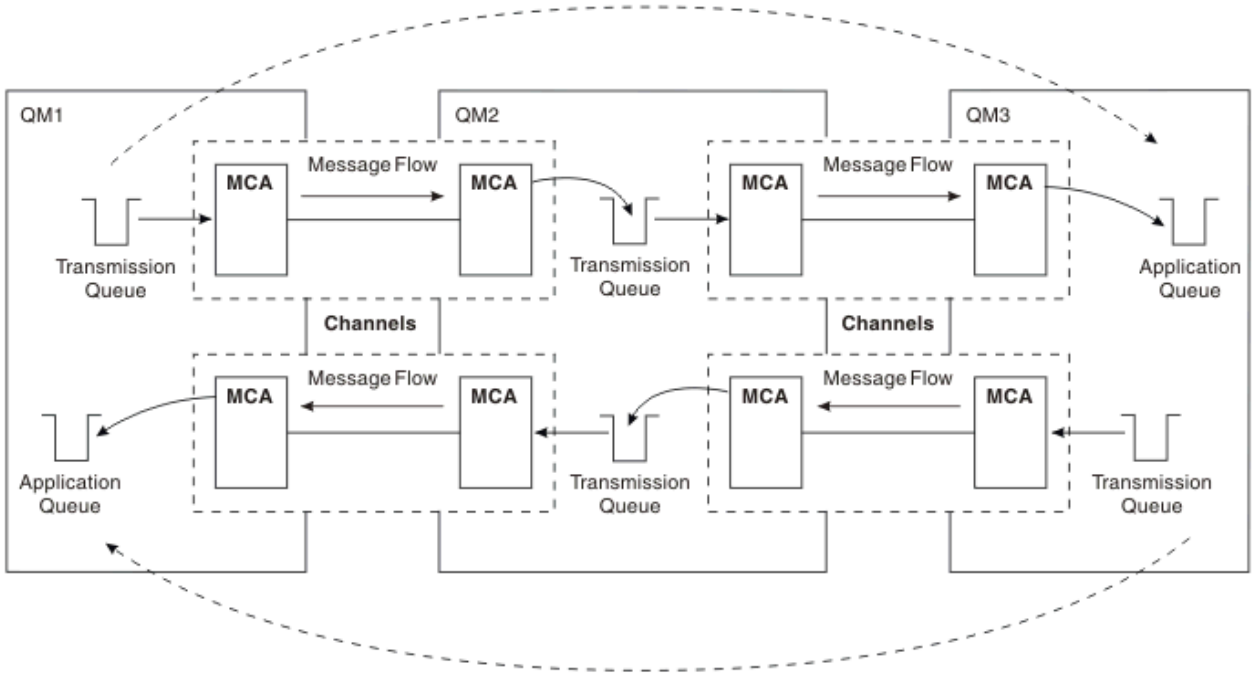
Uzak kuyruk yöneticisine nasıl geçilecek?

Her bir kaynak ve hedef kuyruk yöneticisi arasında her zaman tek bir kanal olmayabilir. Farklı kanallar ve kümeleme kullanarak, çok sekmeli, paylaşım kanalları da dahil olmak üzere, iki arasında bağlantı kurmanın başka yolları da vardır.

Çoklu sekme

Kaynak kuyruk yöneticisi ile hedef kuyruk yöneticisi arasında doğrudan iletişim bağlantısı yoksa, hedef kuyruk yöneticisine giden yolda bir ya da daha çok *ara kuyruk yöneticisi* yoluyla geçiş yapmak mümkündür. Bu, *çoklu sekme noktası*olarak bilinir.

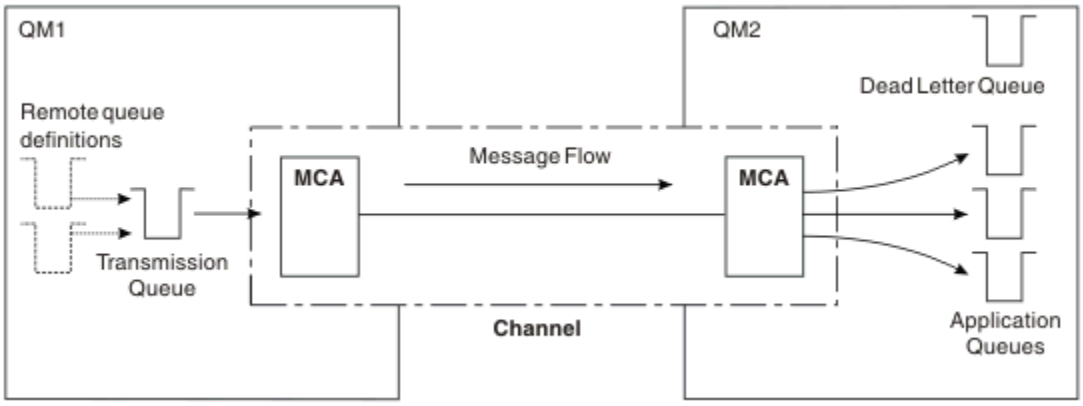
Tüm kuyruk yöneticileri arasındaki kanalları ve ara kuyruk yöneticilerindeki iletim kuyruklarını tanımlamanız gerekir. Bu, Şekil 12 sayfa 48içinde gösterilir.



Şekil 12. Ara kuyruk yöneticilerinden geçme

Kanalları paylaşma

Uygulama tasarımcısı olarak, uygulamalarınızı kuyruk adı ile birlikte uzak kuyruk yöneticisi adını belirtme ya da her uzak kuyruk için bir *uzak kuyruk tanımlaması* yaratma seçeneğiniz vardır. Bu tanımlama, uzak kuyruk yöneticisi adını, kuyruk adını ve iletim kuyruğunun adını içerir. Her iki yöntemle de, aynı uzak yerde bulunan tüm uygulama adresleme kuyruklarından gelen tüm iletiler, aynı iletim kuyruğundan gönderilen iletilerine sahip olur. Bu, Şekil 13 sayfa 48içinde gösterilir.



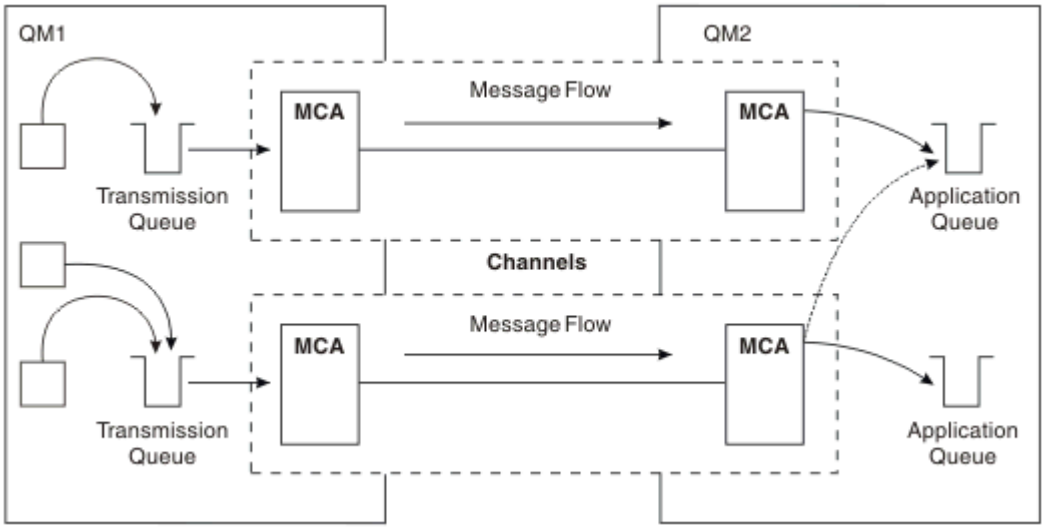
Şekil 13. İletim kuyruğunun paylaşılması

Şekil 13 sayfa 48 , birden çok uzak kuyruğa birden çok uygulamadaki iletilerin aynı kanalı kullanabileceğini gösterir.

Farklı kanalları kullanma

İki kuyruk yöneticisi arasında göndermek üzere farklı tiplere ilişkin iletileriniz varsa, iki kuyruk arasında birden çok kanal tanımlayabilirsiniz. Bazen, güvenlik amacıyla alternatif kanallara gereksinim duyarsınız ya da ileti trafiğindeki toplu olarak teslim hızını takas etmek için gerekir.

İkinci bir kanalı ayarlamak için başka bir kanal ve başka bir iletim kuyruğu tanımlamanız ve yeri ve iletim kuyruğu adını belirten bir uzak kuyruk tanımlaması yaratmanız gerekir. Daha sonra, uygulamalarınız her iki kanalı da kullanabilir, ancak iletiler yine de aynı hedef kuyruklara teslim edilir. Bu, Şekil 14 sayfa 49içinde gösterilir.



Şekil 14. Birden çok kanalı kullanma

Bir iletim kuyruğu belirlemek için uzak kuyruk tanımlarını kullandığınızda, uygulamalarınız **değil** ' un yerini (hedef kuyruk yöneticisi) kendileri belirlemelidir. Bu işlem yapılırsa, kuyruk yöneticisi uzak kuyruk tanımlamalarını kullanmaz. Uzak kuyruk tanımları size konum bağımsızlığı sağlar. Uygulamalar, kuyruğun nerede olduğunu bilmeden iletileri bir *mantıksal* kuyruğa koyabilir ve uygulamalarınızı değiştirmek zorunda kalmadan *fiziksel* kuyruğunu değiştirebilirsiniz.

Kümelemenin kullanılması

Bir küme içindeki her kuyruk yöneticisi bir küme alıcı kanalı tanımlar. Başka bir kuyruk yöneticisi o kuyruk yöneticisine bir ileti göndermek istediğinde, otomatik olarak ilgili küme gönderici kanalını tanımlar. Örneğin, bir kümedeki bir kuyruğun birden çok eşgörünümü varsa, küme gönderen kanalı, kuyruğu barındıran kuyruk yöneticilerinden herhangi birine tanımlanabilir. IBM MQ , bir iletiyi yönlendirmek üzere kullanılabilir bir kuyruk yöneticisi seçmek için çevrimsel sıralı bir yordam kullanan bir iş yükü yönetimi algoritması kullanır. Ek bilgi için Kümelerkonusuna bakın.

Adresleme bilgileri

Bir uygulama, uzak kuyruk yöneticisine gönderilen iletileri koyduğunda, yerel kuyruk yöneticisi, ileti kuyruğuna yerleştirmeden önce bir iletim üstbilgisi ekler. Bu üstbilgi, hedef kuyruk ve kuyruk yöneticisinin adını (*addressing information*) içerir.

Tek bir kuyruk yöneticisi ortamında, bir uygulama iletileri yerleştirmek için bir kuyruk açtığında hedef kuyruğun adresi oluşturulur. Hedef kuyruk aynı kuyruk yöneticisininde olduğundan, herhangi bir adresleme bilgisine gerek yoktur.

Dağıtılmış bir kuyruğa alma ortamında, kuyruk yöneticisinin yalnızca hedef kuyruk adını değil, aynı zamanda o kuyruğun yerini (yani kuyruk yöneticisi adını) ve o uzak yere (yani iletim kuyruğunu) yönlendirmesini de bilmesi gerekir. Bu adresleme bilgileri iletim üstbilgisinde yer alır. Alma kanalı, iletim üstbilgisini kaldırır ve hedef kuyruğu bulmak için bu bilgileri kullanır.

Uzak bir kuyruk tanımlaması kullanıyorsanız, uygulamalarınızın hedef kuyruk yöneticisinin adını belirtmesine gerek kalmamasını önleyebilirsiniz. Bu tanımlama, uzak kuyruğun adını, iletilerin gönderileceği uzak kuyruk yöneticisinin adını ve iletilerin iletilmekte kullanılan iletim kuyruğunun adını belirtir.

Diğer adlar nedir?

Diğer adlar, iletiler için bir hizmet kalitesi sağlamak için kullanılır. Kuyruk yöneticisi diğer adı, bir sistem denetimcisinin, uygulamalarınızı değiştirmenize neden olmadan, hedef kuyruk yöneticisinin adını değiştirmesini sağlar. Ayrıca, sistem denetimcisinin rotayı hedef kuyruk yöneticisine değiştirmesini ya da diğer kuyruk yöneticilerinden (çoklu sekme) geçen bir sayıyı içeren bir rota ayarlamasını da sağlar. Yanıtlama kuyruğu diğer adı, yanıtlar için bir hizmet kalitesi sağlar.

Kuyruk yöneticisi diğer adları ve yanıt kuyruğu diğer adları, boş bir RNAME değeri olan bir uzak kuyruk tanımlaması kullanılarak yaratılır. Bu tanımlamalar gerçek kuyruklar tanımlamaz; fiziksel kuyruk adlarını, kuyruk yöneticisi adlarını ve iletim kuyruklarını çözmek için kuyruk yöneticisi tarafından kullanılır.

Diğer ad tanımlamaları, boş bir RNAME (RNAME) ile karakterize edilir.

Kuyruk adı çözümlemesi

Kuyruk adı çözümlemesi her kuyruk yöneticisi her açıldığında her kuyruk yöneticisinde gerçekleşir. Amacı, hedef kuyruğu, hedef kuyruk yöneticisini (yerel olabilir) ve o kuyruk yöneticisine giden yolu (boş değerli olabilir) belirlemesidir. Çözümlenen ad üç kısıma sahiptir: kuyruk yöneticisi adı, kuyruk adı ve kuyruk yöneticisi uzaksa, iletim kuyruğu.

Bir uzak kuyruk tanımlaması varsa, gönderme yapılan diğer ad tanımlamalarına da gönderme yapılamaz. Uygulama tarafından sağlanan kuyruk adı, hedef kuyruk adı, uzak kuyruk yöneticisi ve uzak kuyruk tanımlamasında belirtilen iletim kuyruğu olarak çözülür. Kuyruk adı çözümlemesiyle ilgili daha ayrıntılı bilgi için Kuyruk adı çözümlemesibaşlıklı konuya bakın.

Uzak kuyruk tanımlaması yoksa ve kuyruk yöneticisi adı belirtildiyse ya da ad hizmetiyle çözülürse, kuyruk yöneticisi, belirtilen kuyruk yöneticisi adıyla eşleşen bir kuyruk yöneticisi diğer adı tanımlaması olup olmadığını görür. Varsa, içindeki bilgiler kuyruk yöneticisi adını hedef kuyruk yöneticisinin adına çözmek için kullanılır. Kuyruk yöneticisi diğer adı tanımlaması, hedef kuyruk yöneticisine iletim kuyruğunu saptamak için de kullanılabilir.

Çözülen kuyruk adı yerel bir kuyruk değilse, hem kuyruk yöneticisi adı, hem de kuyruk adı, uygulama tarafından iletim kuyruğuna eklenen her iletinin iletim üstbilgisine eklenir.

Uzak kuyruk tanımlaması ya da kuyruk yöneticisi diğer adı tanımlaması tarafından değiştirilmedikçe, kullanılan iletim kuyruğunun adı, çözülmüş kuyruk yöneticiyle aynı adı kullanır. Böyle bir iletim kuyruğunu tanımlamadıysanız, ancak bir varsayılan iletim kuyruğu tanımladıysanız, bu kullanılır.

z/OS z/OS üzerinde çalışan kuyruk yöneticilerinin adları dört karakterle sınırlıdır.

Kuyruk yöneticisi diğer ad tanımlamaları

Kuyruk yöneticisi diğer adı tanımlamaları, iletiyi koymak için kuyruk açan bir uygulama, kuyruk adını **ve** kuyruk yöneticisi adını belirtir.

Kuyruk yöneticisi diğer adı tanımlamalarının üç kullanımı vardır:

- İletileri gönderirken, kuyruk yöneticisi adını yeniden eşlerken
- İleti gönderirken, iletim kuyruğunu değiştirme ya da belirleme
- İletileri alırken, yerel kuyruk yöneticisinin bu iletiler için hedeflenen hedef olup olmadığını belirleme

Giden iletileri-kuyruk yöneticisi adının yeniden eşlenmesi

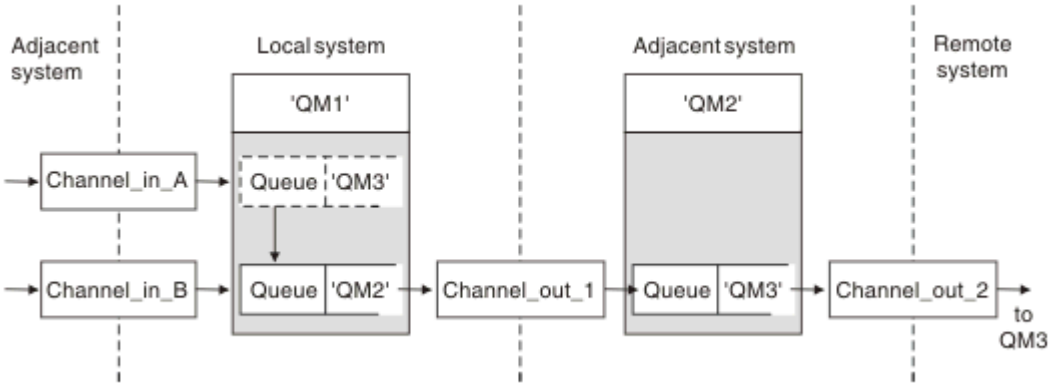
Kuyruk yöneticisi diğer adı tanımlamaları, bir MQOPEN çağrısında belirtilen kuyruk yöneticisi adını yeniden eşlemek için kullanılabilir. For example, an MQOPEN call specifies a queue name of THISQ and a queue manager name of YOURQM. Yerel kuyruk yöneticisinde, aşağıdaki örnek gibi bir kuyruk yöneticisi diğer adı tanımlaması vardır:

```
DEFINE QREMOTE (YOURQM) RQMNAME(REALQM)
```

This shows that the real queue manager to be used, when an application puts messages to queue manager YOURQM, is REALQM. Yerel kuyruk yöneticisi REALQMise, bu ileti, iletileri yerel bir kuyruk olan THISQkuyruğuna yerleştirir. Yerel kuyruk yöneticisi REALQMolarak adlandırılmamışsa, iletiyi REALQMadlı bir iletim kuyruğuna yönlendirir. Kuyruk yöneticisi iletim üstbilgisini YOURQMyerine REALQM demek için değiştirir.

Giden iletiler-iletim kuyruğunu değiştirme ya da belirleme

Şekil 15 sayfa 51 , iletilerin kuyruk yöneticisine (QM1 kuyruk yöneticisi QM3) kuyruk adlarını gösterentransmissionkuyruk yöneticisine vardığı bir senaryoya ilişkin bir senaryo gösterir. In this scenario, QM3 is reachable by multi-hopping through QM2.



Şekil 15. Kuyruk yöneticisi diğer adı

QM3 ile ilgili tüm iletiler kuyruk yöneticisi diğer adı ile QM1 saatinde yakalanır. The queue manager alias is named QM3 and contains the definition QM3 through transmission queue QM2. Tanım aşağıdaki örneğe benzer:

```
DEFINE QREMOTE (QM3) RNAME(' ') RQMNAME(QM3) XMITQ(QM2)
```

The queue manager puts the messages on transmission queue QM2 but does not alter the transmission queue header because the name of the destination queue manager, QM3, does not alter.

QM1 ' a gelen ve QM2 konumunda bir kuyruk adı içeren iletim üstbilgisini gösteren tüm iletiler de QM2 iletim kuyruğuna konabiliyor. Bu şekilde, farklı hedeflere sahip iletiler, uygun bir bitişik sisteme, varış noktalarına iletilmesi için, ortak bir iletim kuyruğunda toplanır.

Gelen iletiler-hedefin saptanması

Alıcı MCA, iletim üstbilgisinde başvurulan kuyruğu açar. Kuyruk yöneticisi diğer adı tanımlaması kuyruk yöneticisiyle aynı adı taşıyan bir diğerad tanımlaması varsa, iletim üstbilgisinde alınan kuyruk yöneticisi adı bu tanımlamadan RQMNAME değeri ile değiştirilir.

Bu işlemin iki kullanımı vardır:

- İletileri başka bir kuyruk yöneticisine yönlendiriyor
- Kuyruk yöneticisi adının yerel kuyruk yöneticiyle aynı olması için değiştirme

Yanıtlama kuyruğu diğer ad tanımlamaları

Yanıt kuyruğu diğer adı tanımlaması, ileti tanımlayıcısındaki yanıt bilgileri için alternatif adları belirtir. Bundan yararlanmanız, uygulamalarınızı değiştirmek zorunda kalmadan bir kuyruk ya da kuyruk yöneticisinin adını değiştirebilmendir.

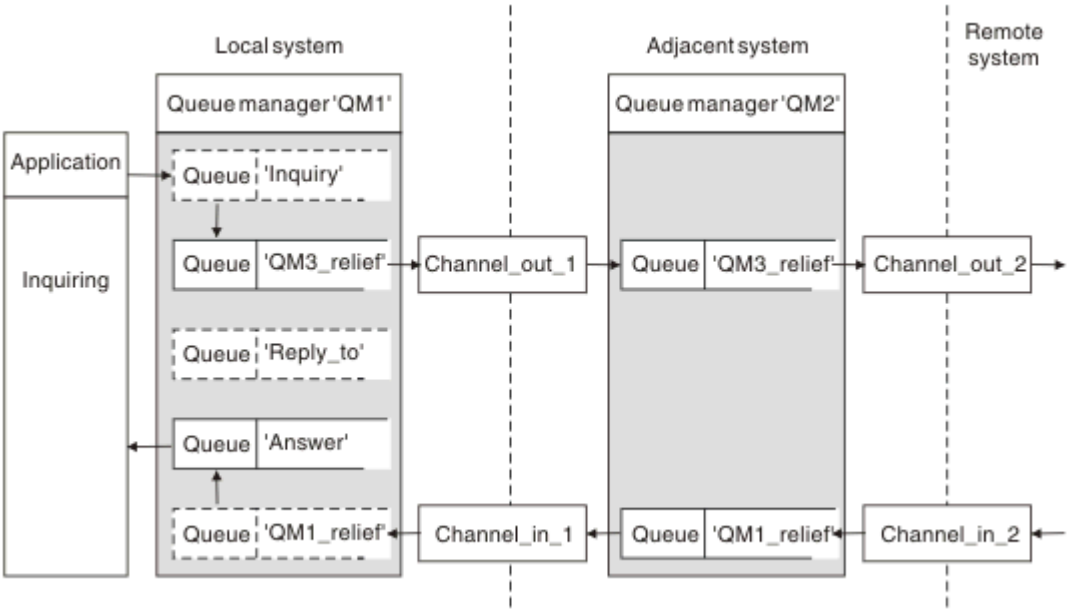
Kuyruk adı çözümlemesi

Bir uygulama bir iletiye yanıt verince, yanıtlanacak kuyruğun adını bulmak için aldığı iletinin *ileti tanımlayıcısı* ' ndeki verileri kullanır. Gönderme uygulaması, yanıtların nereye gönderileceğini ve bu bilgilerin iletilerine nasıl bağlanacağı belirtilir. Bu kavramın, uygulama tasarımınızın bir parçası olarak koordine edilmesi gerekir.

Kuyruk adı çözümlemesi, ileti bir kuyruğa konmadan önce uygulamanızın gönderme bitişindeki yerini alır. Bu nedenle kuyruk adı çözümlemesi, iletinin gönderilmekte olduğu uzak uygulamayla etkileşimden önce oluşur. Bu, bir kuyruğun açılmaması durumunda, ad çözümlemesinde yer alan tek durumdur.

Kuyruk yöneticisi diğer adı kullanılarak kuyruk adı çözümlemesi

Normalde bir uygulama bir yanıt kuyruğu belirler ve yanıtı kuyruk yöneticisi adının boş bırakılacağı şekilde bırakır. Kuyruğa alma sırasında kuyruk yöneticisi kendi adını tamamlar. This method works well except when you want an alternative channel to be used for replies, for example, a channel that uses transmission queue `QM1_relief` instead of the default return channel which uses transmission queue `QM1`. Bu durumda, iletim kuyruğu üstbilgilerinde belirtilen kuyruk yöneticisi adları "real" kuyruk yöneticisi adlarıyla eşleşmiyor, ancak kuyruk yöneticisi diğer ad tanımlamaları kullanılarak yeniden belirtildi. Yanıtları, alternatif rotaları boyunca geri döndürmek için, yanıt kuyruğu verilerinin yanı sıra yanıt kuyruğu diğer ad tanımlamalarını da kullanarak, yanıt kuyruğu verileri eşlenmesi gerekir.



Şekil 16. Yanıtlama-yanıt konumunu değiştirmek için kullanılan diğer kuyruk diğer adı

Şekil 16 sayfa 52örneğinde:

1. Uygulama, MQPUT çağrısını kullanarak bir ileti koyar ve ileti tanımlayıcısında aşağıdaki bilgileri belirtir:

   ```
   ReplyToQ='Reply_to'
   ReplyToQMgr=' '
   ```

   Yanıt kuyruğu diğer adının kullanılabilmesi içinReplyToQMgr boş bırakılmalıdır.
2. You create a reply-to queue alias definition called `Reply_to`, which contains the name `Answer`, and the queue manager name `QM1_relief`.

   ```
   DEFINE QREMOTE ('Reply_to') RNAME ('Answer')
   RQMNAME ('QM1_relief')
   ```

3. İletiler, `ReplyToQ='Answer'` ve `ReplyToQMgr='QM1_relief''`i gösteren bir ileti tanımlayıcısıyla gönderilir.
4. The application specification must include the information that replies are to be found in queue `Answer` rather than `Reply_to`.

Yanıtlara hazırlık yapmak için, koşut dönüş kanalını yaratmanız gerekir:

- QM2adresinde, `QM1_relief`adlı iletim kuyruğu

```
DEFINE QLOCAL ('QM1_relief') USAGE(XMITQ)
```

- QM1konumunda, kuyruk yöneticisi diğer adı `QM1_relief`

```
DEFINE QREMOTE ('QM1_relief') RNAME() RQMNAME(QM1)
```

Bu kuyruk yöneticisi diğer adı koşut dönüş kanallarının zincirini sonlandırır ve QM1için iletileri yakalar.

Bunu ileride bir zamanda yapmak isteyebileceğinizi düşünüyorsanız, uygulamaların başlangıç için diğer ad adını kullandığından emin olun. Bunun için, bu, yanıtlama kuyruğunda olağan bir kuyruk diğer adıdır; ancak daha sonra kuyruk yöneticisi diğer adı olarak değiştirilebilir.

Yanıtın gönderileceği kuyruk adı

Adlandırma yanıtlama kuyruklarıyla ilgilenmeniz gerekir. Bir uygulamanın iletiye yanıt kuyruğu adını koymasının nedeni, yanıtlarının gönderileceği kuyruğu belirleyebilmesini sağlar. Bu adı taşıyan bir yanıt kuyruğu diğer adı tanımlaması yarattığınızda, gerçek yanıt kuyruğunuz (yani, yerel bir kuyruk tanımlaması) aynı adı taşıyan bir kuyruğa sahip olamazsınız. Bu nedenle, yanıtlama kuyruğu diğer adı tanımlamasının yanı sıra kuyruk yöneticisi adının yanı sıra yeni bir kuyruk adı da içermelidir. Uygulama belirtimi, yanıtlarının bu diğer kuyrukta bulduğu bilgileri içermelidir.

Şimdi uygulamalar, özgün iletiyi koydukları sırada, yanıt kuyruğu olarak adlandırdıkları kuyruktan farklı bir kuyruktan iletileri almak zorunda.

Küme bileşenleri

Kümeler, kuyruk yöneticilerinden, küme havuzlarından, küme kanallarından ve küme kuyruklarından oluşur.

Küme bileşenlerinin her biriyle ilgili bilgi için aşağıdaki alt başlıklara bakın:

İlgili kavramlar

Kümeleme ve dağıtılmış kuyruklama karşılaştırması

İlgili görevler

Kuyruk yöneticisi kümesinin yapılandırılması

Yeni bir küme ayarlanıyor

Küme havuzu

Havuz, bir kümenin üyesi olan kuyruk yöneticilerine ilişkin bilgi toplamlıdır.

Havuz bilgileri, kuyruk yöneticisi adlarını, yerlerini, anasistemleri ve diğer bilgileri kuyrukları kuyruklarını içeren yerleri içerir. Bilgiler, `SYSTEM.CLUSTER.REPOSITORY.QUEUE`adlı bir kuyrukta bulunan iletiler biçiminde saklanır. Bu kuyruk, varsayılan nesnelerden biridir. Multi Multiplatformsüzerinde, bir IBM MQ kuyruk yöneticisi oluşturduğunuzda tanımlıdır. z/OS IBM MQ for z/OSüzerinde, kuyruk yöneticisi uyarlamasının bir parçası olarak tanımlıdır.

Tüm havuz ve kısmi havuz

Tipik olarak, bir kümedeki iki kuyruk yöneticisi tam havuzu tutar. Kalan kuyruk yöneticileri bir kısmi havuzu tutar.

Kümedeki her kuyruk yöneticisine ilişkin eksiksiz bir bilgi kümesini barındıran bir kuyruk yöneticisi tam bir havuza sahiptir. Kümedeki diğer kuyruk yöneticileri, tam havuzlardaki bilgilerin bir alt kümesini içeren kısmi havuzlara sahiptir.

Kısmi havuz, yalnızca kuyruk yöneticisinin ileti alışverişi için gereken kuyruk yöneticilerine ilişkin bilgileri içerir. Kuyruk yöneticileri, güncellemeleri gereken bilgilere göre ister; bu nedenle, değişiklik olursa, tüm havuz kuyruk yöneticisi yeni bilgileri gönderir. Çoğu zaman, kısmi bir havuz, bir kuyruk yöneticisinin küme içinde gerçekleştirmesi gereken tüm bilgileri içerir. Bir kuyruk yöneticisinin bazı ek bilgilere gereksinim duyması, tam havuzun sorgularını yapar ve kısmi havuzunu günceller. Kuyruk yöneticileri, havuzlara ilişkin güncellemeleri istemek ve bu güncellemeleri almak için `SYSTEM.CLUSTER.COMMAND.QUEUE` kuyruğunu kullanır.

Bir kümenin üyesi olan kuyruk yöneticilerini geçirirken , kısmi havuzlardan önce tüm havuzları geçirin. Bunun nedeni, daha eski bir havuzun daha yeni bir yayın düzeyinde tanıtılan daha yeni öznitelikleri saklayamayacağı içindir. Onlara tahammül ediyor, ama onları depolamıyor.

Küme kuyruk yöneticisi

Küme kuyruk yöneticisi, küme üyesi olan bir kuyruk yöneticidir.

Kuyruk yöneticisi birden çok kümeden bir üye olabilir. Her küme kuyruk yöneticisinin, üyesi olduğu tüm kümelerde benzersiz bir adı olmalıdır.

Küme kuyruk yöneticisi, kümedeki diğer kuyruk yöneticilerine duyurusu yapan kuyrukları barınebilir. Ancak, bunu yapmak zorunda değil. Bunun yerine, kümenin başka bir yerinde barındırılan kuyruklara ileti besleyebilir ve yalnızca belirtik olarak yönlendirilen yanıtları alabilir.

z/OS IBM MQ for z/OS' ta, bir küme kuyruk yöneticisi bir kuyruk paylaşım grubunun üyesi olabilir. Bu durumda, kuyruk tanımlamalarını aynı kuyruk paylaşım grubundaki diğer kuyruk yöneticileriyle paylaşır.

Küme kuyruğu yöneticileri otonom. Onlar, tanımladıkları kuyruklar ve kanallar üzerinde tam kontrole sahiptir. Tanımları, diğer kuyruk yöneticileri tarafından (aynı kuyruk paylaşım grubundaki kuyruk yöneticileri dışında) değiştiremezler. Havuz kuyruğu yöneticileri, kümedeki diğer kuyruk yöneticilerindeki tanımları denetlemez. Bunlar, gerektiğinde kullanılmak üzere tüm tanımların eksiksiz bir kümesini tutar. Küme, kuyruk yöneticilerinden oluşan bir birleşiktir.

Bir küme kuyruk yöneticisininde bir tanımlama yarattıktan ya da değiştirdikten sonra, bilgiler tam havuz kuyruk yöneticisine gönderilir. Kümedeki diğer havuzlar daha sonra güncelleştirilir.

Tam Havuz Kuyruğu Yöneticisi

Tam havuz kuyruk yöneticisi, kümenin kaynaklarının tam gösterimini tutan bir küme kuyruk yöneticidir. Kullanılabilirliği sağlamak için, her kümede iki ya da daha çok tam havuz kuyruğu yöneticisi ayarlayın. Tam havuz kuyruğu yöneticileri, kümedeki diğer kuyruk yöneticileri tarafından gönderilen bilgileri alır ve havuzlarını günceller. Her ikisinin de küme hakkında yeni bilgiler ile güncel tutulduğundan emin olmak için birbirlerine mesaj gönderirler.

Kuyruk yöneticileri ve havuzlar

Her kümede en az bir (tercihen iki) kuyruk yöneticisi, bir kümedeki kuyruk yöneticilerine, kuyruklara ve kanallara ilişkin bilgileri tam olarak bulunduran bir yöneticiye sahiptir. Bu havuzlar, bilgi güncellemeleri için kümedeki diğer kuyruk yöneticilerinden gelen istekleri de içerir.

Diğer kuyruk yöneticilerinin her biri, iletişim kurmaları gereken kuyruk ve kuyruk yöneticilerinin altkümesine ilişkin bilgileri içeren kısmi bir havuz tutmasını sağlar. Kuyruk yöneticileri, başka bir kuyruğa

ya da kuyruk yöneticisine ilk erişmeleri gerektiğinde sorgular yaparak, kısmi havuzlarını oluşturur. Kuyruk ya da kuyruk yöneticiyle ilgili herhangi bir yeni bilgi bildirileceğini bildirirler.

Each queue manager stores its repository information in messages on a queue called SYSTEM.CLUSTER.REPOSITORY.QUEUE. Kuyruk yöneticileri, SYSTEM.CLUSTER.COMMAND.QUEUEadlı bir kuyrukta bulunan iletilerde havuz bilgilerini değiş tokuş eder.

Bir kümeye katılan her kuyruk yöneticisi, havuzlardan birine bir küme gönderen CLUSSDR(CLUSSDR), kanal tanımlar. Kümedeki diğer kuyruk yöneticilerinin tam olarak hangi havuzlarda tutulmayı öğrendiğini hemen öğrenir. Bundan sonra kuyruk yöneticisi, havuzlardan herhangi birinden bilgi isteyebilirler. Kuyruk yöneticisi, seçilen havuza bilgi gönderdiğinde, bilgileri başka bir havuza (varsa) gönderir.

Tam bir havuz, kuyruk yöneticisi tarafından barındırıldığında, buna bağlı kuyruk yöneticilerinden birinden yeni bilgiler aldığında güncelleştirilir. Yeni bilgiler, bir havuz kuyruğu yöneticisinin hizmet dışı olması durumunda gecikme riskini azaltmak için başka bir havuza de gönderilir. Tüm bilgiler iki kez gönderildiğinden, havuzların yinelemeleri atması gerekir. Her bilgi öğesi, havuzların yinelemeleri tanımlamak için kullandığı bir sıra numarası taşır. Tüm havuzlar, ileti alışverişi yaparak adımlarda birbirlerine adım atmakta tutulur.

Küme kuyrukları

Küme kuyruğu, bir küme kuyruk yöneticisi tarafından barındırılan ve kümedeki diğer kuyruk yöneticilerinin kullanımına sunulan bir kuyruktur.

Kümedeki diğer kuyruk yöneticilerine bir küme kuyruğu tanımlaması tanıtılır. Kümedeki diğer kuyruk yöneticileri, iletileri karşılık gelen uzak kuyruk tanımlamasına gerek kalmadan bir küme kuyruğuna yerleştirebilirler. Küme kuyruğu, küme adı kullanılarak birden çok kümede duyurulabilir.

Bir kuyruk duyurulduğunda, kümedeki herhangi bir kuyruk yöneticisi bu kuyruğa ileti yerleştirebilir. Bir ileti koymak için kuyruk yöneticisinin, kuyruğun bulunduğu tam havuzlardan bunu öğrenmesi gerekir. Daha sonra iletiye bazı yöneltme bilgileri ekler ve iletiyi bir küme iletim kuyruğuna yerleştirir.

z/OS Küme kuyruğu, IBM MQ for z/OS' daki bir kuyruk paylaşım grubunun üyeleri tarafından paylaşılan bir kuyruk olabilir.

İlgili görevler

Küme kuyruklarının tanımlanması

Paylaşılan kuyruklar ve küme kuyrukları arasındaki karşılaştırma

Bu bilgiler, paylaşılan kuyrukları ve küme kuyruklarını karşılaştırmanıza ve sisteminize daha uygun olmaya karar vermenize yardımcı olacak şekilde tasarlanmıştır.

Kanal başlatıcı maliyetleri

Küme kuyruklarında, iletiler kanallar tarafından gönderilir, bu nedenle uygulama maliyetlerine ek olarak kanal başlatıcı maliyetlerine de olanak sağlar. Kanallar ileti alıp yerleştirdiği için ağlarda maliyetler var. Bu maliyetler, paylaşılan kuyruklarla yoktur; bu nedenle, bir kuyruk paylaşım grubundaki kuyruk yöneticileri arasında iletiler taşınırken küme kuyruklarından daha az işlem gücü kullanır.

İletilerin kullanılabilirliği

Bir kuyruğa yerleştirilirken, küme kuyrukları, kuyruk yöneticinize bağlı etkin kanalları olan kuyruk yöneticilerinden birine ileti gönderir. Uzak kuyruk yöneticisininde, iletileri işlemek için kullanılan uygulamalar çalışmıyorsa, iletiler işlenmez ve uygulamalar başlatılıncaya kadar beklemez. Benzer bir şekilde, bir kuyruk yöneticisi kapatılırsa, kuyruk yöneticisine ilişkin iletiler kuyruk yöneticisi yeniden başlatılıncaya kadar kullanılamaz. Bu eşgörünümler, paylaşılan kuyruklar kullanılmasından daha düşük ileti kullanılabilirliğini gösterir.

Paylaşılan kuyruklar kullanılırken, kuyruk paylaşım grubundaki herhangi bir uygulama, gönderilen iletileri alabilir. Kuyruk paylaşım grubunda bir kuyruk yöneticisini kapadıysanız, iletiler diğer kuyruk yöneticilerine kullanılabilir ve küme kuyrukları kullanılmasından daha yüksek ileti kullanılabilirliği sağlar.

Kapasite

Bir bağlaşım olanağı, bir diskten daha pahalıdır; bu nedenle, bir yerel kuyrukta 1.000.000 ileti saklama maliyeti, aynı sayıda iletiyi depolamaya yetecek kapasiteye sahip bir bağlaşım tesisine sahip olmaktan daha düşük bir maliyettir.

Diğer kuyruk yöneticilerine gönderme

Paylaşılan kuyruk iletileri yalnızca bir kuyruk paylaşım grubu içinde kullanılabilir. Kuyruk paylaşım grubu dışında bir kuyruk yöneticisi kullanmak istiyorsanız, kanalları kullanmanız gerekir. Birden çok uzak dağıtılmış kuyruk yöneticisi arasında iş yükü dengelemesi yapmak için kümeleme işlemini kullanabilirsiniz.

İş yükü dengeleme

Hangi kanallara ve kuyruk yöneticilerine gönderilen iletilerin orantısını alabilecekleri ağırlık vermek için kümelemeyi kullanabilirsiniz. Örneğin, iletilerin %60 'ı bir kuyruk yöneticisine ve iletilerin %40 'ını başka bir kuyruk yöneticisine gönderebilirsiniz. Bu yönetim ortamı, uzak kuyruk yöneticisinin işi işleyebilmesi için gereken yeteneye bağlı değildir. İlk kuyruk yöneticisine sahip sistem aşırı yüklenmiş olabilir ve ikinci kuyruk yöneticisine sahip sistem boşta olabilir, ancak iletilerin çoğu ilk kuyruk yöneticisine devam eder.

Paylaşılan kuyruklar ile iki CICS sistemi ileti alabilir. Bir sistem aşırı yüklenirse, diğer sistem iş yükünün çoğunu devralır.

Küme kanalları

Her tam havuzda, bir küme alıcı kanalı ve kümedeki diğer tüm havuzlara bağlanmak için bir küme gönderici kanalı kümesi el ile tanımlandınız. Kısmi bir havuz eklediğinizde, bir küme alıcı kanalı ve tam havuzlardan birine bağlanan tek bir kümeli gönderici kanalı el ile tanımlırsınız. Daha fazla küme gönderici kanalları gerektiğinde küme tarafından otomatik olarak tanımlanır. Otomatik olarak tanımlanan küme gönderici kanalları, giriş kuyruğu yöneticilenindeki ilgili küme alıcı kanalı tanımlamasından özniteliklerini alır.

Küme-alıcı kanalı: CLUSTRVR

CLUSTRVR kanal tanımlaması, bir küme kuyruk yöneticisinin kümedeki diğer kuyruk yöneticilerinden ileti alabileceği bir kanalın sonunu tanımlar.

Her küme kuyruk yöneticisi için en az bir CLUSRCVR kanalı tanımlamanız gerekir. Kuyruk yöneticisi, CLUSTRCVR kanalını tanımlayarak, ileti almak için uygun olan diğer küme kuyruğu yöneticilerini gösterir.

CLUSTRVR kanal tanımlaması, diğer kuyruk yöneticilerinin ilgili küme gönderici kanal tanımlamalarını otomatik olarak tanımlamalarına da olanak sağlar. Bu makalenin “Otomatik olarak tanımlanan küme-gönderen kanalları” sayfa 57 bölümüne bakın.

Küme-gönderici kanalı: CLUSSDR

Kümedeki her tam havuz kuyruk yöneticisinden her tam havuz kuyruk yöneticisinden bir CLUSSDR kanalını el ile tanımlayabilirsiniz. Tüm havuzlar yalnızca bu kanallarda akış yaparak değiş tokuş edilen tüm güncellemelere. Bu kanalları el ile tanımlayarak, tam havuzların ağını açık bir şekilde kontrol edin.

Bir kümeye kısmi bir havuz kuyruk yöneticisi eklediğinizde, tüm havuzlardan birine bağlanmak için tek bir CLUSTSDR kanalı tanımlanınız. Bu, ilk iletişim sorumlusu yapıldıktan sonra, kuyruk yöneticinizin daha fazla küme kuyruk yöneticisi nesneleri (CLUSSDR kanalları da içinde olmak üzere) gerektiği şekilde otomatik olarak tanımlandığından, seçtiğiniz havuzun tam olarak değişmesine neden olur. Bu, kuyruk yöneticinizin küme bilgilerini herhangi bir tam havuza göndermesini ve kümedeki kuyruk yöneticilerine ileti göndermesini sağlar.

Bu makalenin geçerli bölümünde açıklandığı gibi, otomatik olarak tanımlanmış gönderen kanalları, küme alıcı kanalının yapılandırmasına dayanır. Bu nedenle, küme kanallarında ayarladığınız herhangi bir kanal

özelliği, eşleşen CLUSSDR ve küme-alıcı kanallarında ya da yalnızca küme alıcılı kanallarda ayarlanmış olmalıdır.

Daha önce açıklanan nedenlerle, yalnızca CLUSSDR kanallarını el ile tanımlamanız gerekir. Yani, kısmi bir havuzu, tam bir havuza ilk kez bağlamak ya da iki tam havuzu birbirine bağlamak için. Kısmi bir havuza ya da kümede olmayan bir kuyruk yöneticisine bağlanan bir CLUSTSDR kanalının el ile yapılandırılması, AMQ9427 ve AMQ9428 gibi hata iletilerinin yayınlanmasına neden olur. Bu durum bazen geçici bir durum olarak önlenemez olsa da, örneğin, tam bir havuzun konumunu değiştirirken, el ile tanımlamanın en kısa zamanda silinmesi gerekir.

Otomatik olarak tanımlanan küme-gönderen kanalları

Tipik olarak, bir kümeye kısmi bir havuz kuyruğu yöneticisi eklediğinizde, kuyruk yöneticisince elle yalnızca iki küme kanalı tanımlayabilir:

- Bir kümeli gönderici (CLUSSDR) Küme için tam havuz kuyruk yöneticisine kanal.
- Bir küme alıcısı (CLUSTRVR) kanal.

Tanımladığınız CLUSSDR kanalı, kuyruk yöneticisinin küme ile ilk kez iletişim kurmasına olanak sağlar. İlk iletişim sorumlusu yapıldıktan sonra, gerektiğinde küme tarafından otomatik olarak CLUSSDR kanalları tanımlanır.

Otomatik tanımlı bir CLUSSDR kanalı, özniteliklerini alan kuyruk yöneticisiyle ilgili CLUSTUSRVR kanal tanımlamasından alır. El ile tanımlanmış bir CLUSSDR kanalı olsa da, otomatik olarak tanımlanan CLUSSDR kanalından öznitelikler kullanılır. Örneğin, **CONNAME** parametresindeki bir kapı numarası belirtmeden bir CLUSRCVR kanalı tanımladığınızı ve bir kapı numarası belirten bir CLUSSDR kanalını elle tanımladığınızı varsayalım. Otomatik tanımlanan CLUSSDR kanalı el ile tanımlanan bir kanal yerine geçtiğinde, kapı numarası (CLUSTRVR kanalından alınır) boş olur. Varsayılan kapı numarası kullanılır ve kanal başarısız olur.

El ile tanımlanmış bir CLUSSDR kanalı ile karşılık gelen CLUSTRCVR kanal tanımlaması arasındaki yapılandırma farklarının olduğu yerlerde, bazı farklılıklar hemen yürürlüğe girmektedir (örneğin, iş yükü dengeleme parametreleri) ve bazıları yalnızca kanal yeniden başlatıldığında yürürlüğe girmektedir (örneğin, TLS yapılandırması).

Karışıklığı önlemek için, mümkün olduğu kadar aşağıdaki yönergeleri izleyin:

- Tüm havuzları göstermek için yalnızca CLUSSDR kanallarını el ile tanımlayın.
- El ile tanımladığınız CLUSSDR kanallarını el ile tanımladığınızda, bunları, alıcı kuyruk yöneticilikindeki karşılık gelen CLUSTRVR kanal tanımlamasıyla aynı şekilde eşleşecek şekilde yapılandırın.

Ayrıca bkz. Otomatik tanımlı kanallarla çalışma.

İlgili kavramlar

Otomatik tanımlı kanallarla çalışma

Küme iletim kuyruklarıyla ve kümeyle gönderici kanallarla çalışma

İlgili görevler

Yeni bir küme ayarlanıyor

Küme için kuyruk yöneticisi eklenmesi

Küme konuları

Küme konuları, **cluster** özniteliği tanımlanmış yönetimle ilgili konulardır. Küme konularına ilişkin bilgiler bir kümenin tüm üyelerine gönderilir ve birden çok kuyruk yöneticisiyle ilgili bir konu alanının bölümlerini yaratmak için yerel konularla birleştirilir. Bu, bir kuyruk yöneticisinde bir konuda yayınlanan iletilerin kümedeki diğer kuyruk yöneticilerinin aboneliklerine teslim edilmesini sağlar.

Bir kuyruk yöneticisinde küme konusu tanımladığınızda, küme konusu tanımlaması tam havuz kuyruğu yöneticilerine gönderilir. Daha sonra tüm havuzlar küme konusu tanımlamasını küme içindeki tüm kuyruk yöneticilerine geçirerek, aynı küme konusunu kümedeki herhangi bir kuyruk yöneticisindeki yayınlayıcılar ve aboneler için kullanılabilir hale getirir. Küme konusu yarattığınız kuyruk yöneticisi,

küme konusu anasistemi olarak bilinir. Küme konusu kümedeki herhangi bir kuyruk yöneticisi tarafından kullanılabilir, ancak bir küme konusunda yapılan değişiklikler, ilgili konunun tanımlandığı (anasistem) kuyruk yöneticisinde yapılmalıdır; bu noktada değişiklik, tam havuzlar aracılığıyla kümenin tüm üyelerine yayılır.

Küme başlıklarını *doğrudan yöneltme* ya da *konu anasistem yönlendirmesi*kullanacak şekilde yapılandırma hakkında ve kümelenmiş konu edinme ve genel arama abonelikleri hakkında bilgi almak için Küme konularını tanımlamabaşlıklı konuya bakın.

Küme konularını görüntülemek için kullanılacak komutlara ilişkin bilgi için ilgili bilgilere bakın.

İlgili kavramlar

Yönetimle ilgili konularla çalışma

Aboneliklerle çalışma

İlgili başvurular

KONUYU GÖRÜNTÜLE

TANıTıM

GöRüNTüLE

Varsayılan küme nesneleri

Multi Çoklu Platformlar üzerinde, varsayılan küme nesneleri, bir kuyruk yöneticisi tanımladığınızda otomatik olarak yaratılan varsayılan nesneler kümesine dahil edilir. z/OS z/OSüzerinde, varsayılan küme nesnesi tanımlamaları özelleştirme örneklerinde bulunabilir.

Not: MQSC ya da PCF komutlarını çalıştırarak, varsayılan kanal tanımlamalarını diğer kanal tanımlamalarıyla aynı şekilde değiştirebilirsiniz. SYSTEM.CLUSTER.HISTORY.QUEUEdışında varsayılan kuyruk tanımlamalarını değiştirmeyin.

SYSTEM.CLUSTER.COMMAND.QUEUE
Bir kümedeki her kuyruk yöneticisinin, iletileri tam havuza aktarmak için kullanılan SYSTEM.CLUSTER.COMMAND.QUEUE adlı bir yerel kuyruğu vardır. İleti, kuyruk yöneticiyle ilgili yeni ya da değiştirilmiş bilgileri ya da diğer kuyruk yöneticilerine ilişkin bilgi isteklerini içerir. SYSTEM.CLUSTER.COMMAND.QUEUE olağan durumda boştur.

SYSTEM.CLUSTER.HISTORY.QUEUE
Bir kümedeki her kuyruk yöneticisinin SYSTEM.CLUSTER.HISTORY.QUEUEadlı bir yerel kuyruğu vardır. SYSTEM.CLUSTER.HISTORY.QUEUE , hizmet amacıyla küme durumu bilgilerinin geçmişini saklamak için kullanılır.

Varsayılan nesne ayarlarında SYSTEM.CLUSTER.HISTORY.QUEUE , PUT (ENABLED) olarak ayarlıdır. Geçmiş derlemini gizlemek için ayarı PUT (DISABLED) olarak değiştirin.

SYSTEM.CLUSTER.REPOSITORY.QUEUE
Bir kümedeki her kuyruk yöneticisinin SYSTEM.CLUSTER.REPOSITORY.QUEUEadlı bir yerel kuyruğu vardır. Bu kuyruk, tüm havuz bilgilerini saklamak için kullanılır. Bu kuyruk olağan durumda boş değil.

SYSTEM.CLUSTER.TRANSMIT.QUEUE
Her kuyruk yöneticisinin, SYSTEM.CLUSTER.TRANSMIT.QUEUEadlı yerel bir kuyruk için bir tanımlaması vardır. SYSTEM.CLUSTER.TRANSMIT.QUEUE , tüm iletiler için varsayılan iletim kuyruğıdır ve kümeler içinde olan tüm kuyruk yöneticilerine ve kuyruk yöneticilerine iletilir. You can change the default transmission queue for each cluster-sender channel to SYSTEM.CLUSTER.TRANSMIT. *ChannelName*, by changing the queue manager attribute DEFCLXQ . SYSTEM.CLUSTER.TRANSMIT.QUEUEöğesini silemezsiniz. Ayrıca, kullanılan varsayılan iletim kuyruğunun SYSTEM.CLUSTER.TRANSMIT.QUEUE ya da SYSTEM.CLUSTER.TRANSMIT. *ChannelName*olup olmadığını denetleyerek yetki denetimini de tanımlamak için kullanılır.

SYSTEM.DEF.CLUSRCVR
Her bir kümenin SYSTEM.DEF.CLUSRCVRadlı varsayılan CLUSRCVR kanal tanımlaması vardır. SYSTEM.DEF.CLUSRCVR , kümedeki bir kuyruk yöneticisinde bir küme alıcı kanalı yarattığınızda belirtmediğiniz öznitelikler için varsayılan değerleri sağlamak üzere kullanılır.

SYSTEM.DEF.CLUSSDR
Her bir kümenin SYSTEM.DEF.CLUSSDRadlı varsayılan CLUSSDR kanal tanımlaması vardır. SYSTEM.DEF.CLUSSDR , kümedeki bir kuyruk yöneticisinde bir küme gönderici kanalı yarattığınızda belirtmediğiniz öznitelikler için varsayılan değerleri sağlamak üzere kullanılır.

İlgili kavramlar

Varsayılan küme nesneleriyle çalışma

Yayınlama/abone olma ileti alışverişi

Yayınlama/abone olma ileti sistemi, bilgi sağlayıcısını bu bilgilerin tüketicilerinden çözmenize olanak sağlar. Gönderme uygulaması ve alma uygulamasının, gönderilecek ve alınacak bilgiler için birbiriyle ilgili herhangi bir bilgi sahibi olması gerekmez.

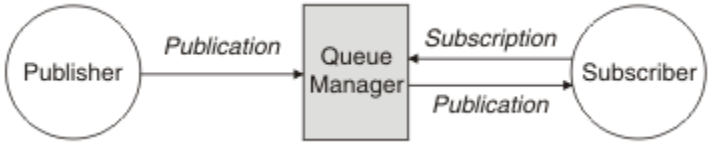
Bir noktadan noktaya IBM MQ uygulaması başka bir uygulamaya ileti göndermeden önce, bu uygulama hakkında bir bilgi sahibi olması gerekir. Örneğin, bilgilerin gönderileceği kuyruğun adını bilmesi ve kuyruk yöneticisi adını da belirtmesi gerekir.

IBM MQ yayınlama/abone olma, uygulamanızın hedef uygulama hakkında herhangi bir bilgi sahibi olması gereksinmesini kaldırır. Tüm gönderme uygulamasının yapması gereken şu:

- Uygulamanın istediği bilgileri içeren bir IBM MQ iletisi*yerleştirin* .
- İletiyi, bilgilerin konusunu belirten bir konuya atayın.
- Bu bilgilerin dağıtımını IBM MQ işlemesine izin verin.

Benzer şekilde, hedef uygulamanın aldığı bilgilerin kaynağı hakkında bilgi sahibi olmak zorunda değildir.

Aşağıdaki şekil, en basit yayınlama/abone olma sistemini göstermektedir. Bir yayınlayıcı, bir kuyruk yöneticisi ve bir abone var. Bir abonelik, bir kuyruk yöneticisininde abone tarafından yapılır, yayıncıdan kuyruk yöneticisine bir yayın gönderilir ve sonra yayınlama, kuyruk yöneticisi tarafından aboneye iletilir.



Şekil 17. Basit yayınlama/abone olma yapılandırması

Tipik bir yayınlama/abone olma sisteminde birden çok yayınlayıcı ve birden çok farklı konu üzerinde birden çok abone var ve genellikle birden çok kuyruk yöneticisi var. Bir uygulama hem yayınlayıcı hem de bir abone olabilir.

Yayınlama/abone olma ileti alışverişi ve noktadan noktaya iletişim arasındaki diğer bir önemli fark, bir noktadan noktaya iletişim kuyruğuna gönderilen bir iletinin yalnızca tek bir tüketen uygulama tarafından işlenmesini sağlar. Birden çok abonenin ilgilendiği bir yayınlama/abone olma konusuna yayınlanan ileti, ilgilenen her abone tarafından işlenir.

Bileşenleri yayınla/abone ol

Yayınlama/abone olma, abonelerin yayıncılardan ileti biçiminde, bilgi alabilecekleri mekanizmadır. Yayıncılar ve aboneler arasındaki etkileşimler, standart IBM MQ olanaklarını kullanarak kuyruk yöneticilerine göre denetlenir.

Tipik bir yayınlama/abone olma sisteminde birden çok yayınlayıcı ve birden çok farklı konu üzerinde birden çok abone var ve genellikle birden çok kuyruk yöneticisi var. Bir uygulama hem yayınlayıcı hem de bir abone olabilir.

Bilgi sağlayıcıya *yayınlayıcı*adı verilir. Yayıncılar, bu bilgilerle ilgilenen uygulamalar hakkında herhangi bir bilgi sahibi olmamaya gerek kalmadan bir konu hakkında bilgi sağlar. Yayıncılar bu bilgileri, bu iletilerin konusunu yayınlamak ve tanımlamak istedikleri *yayınlar* adı verilen iletiler biçiminde oluşturur.

Bilgilerin tüketicisi bir *abone*olarak adlandırılır. Aboneler, abonenin ilgilendiği konuyu açıklayan *abonelikler* oluşturur. Bu nedenle, abonelik, aboneye hangi yayınların iletileceğini belirler. Aboneler birden çok aboneliği gerçekleştirebilirler ve birçok farklı yayıncıdan bilgi alabilirler.

Yayınlanan bilgiler bir IBM MQ iletisinde gönderilir ve bilgilerin konusu *konusu*ile tanımlanır. Yayıncı, bilgileri yayınlarken konuyu belirtir ve abone, yayınları almak istediği konuları belirtir. Aboneye, yalnızca abone olduğu konular hakkında bilgi gönderilir.

Bu, her iletiye noktadan noktaya ileti sisteminde gerekli olduğu şekilde, her iletiye belirli bir hedef dahil etme gereksinmesini kaldırarak, sağlayıcıların ve tüketicilerinin yayınlama/abone olma mesajlarında ayrılmasına izin veren konuların varoluşunu sağlar.

Yayıncılar ve aboneler arasındaki etkileşimler bir kuyruk yöneticisi tarafından denetlenir. Kuyruk yöneticisi yayıncılardan ileti alır ve abonelerden abonelikler (bir dizi konu aralığıyla). Kuyruk yöneticisinin işi, yayınlanan iletileri, iletilerin konusuna ilgi kaydetmiş olan abonelere yöneltmek.

İletileri dağıtmak için standart IBM MQ olanakları kullanılır, böylece uygulamalarınız var olan IBM MQ uygulamalarının kullanımına sunulan tüm özellikleri kullanabilir. Bu, yalnızca bir kez garantili teslim almak için kalıcı iletileri kullanabileceğiniz ve iletilerin, yalnızca yayınlayıcı tarafından kesinleştirildikleri durumlarda aboneye teslim edildiğinden emin olmak için iletilerin işlem biriminin bir parçası olabileceğinin anlamına gelir.

Yayıncılar ve yayınlar

IBM MQ yayınlama/abone olma, belirli bir konu hakkında bilgi veren bir uygulamadır. Bu uygulama, bir yayınlama adı verilen standart bir IBM MQ iletisi biçiminde bir kuyruk yöneticisi tarafından kullanılabilir. Bir yayınlayıcı, birden fazla konu hakkında bilgi yayınlayabilir.

Yayıncılar, daha önce açılmış bir konuya ileti koymak için MQPUT komutunu kullanır, bu ileti bir yayındır. Daha sonra, yerel kuyruk yöneticisi yayını, yayın konusu olan abonelere sahip olan abonelere yönlendirir. Yayınlanan bir ileti birden çok abone tarafından tüketilebilir.

Bir kuyruk yöneticisi, yayınları uygun aboneliklere sahip tüm yerel abonelere dağıtmanın yanı sıra, bu yayını doğrudan ya da abone olan bir kuyruk yöneticisi ağı aracılığıyla bu konuya bağlı olan diğer kuyruk yöneticilerine de dağıtabilir.

Bir IBM MQ yayınlama/abone olma ağlarında, bir yayınlama uygulaması da abone olabilir.

Syncpoint altındaki yayınlar

Yayıncılar, bir iş birimindeki abonelere teslim edilen tüm iletileri içermek için, MQPUT ya da MQPUT1 çağrıları syncpoint 'te yayınlayabilir. MQPMO_RETAIN seçeneği ya da ALL ya da ALLDUR değerleri içeren NPMSGDLV ya da PMSGDLV konu teslim seçenekleri belirtilirse, kuyruk yöneticisi, yayınlayıcı MQPUT ya da MQPUT1 çağrısı kapsamında, syncpoint 'te iç MQPUT ya da MQPUT1 çağrılarını kullanır.

Durum ve olay bilgileri

Yayınlar, hisse senetlerinin geçerli fiyatı gibi devlet yayınları ya da söz konusu stoktaki bir ticaret gibi etkinlik yayınları kategorilendirilebilir.

Durum yayınları

Durum yayınları , hisse senedi fiyatı ya da bir futbol karşılaşmasında geçerli puan gibi bir şeyin geçerli durumuna ilişkin bilgileri içerir. Bir şey olduğunda (örneğin, hisse senedi fiyatı değişiklikleri ya da futbol puanı değişirse), önceki durum bilgileri artık gerekli değildir, çünkü bu bilgiler yeni bilgiler tarafından yerine konmaktadır.

Bir abone başlatıldığında durum bilgilerinin geçerli sürümünü almak ve durum her değiştiğinde yeni bilgiler göndermek isteyecektir.

Bir yayında durum bilgisi varsa, bu yayın genellikle alıkonan bir yayın olarak yayınlanır. Yeni bir abone tipik olarak yürürlükteki durum bilgilerinin hemen olmasını ister; abonenin bilgilerin yeniden yayınlanmasına

neden olan bir olayın beklenmesini beklemez. Abone MQSO_PUBLICATIONS_ON_REQUEST ya da MQSO_NEW_PATICATIONS_ONLY seçeneklerini kullanmadığı sürece abone, abone olduğunda otomatik olarak bir konunun alıkonduğu yayınını alır.

Olay yayınları

Olay yayınları , bir hisse senedi içindeki ticaret ya da belirli bir hedefin puanlaması gibi, ortaya çıkan her bir olaya ilişkin bilgileri içerir. Her olay, diğer olaylardan bağımsızdır.

Bir abone, olaylar hakkında bilgi almak isteyecektir.

Alıkonan yayınlar

Varsayılan olarak, ilgili tüm abonelere bir yayın gönderildikten sonra atılır. Ancak, bir yayıncı, bir yayının kopyasının, konuya ilgi duyan gelecekteki abonelere gönderilebilmesi için alıkonabileceğini belirtebilir.

Tüm ilgili abonelere gönderildikten sonra yayınların silinmesi, etkinlik bilgileri için uygundur, ancak her zaman durum bilgileri için uygun değildir. Bir iletiyi koruyarak, yeni abonelerin ilk durum bilgilerini almadan önce bilgilerin yeniden yayınlanabilmesi için beklemek zorunda kalmazlar. Örneğin, hisse senedi aboneliğine aboneliği olan bir abone, hisse senedi fiyatının değişmesini beklemeden (ve dolayısıyla yeniden yayınlanmak üzere) geçerli fiyatı doğrudan alır.

Kuyruk yöneticisi her konu için yalnızca bir yayını tutabilir; böylece, kuyruk yöneticisinde yeni bir alıkonan yayın geldiğinde, bir konuya ilişkin var olan tutulan yayın yayını silinir. Ancak, var olan yayının silinmesi, yeni alıkonan yayının gelmesiyle zamanuyumlu olarak gerçekleşmeyebilir. Bu nedenle, mümkün olan her yerde, herhangi bir konuyla ilgili olarak saklanan yayınlardan birden fazla yayınlayıcı gönderemez.

Aboneler, MQSO_NEW_YAYINICATIONS_ONLY abonelik seçeneğini kullanarak alıkonan yayınlar almak istemediklerini belirtebilir. Var olan aboneler, alıkonan yayınların kopyalarını çoğaltmak için kendilerine bilgi isteyebilir.

Durum bilgileri için bile, yayınları tutmak istemeyebileceğinizde bazı zamanlar vardır:

- Bir konuya ilişkin tüm abonelikler o konuda herhangi bir yayın yapılmadan önce yapılıyorsa ve bu konuda herhangi bir yayın yapılmadan önce, yeni aboneliklere izin vermezseniz ya da izin vermezseniz, bu yayınlar ilk yayınlandıklarında tüm aboneler kümesine teslim edildiğinden, yayınların alıkonmasına gerek yoktur.
- Yayınların her saniye gibi sık ortaya çıkması durumunda, yeni bir abone (ya da bir hatadan kurtarma bir abone), ilk aboneliklerinden hemen hemen sonra geçerli durumu alır; bu nedenle, bu yayınları alıkoymaya gerek yoktur.
- Yayınlar büyükse, her konu için alıkonan yayını saklamak için çok miktarda depolama alanına gereksinim duyabilirsiniz. Birden çok kuyruk yöneticisi ortamında, tutulan yayınlar, eşleşen bir aboneliğe sahip olan ağdaki tüm kuyruk yöneticileri tarafından saklanır.

Tutulan yayınların kullanımına karar verilirken, uygulamaların bir hatadan nasıl kurtarılacağını göz önünde bulundurun. Yayınlayıcı saklan yayınlarını kullanmıyorsa, abone uygulamasının yürürlükteki durumunu yerel olarak saklamaya gerek olabilir.

Bir yayının alıkonmasını sağlamak için, MQPMO_RETAIN put-message seçeneğini kullanın. Bu seçenek kullanılırsa ve yayın saklanamazsa, ileti yayınlanmaz ve çağrı MQRC_PUT_NOT_ALIMARM ile başarısız olur.

Bir ileti alıkonan bir yayınsa, bu ileti MQIsRetained ileti özelliği ile gösterilir. Bir iletinin devamlılığı, ilk olarak yayınlandığı sırada olduğu gibidir.

İlgili kavramlar

Yayınlama/abone olma kümelerinde tutulan yayınlarda dikkat edilmesi gereken noktalar

Syncpoint altındaki yayınlar

IBM MQ publish/subschede içinde, syncpoint yayıncılar tarafından ya da kuyruk yöneticisi tarafından dahili olarak kullanılabilir.

Yayıncılar, MQPMO_SYNCPOINT seçeneğiyle MQPUT/MQPUT1 çağrıları yayınlandığında syncpoint 'i kullanır. Abonelere teslim edilen tüm iletiler, work.The MAXUMSGS kuyruk yöneticisi özniteliğinde, kesinleştirilmemiş ileti sayısı üst sınırına kadar sayılır. Bu sınır, bu sınırı belirler. Sınıra ulaşılırsa, yayınlayıcı 2024 (07E8) (RC2024): MQRC_SYNCPOINT_LIMIT_UVARD neden kodlarına ulaşır.

Bir yayınlayıcı, MQPMO_RETAIN seçeneğiyle MQPMO_NO_SYNCPOINT işlevini ya da ALL ya da ALLDR değerleri içeren NPMSGDLV/PMSGDLV konu teslim seçeneklerini kullanarak MQPUT/MQPUT1 ' i çağırdığında, kuyruk yöneticisi iletilerin istenildiği gibi teslim edildiğini garanti etmek için iç eşitleme noktaları kullanır. Yayınlayıcı MQPUT/MQPUT1 çağrısı kapsamında sınıra ulaşılırsa, yayınlayıcı 2024 (07E8) (RC2024): MQRC_SYNCPOINT_LIMIT_UVARD neden kodunu alabilir.

Aboneler ve abonelikler

In IBM MQ publish/subscribe, a subscriber is an application that requests information about a specific topic from a queue manager in a publish/subscribe network. Bir abone, aynı ya da farklı konular hakkında, birden çok yayınlayıcıdan ileti alabilir.

Abonelikler, bir MQSC komutu ya da uygulamalar kullanılarak el ile yaratılabilir. Bu abonelikler yerel kuyruk yöneticisine verilir ve abonenin almak istediği yayınlarla ilgili bilgileri içerir:

- Abonenin ilgilendiği konu; genel arama karakterlerinin kullanılması durumunda birden çok konuya çözümlenebilir.
- Yayınlanan iletilere uygulanacak isteğe bağlı bir seçim dizgisi.
- Seçilen yayınların yerleştirileceği ve isteğe bağlı CorrelIdolan bir kuyruğa ilişkin tanıtıcı (*abone kuyruğu* olarak bilinir).

Yerel kuyruk yöneticisi, abonelik bilgilerini saklar ve bir yayın aldığında, yayının konu ve seçim dizgisiyle eşleşen bir abonelik olup olmadığını saptamak için bilgileri tarar. Eşleşen her abonelik için, kuyruk yöneticisi yayını abonenin abone kuyruğuna yönlendirir. Bir kuyruk yöneticisi abonelikleri hakkında sakladığı bilgiler, DIS SUB ve DIS SBSTATUS komutları kullanılarak görüntülenebilir.

Bir abonelik, yalnızca aşağıdaki olaylardan biri gerçekleştiğinde silinir:

- MQCLOSE çağrısını kullanarak abone olmayan abone (abonelik kalıcı olmayan bir şekilde yapıldıysa).
- Aboneliğin süresi dolacak.
- Abonelik, DELETE SUB komutunu kullanarak sistem denetimcisi tarafından silinir.
- Abone uygulaması sona erer (abonelik sürgüsüz olarak yapıldıysa).
- Kuyruk yöneticisi durdurulmuş ya da yeniden başlatıldıysa (abonelik durdurulamaz bir şekilde yapıldıysa).

İletiler alınırken, MQGET çağrısında uygun seçenekleri kullanın. If your application processes only messages for one subscription then, as a minimum, you should use `get-by-correlid`, as demonstrated in the C sample program `amqssbxa.c` and at yönetilmeyen MQ abonesi. Kullanılacak **`CorrelId`** , MQSD 'deki MQSUB' dan döndürülür.**`SubCorrelId`** alanı.

İlgili kavramlar

Eşkopyalanmış ve paylaşılan abonel

İlgili başvurular

sharedSubscription özelliğinin nasıl tanımlamaya ilişkin örnekler

Yönetilen kuyruklar ve yayınlama/abone olma

Bir abonelik oluşturduğunuzda, yönetilen kuyruğa alma seçeneğini kullanmayı seçebilirsiniz. Yönetilen kuyruğa alma işlemi, abonelik yarattığınızda otomatik olarak bir abonelik kuyruğu oluşturulacaksa kullanılır. Yönetilen kuyruklar, aboneliğin dayanıklısına uygun olarak otomatik olarak döşenir. Yönetilen kuyrukların kullanılması, yayınları almak için kuyruklar yaratma konusunda endişelenmenize gerek olmadığı ve sürekli olmayan bir abonelik bağlantısı kapatılmışsa, tüketilmeyen yayınlar otomatik olarak abone kuyruklarından kaldırılır.

Bir uygulamanın, abone kuyruğu olarak belirli bir kuyruğu kullanmaya gerek yoksa, bu uygulamanın aldığı yayınlara ilişkin hedef, MQSO_MANYED abonelik seçeneği kullanılarak *yönetilen aboneliklerden*

kullanılmasını sağlar. Yönetilen bir abonelik yaratırsanız, kuyruk yöneticisi, kuyruk yöneticisinin yayınların alınacağı yerde yarattığı bir abone kuyruğuna ilişkin aboneye bir nesne tanıtıcısı döndürür. This is because a *yönetilen abonelikler* is one where IBM MQ handles the subscription. Kuyruğa göz atmanıza, kuyruğa girmenize ya da kuyruğa sormanıza olanak tanıyan kuyruğun nesne tanıtıcısı döndürülür (geçici dinamik kuyruklara belirtik olarak erişim verilmediyse, yönetilen bir kuyruğun özniteliklerini koymak ya da ayarlamak olanaklı değildir).

Aboneliğin dayanıklılığı, kuyruk yöneticisine abone olunan uygulamanın bağlantısı kesildiğinde yönetilen kuyruğun mı kalacağını belirler.

Yönetilen abonelikler özellikle, kalıcı olmayan aboneliklerle kullanıldığında yararlı olur; çünkü uygulamanın bağlantısı sona erdiğinde, tüketilmeyen iletiler, kuyruk yöneticinizde süresiz olarak yer alan abone kuyruğunda başka bir şekilde kalır. Yönetilen bir abonelik kullanıyorsanız, yönetilen kuyruk geçici bir dinamik kuyruk olur ve aşağıdaki nedenlerden herhangi biri için bağlantı kesildiğinde, tüketilmeyen iletilerle birlikte silinecektir.

- MQCO_REMOVE_SUB ile MQCLOSE kullanılır ve yönetilen Hobj kapatılır.
- Kalıcı olmayan bir abonelik (MQSO_NON_DAYANIKLI) kullanılarak bir uygulama bağlantısı kaybedilir.
- Abonelik süresi dolduğu ve yönetilen Hobj kapatıldığı için bir abonelik kaldırıldı.

Yönetilen abonelikler, sürekli abonelikler ile de kullanılabilir, ancak bağlantı yeniden açıldığında bunların alınabilmesi için, tüketilmeyen iletileri abone kuyruğunda bırakmak isteyebilirsiniz. Bu nedenle, dayanıklı abonelikler için yönetilen kuyruklar kalıcı bir dinamik kuyruk şeklini alır ve abone olan uygulamanın kuyruk yöneticisine bağlantısı kesildiğinde kalır.

Kalıcı bir dinamik yönetilen kuyruk kullanmak istiyorsanız, aboneliğiniz için süre bitimi ayarlayabilirsiniz; böylece, bağlantı kesildikten sonra kuyruk hala var olmaya devam eder, süresiz olarak var olmaya devam etmeyecektir.

Yönetilen kuyruğu silerseniz, bir hata iletisi alırsınız.

Yaratılan yönetilen kuyruklar, her birinin benzersiz olması için, sonundaki sayılarla (zaman damgaları) adlandırılır.

Abonelik dayanıklılığı

Abonelikler kalıcı ya da kalıcı olmayacak şekilde yapılandırılabilir. Abonelik dayanıklılığı, abone olan uygulamaların kuyruk yöneticisiyle bağlantısını kesmesi sırasında aboneliklere ne olacağını belirler.

Sürekli abonelikler

Abone olan bir uygulamanın kuyruk yöneticisiyle bağlantısı kapatıldığında sürekli abonelikler var olmaya devam eder. Bir abonelik dayanıklıysa, abone olan uygulama bağlantıyı kestiğinde abonelik yerinde kalır ve abonelik oluşturulduğunda döndürülen **SubName** kullanılarak abonelik isteği yeniden bağlandığında abone olan uygulama tarafından kullanılabilir.

Sürekli abone olunurken, bir abonelik adı (**SubName**) gereklidir. Abonelik adlarının, bir aboneliği tanımlamak için kullanılabilmesi için kuyruk yöneticisi içinde benzersiz olması gerekir. Sürdürmek istediğiniz bir abonelik belirtilirken, abonelik bağlantısını kasıtlı olarak kapattıysanız (MQCO_KEEP_SUB seçeneğini kullanarak) ya da kuyruk yöneticisiyle bağlantınız kesildiyse, bu tanımlama yöntemi gereklidir. MQSO_RESUME seçeneğiyle MQSUB çağrısıyla var olan bir aboneliği sürdürebilirsiniz. Abonelik adları, SUBTYPE ALL ya da ADMIN ile DISPLAY SBSTATUS komutunu kullanırsanız da görüntülenir.

Bir uygulama artık sürekli abonelik gerektirmediğinde, MQCO_REMOVE_SUB seçeneğiyle MQCLOSE işlev çağrısı kullanılarak kaldırılabilir ya da MQSC komutu DELETE SUB kullanılarak el ile silinebilir.

Bir konuda sürekli aboneliklerin yapılıp yapılamayacağını belirtmek için **DURSUB** konu özniteliğini kullanabilirsiniz.

MQSO_RESUME seçeneği kullanılarak bir MQSUB çağrısından dönüşte, abonelik süre bitimi, kalan süre bitimi değil, aboneliğin özgün süre bitimine ayarlanır.

Bir kuyruk yöneticisi, abone uygulaması bağlı olmasa da sürekli bir aboneliği karşılamak için yayınlar göndermeye devam eder. Bu, abone kuyruğundaki iletilerin oluşturulmasına neden olur. Bu sorunu

önlemenin en kolay yolu, uygun olduğunda sürekli olmayan bir abonelik kullanmaktır. Ancak, sürekli aboneliklerin kullanılması gerektiğinde, abone İzlemeli yayınlar seçeneğini kullanarak abone olursa, iletilerin oluşturulması önlenebilir. Bir abone, MQSUBRQ çağrısıyla yayınları ne zaman alacağını denetleyebilir.

Sürekli olmayan abonelikler

Kalıcı olmayan abonelikler, abone olan uygulamanın kuyruk yöneticisine olan bağlantısı açık kaldığı sürece var olur. Abone olan uygulama, kuyruk yöneticisiyle bağlantıyı bilerek kestiğinde ya da bağlantı kesildiğinde abonelik kaldırılır. Bağlantı kapatıldığında, abonelikle ilgili bilgiler kuyruk yöneticisinden kaldırılır ve DISPLAY SBSTATUS komutunu kullanarak abonelikleri görüntülediğinizde artık görüntülenmez. Abone kuyruğuna başka ileti konmaz.

Sürekli olmayan abonelikler için abone kuyruğundaki tüketilmemiş yayınlara ne olacağı aşağıdaki gibi belirlenir.

- Abone olan bir uygulama yönetilen hedefkullanıyorsa, kullanılmayan yayınlar otomatik olarak kaldırılır.
- Abone olan uygulama abone olduğunda kendi abone kuyruğuna bir tanıtıcı sağlarsa, kullanılmayan iletiler otomatik olarak kaldırılmaz. Uygunsa, kuyruğun temizlenmesi uygulamanın sorumluluğundadır. Kuyruk birden çok abone ya da diğer noktadan noktaya uygulamalar tarafından paylaşılıyorsa, kuyruğun tamamen temizlenmesi uygun olmayabilir.

Kalıcı olmayan abonelikler için gerekli olmasa da, kuyruk yöneticisi tarafından bir abonelik adı sağlandıysa kullanılır. Abonelik adlarının bir aboneliği tanımlamak için kullanılabilmesi için kuyruk yöneticisi içinde benzersiz olması gerekir.

İlgili kavramlar

Klonlanan ve paylaşılan abonelikler

İlgili görevler

JMS 2.0 paylaşılan aboneliklerinin kullanılması

İlgili başvurular

sharedSubscription özelliğinin nasıl tanımlanacağına ilişkin örnekler

Seçim dizgileri

Seçim dizgisi , bir yayınla eşleşip eşleşmediğini belirlemek için yayına uygulanan bir ifadedir. Seçim dizeleri genel arama karakterleri içerebilir.

Abone olduğunuzda, bir konu belirtmenin yanı sıra, bu yayınları ileti özelliklerine göre seçmek için bir seçim dizgisi belirtebilirsiniz.

Seçim dizgisi, yayıncı tarafından her aboneye teslim için değiştirilmeden önce, iletiyle karşılaştırılarak değerlendirilir. Seçim dizgisinde, yayınlama işleminin bir parçası olarak değiştirilebilecek alanları kullanırken dikkatli olun. Örneğin, MQMD alanları `UserIdentifier`, `MsgId`ve `CorrelId`.

Seçim dizgileri, yayınlama işleminin bir parçası olarak kuyruk yöneticisi tarafından eklenen ileti özelliği alanlarının hiçbirine (bkz. Yayınlama/abone olma ileti özellikleri) gönderme yapmak, yayın için konu dizesini içeren `MQTopicStrin`gileti özelliği dışında.

İlgili kavramlar

Seçim dizgisi kuralları ve kısıtlamaları

Konular

Konu, yayınlama/abone olma iletisinde yayınlanan bilgilerin konusu.

Noktadan noktaya iletişim sistemlerindeki iletiler belirli bir hedef adrese gönderilir. Konuya dayalı yayınlama/abone olma sistemlerindeki iletiler, iletinin içeriğini açıklayan konuya dayalı olarak abonelere gönderilir. İçeriğe dayalı sistemlerde, iletiler iletinin içeriğini temel alan abonelere gönderilir.

IBM MQ yayınlama/abone olma sistemi, konuya dayalı bir yayınlama/abone olma sistemidir. Yayınlayıcı bir ileti oluşturur ve bunu, yayının konusuna en uygun olan bir konu dizesiyle yayınlar. Yayınları almak

için, bir abone, yayın konularını seçmek için bir kalıp eşleştirmesi konu dizgisiyle bir abonelik oluşturur. Kuyruk yöneticisi, yayınlara, yayın konularıyla eşleşen abonelikleri olan ve yayınları alma yetkisine sahip abonelere teslim eder. “Konu dizgileri” sayfa 65makalesi, bir yayının konusunu tanımlayan konu dizgilerinin sözdizimini açıklar. Aboneler, hangi konuların alınacağı seçmek için konu dizgileri de oluşturur. Abone yaratılacak konu dizgileri, yayınlardaki konu dizgileriyle eşleşen örüntü eşleştirmesi için iki alternatif genel arama şemasını içerebilir. Kalıp eşleştirme, “Joker şemalar” sayfa 66içinde açıklanmıştır.

Konu tabanlı yayınlama/abone olma, yayıncılar ya da denetimciler, konuları konu başlıklarına sınıflandırmaktan sorumludur. Genellikle konular, konu dizgisinde alt konular yaratmak için `'/'` karakteri kullanılarak, konu ağaçlarına sıradüzensel olarak sıralanır. Konu ağaçlarına ilişkin örnekler için bkz. “Konu ağaçları” sayfa 72 . Konular, konu ağacındaki düğümlerdir. Konular, başka alt konuları olmayan ya da alt konuları olan ara düğümler olan yaprak düğümler olabilir.

Konuları sıradüzenli bir konu ağacına düzenlemeyle paralel olarak, konuları yönetici konu nesneleriyle ilişkilendirebilirsiniz. Bir konuya, konunun bir yönetim konusu nesnesiyle ilişkilendirilerek, konunun bir kümede dağıtılıp dağıtılmayacağı gibi, bir konuya öznitelik atayabilirsiniz. İlişkilendirme, yönetimle ilgili konu nesnesinin `TOPICSTR` özniteliğini kullanarak konuyu adlandırılarak yapılır. Bir yönetim konusu nesnesini belirtik olarak bir konu ile ilişkilendirmezseniz, konu, bir denetim konusu nesnesiyle ilişkilendirdiğiniz *sahip* konu ağacındaki en yakın atasının özniteliklerini devralır. Herhangi bir üst başlık tanımlamadıysanız, bu başlık `SYSTEM.BASE.TOPIC`. Yönetimle ilgili konu nesneleri “Yönetimle ilgili konu nesneleri” sayfa 73içinde açıklanmıştır.

Not: Bir konunun tüm özniteliklerini `SYSTEM.BASE.TOPIC`, doğrudan `SYSTEM.BASE.TOPIC`. For example, in the topic space of US states, `USA/Alabama` `USA/Alaska`, and so on, USA is the root topic. Kök konunun ana amacı, yanlış aboneliklerle eşleşen yayınlardan kaçınmak için çakışmayan, çakışmayan konu alanları oluşturmaktır. Ayrıca, tüm konu alanınızı etkileyecek şekilde, kök konuunuzun özniteliklerini değiştirebilirsiniz. Örneğin, **CLUSTER** özniteliğinin adını ayarlayabilirsiniz.

Bir konuya yayınlayıcı ya da abone olarak başvuruda bulunduğunuzda, bir konu dizgisi belirtme ya da bir konu nesnesinden gönderme yapmak için bir seçeneğiniz vardır. Ya da her ikisini de yapabilirsiniz; bu durumda, kaynağınız, konu nesnesinin bir alt konusunu tanımlamanızı sağlar. Kuyruk yöneticisi konuyu, konu nesnesinde belirtilen konu dizgisi önekine ekleyerek, iki konu dizgisinin arasına ek bir `'/'` ekleme (örneğin, *konu dizilimi/nesne dizgisi*) ile konuyu tanıtır. “Konu dizgileri birleştirilmesi” sayfa 70 , bunu daha ayrıntılı olarak açıklar. Sonuçta ortaya çıkan konu dizesi, konuyu tanımlamak ve bunu bir yönetici konu nesnesiyle ilişkilendirmek için kullanılır. Denetim konusu nesnesi, ana konuya karşılık gelen konu nesnesiyle aynı konu nesnesiyle aynı olmak zorunda değildir.

İçerik tabanlı yayınlama/abone olma içinde, almak istediğiniz iletileri, her iletinin içeriğini arayan seçim dizgileri sağlayarak tanımladınız. IBM MQ , iletinin tam içeriği yerine ileti özelliklerini tarayan ileti seçicileri kullanarak bir ara içerik tabanlı içerik tabanlı yayınlama/abone olma özelliğini sağlar; bkz. Seçiciler. İleti seçicilerinin oksal kullanımı bir konuya abone olmak ve sonra seçimi sayısal bir özellikle nitelemek. Seçici, yalnızca belirli bir aralıktaki değerlerle ilgilendiğinizi belirtmenizi sağlar; karakter ya da konu tabanlı joker karakter kullanarak yapamadığınız bir şey. İletinin tam içeriğine dayalı olarak süzgeç uygulamaya gerek duyarsanız, IBM Integration Bus' u kullanmanız gerekir.

Konu dizgileri

Bir konu dizesini kullanarak konu olarak yayınladığınız etiket bilgileri. Karakter ya da konu tabanlı genel arama karakteri konu dizgilerini kullanarak konu gruplarına abone olun.

Konular

Konu dizesi bir yayınlama/abone olma iletisinin konusunu tanımlayan bir karakter dizgisidir. Konu dizesini oluştururken beğendiğiniz karakterleri kullanabilirsiniz.

Konu dizesi

IBM WebSphere MQ 7 yayınlama/abone olma özelinde üç karakterin özel anlamı vardır. Bunlar, bir konu dizgisinde herhangi bir yere izin verilir, ancak bunları dikkatli bir şekilde kullanın. Özel karakterlerin kullanımı “Konu tabanlı genel arama karakteri şeması” sayfa 67içinde açıklanmıştır.

Eğik çizgi (/)

Konu düzeyi ayırıcısı. Konuyu bir konu ağacına yapılandırmak için `'/'` karakterini kullanın.

Avoid empty topic levels, `'//'`, if you can. Bu, konu sıradüzenindeki konu dizgisindeki düğümlere karşılık gelir. Bir konu dizgisindeki baştaki ya da sondaki `'/'` , baştaki ya da sondaki boş bir düğüme karşılık gelir ve önlenmelidir.

Hash işareti (#)

Aboneliklerde çok düzeyli bir joker karakter oluşturmak için `'/'` ile birlikte kullanılır. Yayınlanan konuları adlamak için kullanılan konu dizgilerinde `'/'` bitişiğindeki `'#'` komutunu kullanarak dikkatli olun. “Konu dizgileri örnekleri” sayfa 66 shows a sensible use of `'#'`.

The strings `'.../#/...'`, `'#/...'` and `'.../#'` have a special meaning in subscription topic strings. Dizgiler, konu sıradüzenindeki bir ya da daha fazla düzeydeki tüm konuları eşleştirir. Bu nedenle, bu sıralardan biriyle bir konu oluşturduysa, konu sıradüzeninde birden çok düzeyde bulunan tüm konulara da abone olmadan, bu sıralara abone olamazdınız.

Artı işareti (+)

Aboneliklerde tek düzeyli joker karakter oluşturmak için `'/'` ile birlikte kullanılır. Yayınlanan konuları adlamak için kullanılan konu dizgilerinde `'/'` bitişiğindeki `'+'` komutunu kullanarak dikkatli olun.

The strings `'.../+/...'`, `'+/...'` and `'.../+'` have a special meaning in subscription topic strings. Dizgiler, konu sıradüzeninde bir düzey altındaki tüm konuları eşleştirir. Bu nedenle, bu sıralardan biriyle bir konu oluşturursanız, konu sıradüzenindeki bir düzeydeki tüm konulara da abone olmadan, bu diziye abone olamazdınız.

Konu dizgileri örnekleri

```
IBM/Business Area#/Results
IBM/Diversity/%African American
```

İlgili başvurular

Konu

Joker şemalar

Birden çok konuya abone olmak için kullanılan iki genel arama tipi şeması vardır. Şema seçimi bir abonelik seçeneğidir.

MQSO_WILDCARD_KONUSU
Konu tabanlı genel arama şemasını kullanarak abone olmak için konuları seçin.

Genel arama karakteri belirtik olarak seçilmezse, bu varsayılan değer olarak varsayılan değer olur.

MQSO_WILDCARD_CHAR
Karakter tabanlı genel arama şemasını kullanmak için abone olunması gereken konuları seçin.

DEFINE SUB komutunda **wschema** parametresini belirterek şemayı ayarlayın. Ek bilgi için bkz. DEFINE ALT.

Not: IBM WebSphere MQ 7.0 öncesinde oluşturulan abonelikler her zaman karakter tabanlı joker karakter şemasını kullanır.

Örnekler

```
IBM/+/Results
#/Results
```

```
IBM/Software/Results
IBM/*ware/Results
```

Konu tabanlı genel arama karakteri şeması
Konu tabanlı genel arama karakterleri, abonelerin bir kerede birden çok konuya abone olmalarına olanak sağlar.

Topic-based wildcards are a powerful feature of the topic system in IBM MQ publish/subscribe. Abonelikler için çok düzeyli genel arama karakteri ve tek düzeyli genel arama karakteri kullanılabilir, ancak bir iletinin yayıncısı tarafından bir konu içinde kullanılamaz.

Konu tabanlı genel arama şeması, konu düzeyine göre gruplanmış yayınları seçmenizi sağlar. Konu sıradüzenindeki her düzey için, o konu düzeyindeki aboneliğin dizginin yayındaki dizgiyle tam olarak eşleşmesi gerekip gerekmediğini seçebilirsiniz. Örneğin, `IBM/+/Results` aboneliği tüm konuları seçer,

```
IBM/Software/Results
IBM/Services/Results
IBM/Hardware/Results
```

İki tip genel arama karakteri vardır.

Çok düzeyli genel arama karakteri

- Çok düzeyli genel arama karakteri aboneliklerde kullanılır. Bir yayında kullanıldığında, hazır bilgi olarak kabul edilir.
- The multilevel wildcard character `'#'` is used to match any number of levels within a topic. For example, using the example topic tree, if you subscribe to `'USA/Alaska/#'`, you receive messages on topics `'USA/Alaska'` and `'USA/Alaska/Juneau'`.
- Çok düzeyli genel arama karakteri sıfır ya da daha fazla düzeyi gösterebilir. Bu nedenle, `'USA/#'` tekil `'USA'` ile de eşleşebilir; burada `'#'` , sıfır düzeylerini gösterir. Ayrı bir düzey olmadığı için, konu düzeyi ayırıcısı bu bağlamda anlamsızdır.
- Çok düzeyli genel arama karakteri, yalnızca kendi başına ya da konu düzeyi ayırıcı karakterinin yanında belirtildiğinde etkili olur. Bu nedenle, `'#'` ve `'USA/#'` , `'#'` karakterinin genel arama karakteri olarak işlem gördüğü geçerli konulardır. Ancak, `'USA#'` aynı zamanda geçerli bir konu dizgisiyse, `'#'` karakteri genel arama karakteri olarak kabul edilmez ve özel bir anlamı yoktur. Ek bilgi için “Konu tabanlı genel arama karakterleri genel arama karakteri değilse” sayfa 69 başlıklı konuya bakın.

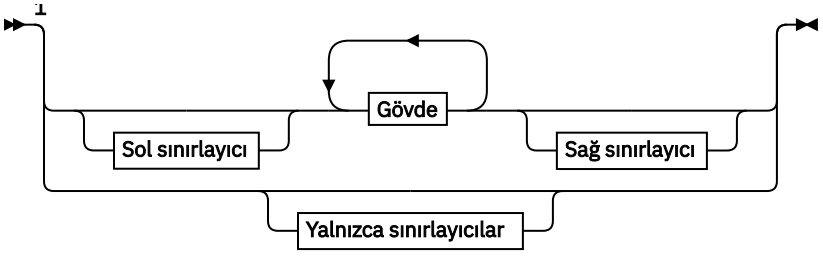
Tek düzeyli genel arama karakteri

- Aboneliklerde tek genel arama karakteri kullanılır. Bir yayında kullanıldığında, hazır bilgi olarak kabul edilir.
- Tek düzeyli genel arama karakteri `'+'` , bir ve konu düzeyinde tek bir genel arama karakteriyle eşleşir. For example, `'USA/+'` matches `'USA/Alabama'`, but not `'USA/Alabama/Auburn'`. Tek düzeyli genel arama karakteri yalnızca tek bir düzeyle eşleştiğinden `'USA/+'` , `'USA'` ile eşleşmez.
- Tek düzeyli joker karakter, konu ağacındaki herhangi bir düzeyde ve çok düzeyli genel arama karakteriyle birlikte kullanılabilir. Tek düzeyli genel arama karakteri, konu düzeyi ayırıcının yanında belirtilmelidir; ancak, bu ayırıcı kendi başına belirtildiğinde belirtilmelidir. Bu nedenle, `'+'` ve `'USA/+'` , `'+'` karakterinin genel arama karakteri olarak işlem gördüğü geçerli konulardır. Ancak, `'USA+'` aynı zamanda geçerli bir konu dizgisiyse, `'+'` karakteri genel arama karakteri olarak kabul edilmez ve özel bir anlamı yoktur. Ek bilgi için “Konu tabanlı genel arama karakterleri genel arama karakteri değilse” sayfa 69 başlıklı konuya bakın.

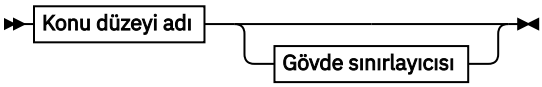
Konuya dayalı genel arama karakterinin sözdiziminin çıkış karakteri yok. `'#'` ve `'+'` 'in genel arama karakteri olarak mı değerlendirileceğini, yoksa bunların bağlamına bağlı olup olmadığını belirleyin. Ek bilgi için “Konu tabanlı genel arama karakterleri genel arama karakteri değilse” sayfa 69 başlıklı konuya bakın.

Not: Bir konu dizgisinin başı ve sonu özel bir şekilde işlem görür. Dizginin sonunu belirtmek için `'$'` u kullanarak, `'$#/...'` çok düzeyli bir genel arama karakteridir ve `'$/#/..'`. kökteki boş bir düğümdür ve çok düzeyli bir genel arama karakteridir.

Konu tabanlı genel arama karakteri dizesi

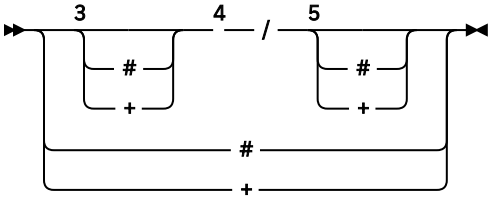


Gövde



Konu düzeyi adı

Yalnızca sınırlayıcılar



Notlar:

[1] Boş değerli ya da sıfır uzunluklu bir konu dizgisi geçersiz

[2] Char tabanlı ve konu tabanlı genel arama karakterleri arasında uyumluluk sağlamak için düzey adı dizgilerinde `*`, `?`, `%` hiçbirini kullanmamanız önerilir.

[3] Bu durumlar, *sol sınırlayıcı* kalıbına eşdeğerdir.

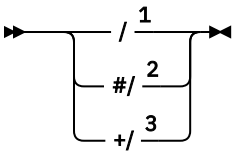
[4] Genel arama karakteri olmayan`/` , tek boş bir konu ile eşleşmez.

[5] Bu durumlar *sağ sınırlayıcı* örüntüye eşdeğerdir.

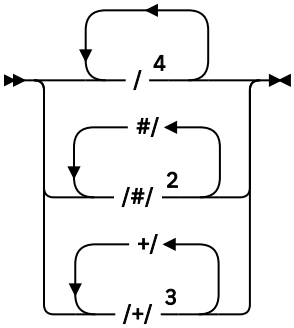
[6] Her konuyu eşleştirin.

[7] Sadece bir seviyenin olduğu her konuyu eşleştirin.

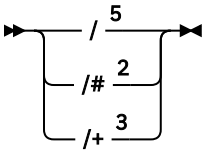
Sol sınırlayıcı



Gövde sınırlayıcı



Sağ sınırlayıcı



Notlar:

[1] Konu dizgisi boş bir konu ile başlıyor
[2] Sıfır ya da daha fazla düzeye eşleştirir. Çok düzeyli birden çok eşleşme dizesi, tek bir çok düzeyli eşleşme dizgisiyle aynı etkiye sahiptir.
[3] Tam olarak bir düzey ile eşleşir.
[4] / / boş bir konudur-konu dizgisi olmayan bir konu nesnesidir.
[5] Konu dizesi boş bir konu ile bitiyor

Konu tabanlı genel arama karakterleri genel arama karakteri değilse

'+' ve '#' genel arama karakterleri, bir konu düzeyinde diğer karakterlerle (kendileri de içinde olmak üzere) karıştırıldığında özel bir anlam ifade etmiyorlardı.

Bu, '+' ya da '#' sözcüğünü içeren konuların bir konu düzeyindeki diğer karakterlerle birlikte yayınlanabileceği anlamına gelir.

Örneğin, aşağıdaki iki konuyu göz önünde bulundurun:

1. level0/level1/+/level4/#
2. level0/level1/#+/level4/level#

İlk örnekte, '+' ve '#' karakterleri joker karakterler olarak ele alınır ve bu nedenle, yayınlanacak olan ancak bir abonelikte geçerli olan bir konu dizgisinde geçerli değildir.

İkinci örnekte, '+' ve '#' karakterleri joker karakterler olarak işlenmez ve dolayısıyla konu dizgisi hem yayınlanabilir hem de abone olunabilir.

Örnekler

```
IBM/+/Results
#/Results
IBM/Software/Results
```

Karaktere dayalı genel arama karakteri şeması
Karakter tabanlı genel arama şeması, geleneksel karakter eşleştirmesine dayalı konuları seçmenize olanak tanır.

'*'dizgisini kullanarak bir konu sıradüzeninde birden çok düzeyde tüm konuları seçebilirsiniz. Karakter tabanlı genel arama şemasındaki '*' kullanılması,topickonu tabanlı genel arama karakteri dizgisinin kullanılmasının eşdeğeridir. '#'

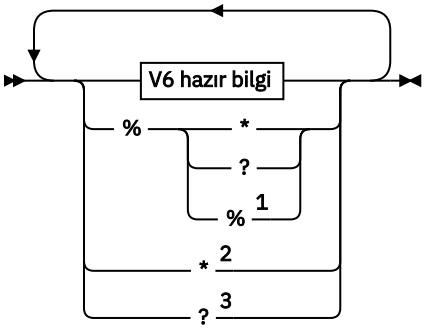
' *x*/*/*y* ' is equivalent to ' *x*/#/*y* ' in the topic-based scheme, and selects all topics in the topic hierarchy between levels ' *x* and *y* ', where ' *x* ' and ' *y* ' are topic names that are not in the set of levels returned by the wildcard.

Konu tabanlı şemada'/+/' , karakter tabanlı şemada tam olarak eşdeğer bir değer içermiyor. 'IBM/*/Results', 'IBM/Patents/Software/Results'öğesini de seçerdi. Yalnızca, sıradüzeninin her düzeyindeki konu adları kümesi benzersizse, her zaman aynı eşleşmeleri veren iki şemaya sahip sorgular oluşturabilir misiniz?

Karakter tabanlı şemada genel bir şekilde kullanılan '*' ve '?' , konu tabanlı şemada eşdeğerleri yoktur. Konu tabanlı şema, joker karakterler kullanılarak kısmi eşleştirme gerçekleştirmiyor. The character based wildcard subscription 'IBM/*ware/Results' has no topic-based equivalent.

Not: Karakter genel arama karakteri abonelikleri kullanılarak eşleşmeler, konu tabanlı abonelikler kullanılarak eşleşmelere göre daha düşüktür.

Karakter tabanlı genel arama dizgisi



V6 hazır bilgi

Notlar:

[1] Bir hazır bilgi olarak işlem görebilmesi için, aşağıdaki karakterden kaçış yolu anlamına gelir. '%' must be followed by either '*', '?' or '%'. Bkz. “Konu dizgileri örnekleri” sayfa 66.

[2] Abonelikte sıfır ya da daha fazla karakter eşleştir anlamına gelir.

[3] Bir abonelikte tam olarak bir karakter eşleşmesi anlamına gelir.

Örnekler

```
IBM/*/Results
IBM/*ware/Results
```

Konu dizgileri birleştirilmesi

Abonelikleri oluştururken ya da konuları yayınlayabilmeniz için konuları açarken, konu dizgisi iki ayrı alt konu dizesiyle ya da "alt konular"birleştirilerek oluşturulabilir. Uygulama ya da denetim komutu bir konu dizgisi olarak bir alt konu, diğeri bir konu nesnesiyle ilişkilendirilmiş konu dizgisiyle sağlanır. Alt konuyu kendi başına konu dizesi olarak kullanabilir ya da yeni bir konu adı oluşturmak için birleştirebilirsiniz.

For example, when you define a subscription using the MQSC command **DEFINE SUB**, the command can take either **TOPICSTR** (topic string) or **TOPICOBJ** (topic object) as an attribute, or both together. Yalnızca **TOPICOBJ** sağlandıysa, o konu nesnesiyle ilişkilendirilmiş konu dizgisi konu dizgisi olarak kullanılır. Yalnızca **TOPICSTR** sağlandıysa, konu dizgisi olarak kullanılır. If both are provided, they are concatenated to form a single topic string in the form **TOPICOBJ** / **TOPICSTR**, where the **TOPICOBJ** configured topic string is always first and the two parts of the string are always separated by a "/" character.

Similarly, in an MQI program the full topic name is created by MQOPEN. Yayınlama/abone olma MQI çağrılarında kullanılan iki alandan oluşur ve listelenen sırada:

1. **ObjectName** alanında adlandırılan konu nesnesinin **TOPICSTR** özniteliği.
2. Uygulama tarafından sağlanan alt konuyu tanımlayan **ObjectString** parametresi.

Sonuçtaki konu dizgisi, **ResObjectString** parametresine döndürülür.

Her alanın ilk karakteri boş değerli ya da boş bir karakter değilse ve alan uzunluğu sıfırdan büyükse, bu alanlar var olarak kabul edilir. Alanlardan yalnızca biri varsa, bu alan konu adı olarak değiştirilmeden kullanılır. Her iki alanda bir değer yoksa, çağrı neden kodu MQRC_UNKNOWN_OBJECT_NAMEile başarısız olur ya da tam konu adı geçersiz ise MQRC_TOPIC_STRING_ERROR .

Her iki alan da mevcutsa, sonuçtaki birleşik konu adının iki öğesi arasına bir "/" karakteri eklenir.

Çizelge 2 sayfa 71 , konu dizgi bitiştirme örneklerini gösterir:

Çizelge 2. Konu dizgi bitiştirme örnekleri

Konu nesnesininTOPICSTR	**Uygulama ya da DEFINE SUB komutu tarafından sağlanan konu dizesi**	**Tam konu adı**	**Açıklama**
Futbol/İskoçya	''	Futbol/İskoçya	Konu nesnesinin TOPICSTR tek başına kullanılır.
''	Futbol/İskoçya	Futbol/İskoçya	ObjectString/TOPICSTR yalnız kullanılır.
Futbol	Puanlar	Futbol/İskoçya	Bitiştirme noktasına bir "/" karakteri eklenir.
Futbol	/İskolar	Futbol/İskoçya	İki dizgi arasında 'boş düğüm' üretildi. Bu, "Futbol/Scores"' dan farklıdır.
/Futbol	Puanlar	/Futbol/Scores	Konu, 'boş düğüm' ile başlar. Bu, "Futbol/Scores"' dan farklıdır.

The "/" character is considered as a special character, providing structure to the full topic name in“Konu ağaçları” sayfa 72. Konu ağacının yapısı etkilendiği için, "/" karakteri başka bir nedenle kullanılmamalıdır. The topic "/Football" is not the same as the topic "Football".

Not: Abonelik yaratırken bir konu nesnesi kullanırsanız, konu nesnesi konu dizgisinin değeri, tanımlama sırasında aboneliğe sabittir. Konu nesnesindeki sonraki değişiklikler, aboneliğin tanımlandığı konu dizesini etkilemez.

Konu dizgilerinde genel arama karakterleri

Aşağıdaki genel arama karakterleri özel karakterlerdir:

- artı işareti (+)
- sayı işareti (#)
- yıldız işareti (*)
- soru işareti (?)

Joker karakterler, abonelik tarafından kullanıldığında özel anlamlara sahiptir. Bu karakterler başka bir yerde kullanıldığında geçersiz olarak kabul edilmiyor, ancak bunların nasıl kullanıldığını anladığınızdan

emin olmanız ve konu nesnelerini yayınlarken ya da tanımlarken bu karakterleri konu dizilimlerinizde kullanmamayı tercih etmeniz gerekir.

If you publish on a topic string with # or + mixed in with other characters (including themselves) within a topic level, the topic string can be subscribed to with either wildcard scheme.

Bir konu dizgisinde, iki / karakteri arasındaki tek karakter olarak # ya da + ile bir konu dizgisinde yayınlanırsa, MQSO_WILDCARD_TOPIC genel arama karakteri kullanılarak bir uygulama tarafından belirtik olarak bir uygulama abone olunamaz. Bu durum, uygulamanın beklenenden daha fazla yayın elde etmesi sonucunu elde eder.

Tanımlı bir konu nesnesinin konu diziliminde genel arama karakteri kullanmamalısınız. Bunu yapmazsanız, nesne bir yayınlayıcı tarafından kullanıldığında ve abonelik tarafından kullanıldığında genel arama karakteri olarak karakter olarak kabul edilir. Bu karışıklığa yol açabilir.

Örnek kod parçacığı

Örnek 2: Bir değişken konusuna yayınlayıcıörnek programından çıkarılan bu kod parçacığı, bir konu nesnesini bir değişken konu dizgisiyle birleştirir.

```
MQOD   td = {MQOD_DEFAULT}; /* Object Descriptor      */
td.ObjectType = MQOT_TOPIC;  /* Object is a topic      */
td.Version = MQOD_VERSION_4;  /* Descriptor needs to be V4  */
strncpy(td.ObjectName, topicName, MQ_TOPIC_NAME_LENGTH);
td.ObjectString.VSPtr = topicString;
td.ObjectString.VSLength = (MQLONG)strlen(topicString);
td.ResObjectString.VSPtr = resTopicStr;
td.ResObjectString.VSBufSize = sizeof(resTopicStr)-1;
MQOPEN(Hconn, &td, MQOO_OUTPUT | MQOO_FAIL_IF_QUIESCING, &Hobj, &CompCode, &Reason);
```

Konu ağaçları

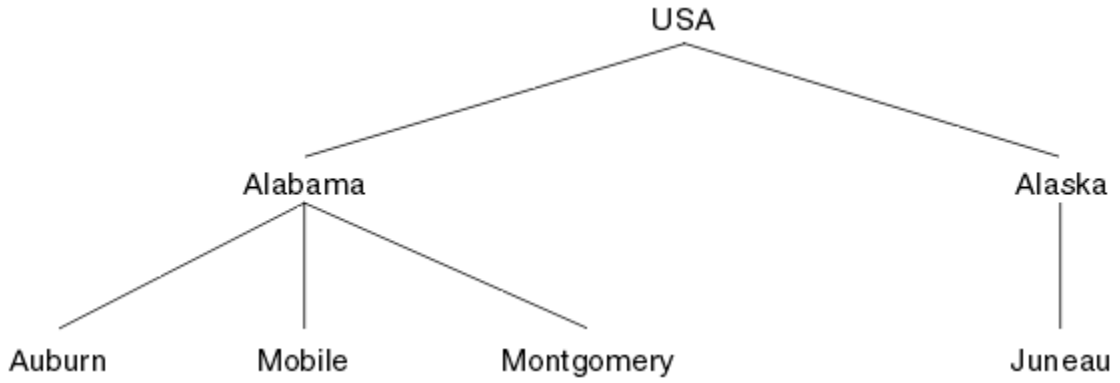
Tanımladığınız her konu, konu ağacında bir öğedir ya da düğümdür. Konu ağacı, önceden MQSC ya da PCF komutlarını kullanarak önceden tanımlanmış konuları içeren ya da içerebilecek boş olabilir. Yeni bir konuyu, konu yaratma komutlarını kullanarak ya da bir yayınlama ya da abonelikte ilk kez konuyu belirterek tanımlayabilirsiniz.

Bir konunun konu dizgisini tanımlamak için herhangi bir karakter dizgisi kullanabilmenize rağmen, sıradüzensel bir ağaç yapısına uygun bir konu dizgisi seçmeniz önerilir. Konu sokmaları ve konu ağaçlarının düşünceli tasarımı aşağıdaki işlemleri yapabilmenize yardımcı olabilir:

- Birden çok konuya abone olma.
- Güvenlik ilkeleri oluşturuluyor.

Düz, doğrusal bir yapı olarak bir konu ağacı oluşturabilmenize rağmen, bir ya da daha fazla kök konumla sıradüzensel bir yapıda bir konu ağacı oluşturmak daha iyi olur. Güvenlik planlaması ve konularıyla ilgili daha fazla bilgi için Güvenlik yayınlama/abone olmabaşlıklı konuya bakın.

Şekil 18 sayfa 72 , bir kök konusu olan bir konu ağacına ilişkin bir örneği gösterir.



Şekil 18. Konu ağacı örneği

Şekildeki her karakter dizgisi, konu ağacındaki bir düğümü temsil eder. Genel bir konu dizesi, konu ağacındaki bir ya da daha fazla düzeyden düğümler toplayarak yaratılır. Düzeyler "/" karakteriyle ayrılır. Tam olarak belirtilmiş bir konu dizgisinin biçimi şöyledir: "root/level2/level3".

Şekil 18 sayfa 72 içinde gösterilen konu ağacındaki geçerli konular şunlardır:

"ABD"
"ABD/Alabama"
"USA/Alaska"
"USA/Alacama/Auburn"
"USA/Alacama/Mobile"
"USA/Alacama/Montgomery"
"USA/Alaska/Juneau"

Konu dizgileri ve konu ağaçlarını tasarladığınızda, kuyruk yöneticisinin yorumlamadığını ya da konu dizgisinin kendisinden anlam türetmeyi denediğini unutmayın. Bu konu, seçilen iletileri o konunun abonelerine göndermek için konu dizesini kullanır.

Bir konu ağacının yapısı ve içeriği için aşağıdaki ilkeler geçerlidir:

- Bir konu ağacındaki düzeylerin sayısı için bir sınır yoktur.
- Bir konu ağacında bir düzeyin adının uzunluğuna ilişkin bir sınır yoktur.
- Herhangi bir sayıda "kök" düğüm olabilir; yani, herhangi bir sayıda konu ağacı olabilir.

İlgili görevler
Konu ağacındaki istenmeyen konuların sayısını azaltma

Yönetimle ilgili konu nesneleri

Yönetimle ilgili bir konu nesnesini kullanarak, konulara varsayılan, varsayılan olmayan öznitelikler atayabilirsiniz.

Şekil 19 sayfa 73 , `Sport` üst düzey bir konunun farklı sporları kapsayan ayrı konulara nasıl bölüneceğini, konu ağacı olarak nasıl görselleştirdiğini gösterir:

Şekil 19. Konu ağacını görselleştirme

Şekil 20 sayfa 73 , konu ağacının her bir sporla ilgili farklı bilgi tiplerini birbirinden ayırmak için nasıl daha fazla bölünebileceğini gösterir:

Şekil 20. Ek konu ağacı

Konu ağacını gösterebilmek için, hiçbir denetim konusu nesnesi tanımlanmalı. Bu ağaçtaki düğümlerin her biri, yayınlama ya da abone olma işleminde yaratılan bir konu dizesiyle tanımlanır. Ağaçtaki her konu, özniteliklerini üst öğelerinden devralır. Varsayılan olarak tüm öznitelikler ASPARENTolarak ayarlandığından, öznitelikler üst konu nesnesinden devralınır. Bu örnekte, her konu `Sport` ile

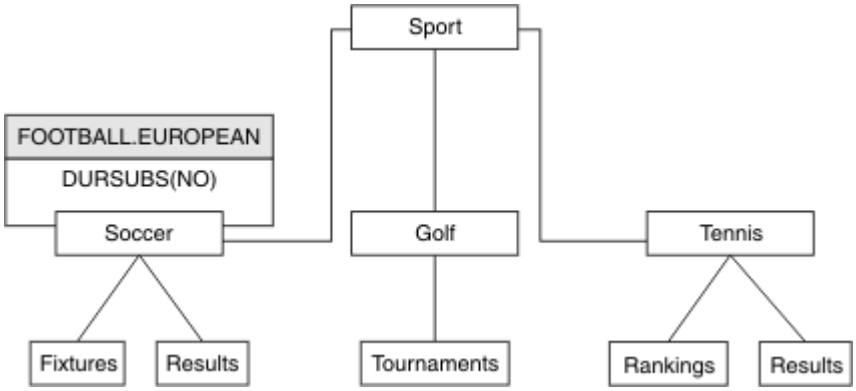
aynı özniteliklere sahiptir. Sport konularının denetim konusu nesnesi yoktur ve öznitelikleri SYSTEM.BASE.TOPIC.

Note, that it is not good practice to give authorities for non-mqm users at the root node of the topic tree, which is SYSTEM.BASE.TOPIC, because the authorities are inherited but cannot be restricted. bu nedenle otoriteleri bu seviyede vererek, yetkililere tüm ağacı verebiliyorsun. Hiyerarşide daha düşük bir konu düzeyinde yetki vermelisiniz.

Denetim konusu nesneleri, konu ağacındaki belirli düğümlere ilişkin belirli öznitelikleri tanımlamak için kullanılabilir. Aşağıdaki örnekte, yönetici konu nesnesi, soccer konularının DURALB özelliğini NOdeğerine ayarlamak için tanımlıdır:

```
DEFINE TOPIC(FOOTBALL.EUROPEAN)
TOPICSTR('Sport/Soccer')
DURSUB(NO)
DESCR('Administrative topic object to disallow durable subscriptions')
```

Konu ağacı şu şekilde görselleştirilebilir:



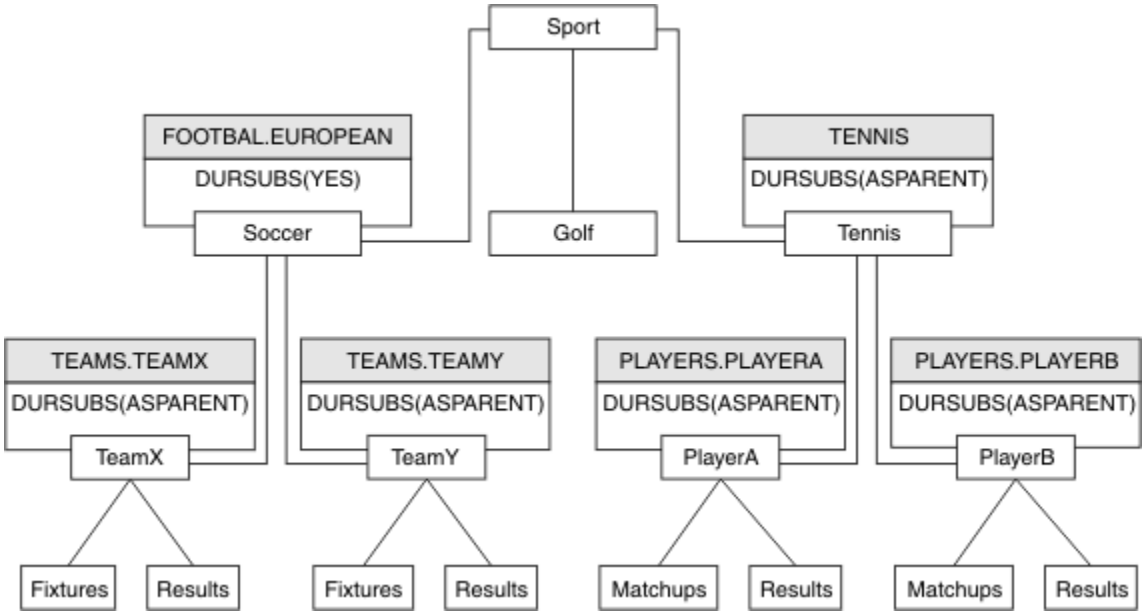
Şekil 21. Spor/Soccer konusuyla ilişkili bir yönetici konu nesnesinin görselleştirilmesi

Ağaçtaki Soccer altındaki konulara abone olan uygulamalar, denetim konusu nesnesi eklenmeden önce kullandıkları konu dizelerini kullanmaya devam edebilir. However, an application can now be written to subscribe using the object name FOOTBALL.EUROPEAN, instead of the string /Sport/Soccer. For example, to subscribe to /Sport/Soccer/Results, an application can specify MQSD.ObjectName as FOOTBALL.EUROPEAN and MQSD.ObjectString as Results.

Bu özellik sayesinde, uygulama geliştiricilerinden konu ağacının bir kısmını gizleyebilirsiniz. Konu ağacındaki belirli bir düğümde bir denetim konusu nesnesi tanımlayın ve uygulama geliştiricileri, kendi konularını düğümün alt öğeleri olarak tanımlayabilir. Geliştiricilerin üst konu hakkında bilgi sahibi olması gerekir, ancak üst ağaçtaki diğer düğümlerle ilgili bilgi sahibi olmamalıdır.

Öznitelikleri edinir

Bir konu ağacında birçok yönetici konu nesnesi varsa, her bir denetim konusu nesnesi varsayılan olarak, özniteliklerini en yakın üst düzey yönetici konusundan edinir. Önceki örnek Şekil 22 sayfa 75içinde genişletilmiştir:



Şekil 22. Çok sayıda denetim konusu nesnesi olan konu ağacı

For example use inheritance to give all the child topics of `/Sport/Soccer` the property that subscriptions are non-durable. FOOTBALL.EUROPEAN ' nin DURSUB özniteliğini NOolarak değiştirin.

Bu öznitelik aşağıdaki komutu kullanarak ayarlanabilir:

```
ALTER TOPIC(FOOTBALL.EUROPEAN) DURSUB(NO)
```

All the administrative topic objects of child topics of `Sport/Soccer` have the property DURSUB set to the default value ASPARENT. FOOTBALL.EUROPEAN , DURSUB özellik değerini NOolarak değiştirdikten sonra, `Sport/Soccer` alt konuları, NO DURSUB özellik değerini devralır. All child topics of `Sport/Tennis` inherit the value of DURSUB from SYSTEM.BASE.TOPIC object. SYSTEM.BASE.TOPIC , YESdeğerine sahiptir.

`Sport/Soccer/TeamX/Results` konusuna dayanıklı bir abonelik yapmaya çalışmak artık başarısız olacaktır; ancak, `Sport/Tennis/PlayerB/Results` ' e kalıcı bir abonelik yapmaya çalışmak başarılı olur.

Controlling wildcard usage with the JOKER property

Yayınların genel arama karakteri konu dizgisi adlarını kullanan abone uygulamalarına teslim edilmesini denetlemek için MQSC **Topic** JOKER özelliğini ya da eşdeğer PCF Topic `WildcardOperation` özelliğini kullanın. JOKER özelliği, iki olası değerden birine sahip olabilir:

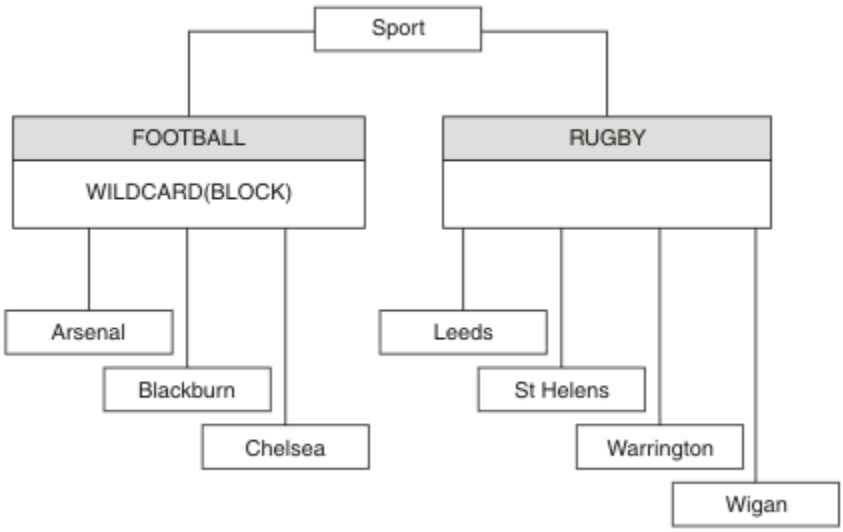
Genel arama karakteri
Bu konuya ilişkin genel arama karakteri aboneliklerinin davranışı.

Passthru
Bu konu nesnesindeki konu dizgisine göre daha az genel lengüzel bir konuya yapılan abonelikler, bu konu ve konu dizgilerine bu konudan daha özel yayınlar alır.

Öbek
Bu konu nesnesindeki konu dizgisinden daha az özel bir konuya yapılan abonelikler, bu konu ya da konu dizgilerine bu konudan daha özel yayınlar almaz.

Abonelikler tanımlandığında bu özniteliğin değeri kullanılır. Bu özniteliği değiştirirseniz, var olan aboneliklerin kapsadığı konular kümesi değişiklikten etkilenmez. Bu senaryo, konu nesneleri yaratıldığında ya da silindiğinde topoloji değiştirildiğinde de geçerlidir; WILDCARD özniteliğinin değiştirilmesinden sonra yaratılan aboneliklerle eşleşen konular kümesi, değiştirilen topoloji kullanılarak yaratılır. Eşleşen konu kümesini var olan abonelikler için yeniden değerlendirilmeye zorlamak istiyorsanız, kuyruk yöneticisini yeniden başlatmanız gerekir.

In the example, “Örnek: Sport yayınlama/abone olma kümesini yaratın” sayfa 79, you can follow the steps to create the topic tree structure shown in Şekil 23 sayfa 76.



Şekil 23. JOKER özelliğini kullanan bir konu ağacı, BLOCK

A subscriber using the wildcard topic string `#` receives all publications to the `Sport` topic and the `Sport/Rugby` subtree. The subscriber receives no publications to the `Sport/Football` subtree, because the `JOKER` property value of the `Sport/Football` topic is `BLOCK`.

Varsayılan değerPASSTHRU ' dir. `Sport` ağacındaki düğümlere `JOKER` özellik değerini PASSTHRU olarak ayarlayabilirsiniz. Düğümlerde `BLOCK` `JOKER` özellik değeri yoksa, PASSTHRU ayarı abonelerin `Sports` ağacındaki düğümlere göre gözlemlediği davranışı değiştirmez.

Örnekte, genel arama karakterinin, teslim edilen yayınları nasıl etkilediğini görmek için abonelikler oluşturun; bkz. Şekil 27 sayfa 81. Bazı yayınlar yaratmak için Şekil 30 sayfa 82 içinde yayınlama komutunu çalıştırın.

```
pub QMA
```

Şekil 24. QMA ' ya Yayınla

Sonuçlar Çizelge 3 sayfa 76içinde gösterilir. `JOKER` propertyözellik değerinin `BLOCK`' in genel arama karakteri kapsamında, genel arama karakterlerinin genel arama karakterlerine sahip aboneliklerini nasıl önlediğini fark edin.

Çizelge 3. QMAtarihinde alınan yayınlar

Abonelik	**Konu dizesi**	**Alınan yayınlar**	**Notlar**

SPORTS	`Sports/#`	`Sports`
		`Sports/Rugby`
		`Sports/Rugby/Leeds`
			All publications to Football subtree blocked by `WILDCARD(BLOCK)` on `Sports/Football`

SARSENAL	`Sports/#/Arsenal`	-	`Sports/Football` üzerinde`WILDCARD(BLOCK)` , `Arsenal`üzerinde genel arama karakteri aboneliğini önler

Çizelge 3. QMAtarihinde alınan yayınlar (devamı var)			
Abonelik	**Konu dizesi**	**Alınan yayınlar**	**Notlar**
SLEEDS	Sports/#/Leeds	Sports/Rugby/Leeds	Sports/Rugby için varsayılan WILDCARD , Leedsüzerinde genel arama karakteri aboneliğini engellememektedir.

Not:

Suppose a subscription has a wildcard that matches a topic object with the JOKER property value BLOCK. Aboneliğin aynı zamanda eşleşen genel arama karakterinin sağında bir konu dizgisi varsa, abonelik hiçbir zaman bir yayını almaz. Engellenen yayınların kümesi, engellenen genel arama karakterinin ebeveynleri olan konular için yayınlardır. Publications to topics that are children of the topic with the BLOCK property value are blocked by the wildcard. Bu nedenle, genel arama karakterinin sağına bir konu içeren abonelik konu dizgileri, hiçbir yayın için hiçbir yayın almayacak.

WILDCARD özellik değerinin BLOCK olarak ayarlanması, genel arama karakterleri içeren bir konu dizgisini kullanarak abone olamayacağınız anlamına gelmez. Böyle bir abonelik normal. The subscription has an explicit topic that matches the topic with a topic object having a JOKER property value BLOCK. It uses wildcards for topics that are parents or children of the topic with the JOKER property value BLOCK. Şekil 23 sayfa 76örneğinde, Sports/Football/# gibi bir abonelik yayınları alabilir.

Genel arama karakterleri ve küme konuları

Küme konu tanımları, bir kümedeki her kuyruk yöneticisine dağıtılır. Kuyruk yöneticisinde bir küme yöneticisinde bulunan bir küme konusuna abonelik, yetkili abonelikleri oluşturan kuyruk yöneticisinde yer alan kuyruk yöneticisinde yer alan bir sonuçla sonuçlanır. Bir yetkili sunucu aboneliği, kümedeki diğer her kuyruk yöneticisinde yaratılır. Genel arama karakterleri içeren, küme konularıyla birleştirilen konuları kullanan abonelikler, davranışı tahmin etmek için zor bir yol gösterebilirler. Bu davranış, aşağıdaki örnekteki açıklamadır.

In the cluster set up for the example, "Örnek: Sport yayınlama/abone olma kümesini yaratın" sayfa 79, QMB has the same set of subscriptions as QMA, yet QMB received no publications after the publisher published to QMA, see Şekil 24 sayfa 76. Sports/Football ve Sports/Rugby konuları küme başlıklarına rağmen, fullsubs.tst içinde tanımlanan abonelikler bir küme konusuna gönderme yapmamış olur. Yetkili sunucu aboneliği, QMB 'tan QMA' a yayılmaz. Yetkili abonelikler olmadan, QMA hiçbir yayın QMB' a iletilmedi.

Some of the subscriptions, such as Sports/#/Leeds, might seem to reference a cluster topic, in this case Sports/Rugby. The Sports/#/Leeds subscription actually resolves to the topic object SYSTEM.BASE.TOPIC.

Sports/#/Leeds gibi bir aboneliğin gönderme yaptığı konu nesnesinin çözülmesine ilişkin kural şu şekildir. Konu dizesini ilk genel arama karakterine kes. İlişkili bir denetim konusu nesnesi olan ilk konuyu aramak için konu dizgisinden sola doğru tarayın. Konu nesnesi bir küme adı belirtebilir ya da bir yerel konu nesnesi tanımlayabilir. In the example, Sports/#/Leeds, the topic string after truncation is Sports, which has no topic object, and so Sports/#/Leeds inherits from SYSTEM.BASE.TOPIC, which is a local topic object.

Kümelenmiş konulara abone olunmanın, genel arama karakterinin yayılmasını nasıl değiştirebileceğini görmek için, upsubs.battoplu iş komut dosyasını çalıştırın. Komut dosyası abonelik kuyruklarını temizler ve fullsubs.tstiçinde küme başlığı aboneliklerini ekler. Bir dizi yayın yaratmak için puba.bat komutunu yeniden çalıştırın; bkz. Şekil 24 sayfa 76.

Çizelge 4 sayfa 78 , yayınların yayınlandığı kuyruk yöneticisine iki yeni abonelik eklenmesinin sonucunu gösterir. Sonuç beklendiği gibi, yeni abonelikler her biri bir yayın alır ve diğer abonelikler tarafından alınan yayınların sayıları değişmez. Diğer küme kuyruk yöneticisininde beklenmeyen sonuçlar ortaya çıkar; bkz. Çizelge 5 sayfa 78.

Çizelge 4. QMAtarihinde alınan yayınlar			
Abonelik	**Konu dizesi**	**Alınan yayınlar**	**Notlar**
SPORTS	Sports/#	Sports Sports/Rugby Sports/Rugby/Leeds	All publications to Football subtree blocked by WILDCARD(BLOCK) on Sports/Football
SARSENAL	Sports/#/Arsenal	-	Sports/Football üzerindeWILDCARD(BLOCK) , Arsenalüzerinde genel arama karakteri aboneliğini önler
SLEEDS	Sports/#/Leeds	Sports/Rugby/Leeds	Sports/Rugby için varsayılan WILDCARD , Leedsüzerinde genel arama karakteri aboneliğini engellememektedir.
FARSENAL	Sports/Football/Arsenal	Sports/Football/Arsenal	Arsenal , aboneliğin bir genel arama karakterinin olmadığı için bir yayın alır.
FLEEDS	Sports/Rugby/Leeds	Sports/Rugby/Leeds	Leeds , herhangi bir etkinlikte bir yayın alır.

Çizelge 5 sayfa 78 shows the results of adding the two new subscriptions on QMB and publishing on QMA. QMB ' in bu iki yeni aboneliği olmadan hiçbir yayın almadığını geri çağırın. As expected, the two new subscriptions receive publications, because Sports/FootBall and Sports/Rugby are both cluster topics. QMB forwarded proxy subscriptions for Sports/Football/Arsenal and Sports/Rugby/Leeds to QMA, which then sent the publications to QMB.

Beklenmeyen sonuç, daha önce yayın içermeyen Sports/# ve Sports/#/Leeds adlı aboneliklerin artık yayınları aldığından kaynaklanır. Bunun nedeni, diğer abonelikler için QMB 'e iletilen Sports/Football/Arsenal ve Sports/Rugby/Leeds yayınlarının artık QMB' e bağlı herhangi bir abone için kullanılabilmesinin nedeni. Consequently the subscriptions to the local topics Sports/# and Sports/#/Leeds receive the Sports/Rugby/Leeds publication. Spor/Futbol, JOKER özellik değeri BLOCKolarak ayarlanmış olduğundan,Sports/#/Arsenal bir yayın almamaya devam eder.

Çizelge 5. QMBtarihinde alınan yayınlar			
Abonelik	**Konu dizesi**	**Alınan yayınlar**	**Notlar**
SPORTS	*Sports/#*	*Sports/Rugby/Leeds*	*All publications to Football subtree blocked by WILDCARD(BLOCK) on Sports/Footbal*
SARSENAL	Sports/#/Arsenal	-	Sports/Football üzerindeWILDCARD(BLOCK) , Arsenalüzerinde genel arama karakteri aboneliğini önler
SLEEDS	*Sports/#/Leeds*	*Sports/Rugby/Leeds*	*Sports/Rugby için varsayılan WILDCARD , Leedsüzerinde genel arama karakteri aboneliğini engellememektedir.*
FARSENAL	Sports/Football/Arsenal	Sports/Football/Arsenal	Arsenal , aboneliğin bir genel arama karakterinin olmadığı için bir yayın alır.

Çizelge 5. QMBtarihinde alınan yayınlar (devamı var)			
Abonelik	**Konu dizesi**	**Alınan yayınlar**	**Notlar**
FLEEDS	Sports/Rugby/Leeds	Sports/Rugby/Leeds	Leeds , herhangi bir etkinlikte bir yayın alır.

Çoğu uygulamada, bir aboneliğin başka bir aboneliğin davranışını etkilemesi istenmeyen bir davranışa neden olur. One important use of the JOKER property with the value BLOCK is to make the subscriptions to the same topic string containing wildcards behave uniformly. Aboneliğin yayıncı ile aynı kuyruk yöneticisine mi, yoksa farklı bir aboneliğe mi ait olduğunu, aboneliğin sonuçlarının aynı olduğunu mu?

Joker karakterler ve akışlar

Yayınlama/abone olma API ' ya yazılan yeni bir uygulama için geçerli olan etki, * aboneliğinin yayınsız olarak alınmamasını sağlar. To receive all the Sports publications you must subscribe to Sports/*, or Sports/#, and similarly for Business publications.

Yayınlama/abone olma aracısı IBM WebSphere MQ 7 ' a ve sonraki sürümlere geçirildiğinde, kuyruğa alınan var olan bir yayınlama/abone olma uygulamasının davranışı değişmez. The **StreamName** property in the **Publish**, **Register Publisher**, or **Subscriber** commands is mapped to the name of the topic the stream has been migrated to.

Genel arama karakterleri ve abonelik noktaları

Yayınlama/abone olma API ' ya yazılan yeni bir uygulama için, geçişin etkisi, * aboneliğinin yayınsız olarak alınmamasını sağlar. To receive all the Sports publications you must subscribe to Sports/*, or Sports/#, and similarly for Business publications.

Yayınlama/abone olma aracısı IBM WebSphere MQ 7 ' a ve sonraki sürümlere geçirildiğinde, kuyruğa alınan var olan bir yayınlama/abone olma uygulamasının davranışı değişmez. **Publish**, **Register Publisher**ya da **Subscriber** komutlarındaki **SubPoint** özelliği, aboneliğin yeni düzeye geçirilmiş olduğu konunun adıyla eşlenir.

Örnek: Sport yayınlama/abone olma kümesini yaratın

Aşağıdaki adımlar, dört kuyruk yöneticisi içeren bir küme (CL1) oluşturur: iki tam havuz, CL1A ve CL1B, ve iki kısmi havuz, QMA ve QMB. Tüm havuzlar yalnızca küme tanımlamalarını tutmak için kullanılır. QMA , küme konusu anasistemine atanır. Sürekli abonelikler hem QMA hem de QMBüzerinde tanımlanır.

Not: Örnek, Windowsiçin kodlanmıştır. Örneğin, diğer platformlardaki örneği yapılandırmak ve sınamak için Create qmgrs.bat ve pub.batoluşturma kodunu yeniden kodlamanız gerekir.

1. Komut dosyalarını oluşturun.
   a. topics.tstyarat
   b. Create wildsubs.tst
   c. Create fullsubs.tst
   d. Create qmgrs.bat
   e. oluştur pub.bat
2. Yapılandırmayı oluşturmak için Create qmgrs.bat komutunu çalıştırın.

```
qmgrs
```

Create the topics in Şekil 23 sayfa 76. The script in figure 5 creates the cluster topics Sports/Football and Sports/Rugby.

Not: REPLACE seçeneği, bir konunun TOPICSTR özelliklerini değiştirmez. TOPICSTR , örnekte farklı konu ağaçlarını test etmek için kullanılan bir özeldir. Konuları değiştirmek için önce konuyu silin.

```
DELETE TOPIC ('Sports')
DELETE TOPIC ('Football')
DELETE TOPIC ('Arsenal')
DELETE TOPIC ('Blackburn')
DELETE TOPIC ('Chelsea')
DELETE TOPIC ('Rugby')
DELETE TOPIC ('Leeds')
DELETE TOPIC ('Wigan')
DELETE TOPIC ('Warrington')
DELETE TOPIC ('St. Helens')

DEFINE TOPIC ('Sports')   TOPICSTR('Sports')
DEFINE TOPIC ('Football')  TOPICSTR('Sports/Football') CLUSTER(CL1) WILDCARD(BLOCK)
DEFINE TOPIC ('Arsenal')  TOPICSTR('Sports/Football/Arsenal')
DEFINE TOPIC ('Blackburn') TOPICSTR('Sports/Football/Blackburn')
DEFINE TOPIC ('Chelsea')  TOPICSTR('Sports/Football/Chelsea')
DEFINE TOPIC ('Rugby')   TOPICSTR('Sports/Rugby')  CLUSTER(CL1)
DEFINE TOPIC ('Leeds')   TOPICSTR('Sports/Rugby/Leeds')
DEFINE TOPIC ('Wigan')   TOPICSTR('Sports/Rugby/Wigan')
DEFINE TOPIC ('Warrington') TOPICSTR('Sports/Rugby/Warrington')
DEFINE TOPIC ('St. Helens') TOPICSTR('Sports/Rugby/St. Helens')
```

Şekil 25. Konuları silin ve yaratın: topics.tst

Not: REPLACE konu dizgilerinin yerini değiştirmediği için konuları silin.

Joker karakterler içeren abonelikler oluşturun. Genel arama karakterleri, Şekil 23 sayfa 76içindeki konu nesneleriyle ilgili konulara karşılık gelir. Her abonelik için bir kuyruk yaratın. Kuyruklar temizlenir ve komut dosyası çalıştırıldığında ya da yeniden çalıştırıldığında abonelikler silinir.

Not: REPLACE seçeneği, bir aboneliğin TOPICOBJ ya da TOPICSTR özelliklerini değiştirmez. TOPICOBJ or TOPICSTR are the properties that are usefully varied in the example to test different subscriptions. Bunları değiştirmek için önce aboneliği silin.

```
DEFINE QLOCAL(QSPORTS)  REPLACE
DEFINE QLOCAL(QSARSENAL) REPLACE
DEFINE QLOCAL(QSLEEDS)  REPLACE
CLEAR QLOCAL(QSPORTS)
CLEAR QLOCAL(QSARSENAL)
CLEAR QLOCAL(QSLEEDS)

DELETE SUB  (SPORTS)
DELETE SUB  (SARSENAL)
DELETE SUB  (SLEEDS)
DEFINE SUB  (SPORTS)  TOPICSTR('Sports/#')     DEST(QSPORTS)
DEFINE SUB  (SARSENAL) TOPICSTR('Sports/+/Arsenal') DEST(QSARSENAL)
DEFINE SUB  (SLEEDS)  TOPICSTR('Sports/+/Leeds')  DEST(QSLEEDS)
```

Şekil 26. Genel arama karakteri abonelikleri yarat: wildsubs.tst

Küme konusu nesnelerine başvuruda bulunan abonelikler oluşturun.

Not:

The delimiter, /, is automatically inserted between the topic string referenced by TOPICOBJ, and the topic string defined by TOPICSTR.

DEFINE SUB(FARSENAL) TOPICSTR('Sports/Football/Arsenal') DEST(QFARSENAL) , aynı aboneliği oluşturur. TOPICOBJ , önceden tanımladığınız konu dizgisine hızlı bir şekilde gönderme yapmak için kullanılır. Abonelik, yaratıldığında, artık konu nesnesini ifade etmimez.

```
DEFINE QLOCAL(QFARSENAL) REPLACE
DEFINE QLOCAL(QRLEEDS)  REPLACE
CLEAR QLOCAL(QFARSENAL)
CLEAR QLOCAL(QRLEEDS)

DELETE SUB  (FARSENAL)
DELETE SUB  (RLEEDS)
DEFINE SUB  (FARSENAL) TOPICOBJ('Football') TOPICSTR('Arsenal') DEST(QFARSENAL)
DEFINE SUB  (RLEEDS)  TOPICOBJ('Rugby')  TOPICSTR('Leeds')  DEST(QRLEEDS)
```

Şekil 27. Abonelikleri sil ve yarat: fullsubs.tst

İki havuzu olan bir küme oluşturun. Yayınlama ve abone olma için iki kısmi havuz oluşturun. Her şeyi silmek ve yeniden başlamak için komut dosyasını yeniden çalıştırın. Komut dosyası aynı zamanda konu sıradüzenini ve ilk genel arama karakteri aboneliklerini de yaratır.

Not:

Diğer platformlarda, benzer bir komut dosyası yazın ya da tüm komutları yazın. Bir komut dosyasının kullanılması, her şeyi silmesini ve aynı bir yapılandırmayla yeniden başlatılmasını hızlı bir şekilde sağlar.

```
@echo off
set port.CL1B=1421
set port.CLIA=1420
for %%A in (CL1A CL1B QMA QMB) do call :createQM %%A
call :configureQM CL1A CL1B %port.CL1B% full
call :configureQM CL1B CL1A %port.CL1A% full
for %%A in (QMA QMB) do call :configureQM %%A CL1A %port.CL1A% partial
for %%A in (topics.tst wildsubs.tst) do runmqsc QMA < %%A
for %%A in (wildsubs.tst) do runmqsc QMB < %%A
goto:eof

:createQM
echo Configure Queue manager %1
endmqm -p %1
for %%B in (dlt crt str) do %%Bmqm %1
goto:eof

:configureQM
if %1==CL1A set p=1420
if %1==CL1B set p=1421
if %1==QMA set p=1422
if %1==QMB set p=1423
echo configure %1 on port %p% connected to repository %2 on port %3 as %4 repository
echo DEFINE LISTENER(LST%1) TRPTYPE(TCP) PORT(%p%) CONTROL(QMGR) REPLACE | runmqsc %1
echo START LISTENER(LST%1) | runmqsc %1
if full==%4 echo ALTER QMGR REPOS(CL1) DEADQ(SYSTEM.DEAD.LETTER.QUEUE) | runmqsc %1
echo DEFINE CHANNEL(TO.%2) CHLTYPE(CLUSSDR) TRPTYPE(TCP) CONNAME('LOCALHOST(%3)') CLUSTER(CL1)
REPLACE | runmqsc %1
echo DEFINE CHANNEL(TO.%1) CHLTYPE(CLUSRCVR) TRPTYPE(TCP) CONNAME('LOCALHOST(%p%)')
CLUSTER(CL1) REPLACE | runmqsc %1
goto:eof
```

Şekil 28. Kuyruk yöneticileri yarat: qmgrs.bat

Küme konularına abonelikleri ekleyerek yapılandırmayı güncelleyin.

```
@echo off
for %%A in (QMA QMB) do runmqsc %%A < wildsubs.tst
for %%A in (QMA QMB) do runmqsc %%A < upsubs.tst
```

Şekil 29. Abonelikleri güncelleştir: upsubs.bat

Yayın konusu dizgisini içeren iletileri yayınlamak için, parametre olarak kuyruk yöneticisiyle `pub.bat`komutunu çalıştırın. `Pub.bat` uses the sample program **amqspub**.

```
@echo off
@rem Provide queue manager name as a parameter
set S=Sports
set S=6 Sports/Football Sports/Football/Arsenal
set S=6 Sports/Rugby Sports/Rugby/Leeds
for %%B in (6) do echo %%B | amqspub %%B %1
```

Şekil 30. Yayınla: pub.bat

Akışlar ve konular

Kuyruğa alınan yayınlama/abone olma, tümleştirilmiş yayınlama/abone olma modelinde var olmayan bir yayın akışı kavramına sahip. Kuyruklanan yayınlama/abone olma yolunda akışlar, farklı konulara ilişkin bilgi akışını birbirinden ayırmanın bir yolunu sağlar. From IBM WebSphere MQ 7.0 onwards, a stream is implemented as a top-level topic that can be mapped to a different topic identifier administratively.

Varsayılan akış SYSTEM.BROKER.DEFAULT.STREAM , bir ağdaki tüm aracılar ve kuyruk yöneticileri için otomatik olarak ayarlanır ve varsayılan akışın kullanılması için ek bir yapılandırma gerekmez. Varsayılan akışı, adlandırılmamış bir varsayılan konu alanı olarak düşünün. Varsayılan akışa yayınlanan konular, IBM WebSphere MQ 7.0 ' den başlayarak, kuyruğa alınmış yayınlama/abone olma etkin olan tüm bağlı kuyruk yöneticileri tarafından hemen kullanılabilir olur. Adlandırılan akışlar, adlandırılmış, konu alanları gibi ayrı bir alanlardır. Adı belirtilen akış, kullanıldığı her bir aracıda tanımlanmalıdır.

Yayıncılar ve aboneler farklı kuyruk yöneticileriyse, aracılar aynı aracı sıradüzenine bağlandıktan sonra yayınlar için başka bir yapılandırma gerekmez ve bunlar arasında akışa ilişkin abonelikler de yoktur. Aynı birlikte çalışabilirlik de aynı şekilde devam ediyor.

Adlandırılan akışlar

Kuyruğa alınmış yayınlama/abone olma programlama modeli ile çalışan bir çözüm tasarımcısı, tüm spor yayınlarını Sportadlı bir adlandırılan akışa yerleştirmeye karar verebilir. IBM WebSphere MQ 6.0 içinde bir akış, genellikle model kuyruğunu kullanan diğer aracılarda otomatik olarak eşlenir, SYSTEM.BROKER.MODEL.STREAM. However, for the stream to be available to a queue manager that runs on IBM WebSphere MQ 7.0 onwards with queued publish/subscribe enabled, the stream must be added manually.

Queued publish/subscribe applications that subscribe to Soccer/Results on stream Sport work without change. Integrated publish/subscribe applications that subscribe to the topic Sport using MQSUB, and supplying the topic string Soccer/Results receive the same publications too.

Akış ekleme görevi, Akış eklenmesibaşlıklı konuda açıklanmaktadır. İki nedenden dolayı akımları el ile eklemeniz gerekebilir.

1. Uygulamaları tümleşik yayınlama/abone olma MQI arabirimine geçirmek yerine, daha sonraki sürüm kuyruğu yöneticilerindeki kuyruklanmış yayınlama/abone olma uygulamalarınızı geliştirmeye devam edebilirsiniz.
2. Akışların konuları varsayılan olarak eşleştirmesi, konu alanında "çarpışma" a yol açar ve bir akıştaki yayınların konu dizgisinin başka bir yerden yayınlarla aynı olması gerekir.

Yetkiler

Varsayılan olarak, konu ağacının kökünde birden çok konu nesnesi vardır: SYSTEM.BASE.TOPIC, SYSTEM.BROKER.DEFAULT.STREAMve SYSTEM.BROKER.DEFAULT.SUBPOINT. Authorities (for example, for publishing or subscribing) are determined by the authorities on the SYSTEM.BASE.TOPIC ; any authorities on SYSTEM.BROKER.DEFAULT.STREAM or SYSTEM.BROKER.DEFAULT.SUBPOINT are ignored. SYSTEM.BROKER.DEFAULT.STREAM ya da SYSTEM.BROKER.DEFAULT.SUBPOINT silinir ve boş olmayan bir konu dizgisiyle yeniden yaratılırsa, bu nesneler üzerinde tanımlanan yetkiler, normal bir konu nesnesiyle aynı şekilde kullanılır.

Akışlar ve konular arasında eşleme

Kuyruğa alınmış bir yayınlama/abone olma akışı, IBM WebSphere MQ 7.0 ' ta bir kuyruk yaratılarak ve akıyla aynı adı verilerek taklit edilir. Bazen kuyruğun adı akış kuyruğu olarak adlandırılır, çünkü bu şekilde kuyruğa yollanan yayınlama/abone olma uygulamaları söz edilir. The queue is identified to the publish/subscribe engine by adding it to the special namelist called SYSTEM.QPUBSUB.QUEUE.NAMELIST. Ad listesine ek özel kuyruklar ekleyerek, gereksinim duyarken istediğiniz sayıda akış ekleyebilirsiniz. Son olarak, akışlar ve akış adıyla aynı konu dizgileriyle aynı adı taşıyan konular eklemeli ve bu konuları yayınlayabilmeniz ve konulara abone olmanız gerekir.

Ancak, istisnai durumlarda, konuları tanımlarken seçtiğiniz konulara ilişkin konulara karşılık gelen konuları da verebilirsiniz. Konu dizgisinin amacı, konuya konu alanında benzersiz bir ad vermeğidir. Genellikle akış adı bu amaca mükemmel bir şekilde hizmet eder. Bazen, bir akış adı ve var olan bir konu adı çarpıştırır. Sorunu çözmek için akımla ilişkili konu için başka bir konu dizesi seçin. Herhangi bir konu dizgisi seçin; bunun benzersiz olmasını sağlayın.

Konu tanımında tanımlanan konu dizgisinin MQOPEN ya da MQSUB MQI çağrıları kullanılarak yayıncılar ve aboneler tarafından sağlanan konu dizgisine olağan şekilde önekli olması gerekir. Konu nesnelerini kullanan konulara başvuran uygulamalar, önek konusu dizgisinin seçiminden etkilenmez; bu nedenle, yayınların konu alanında benzersiz olmasını sağlayan herhangi bir konu dizesini seçebilirsiniz.

Farklı konulara ilişkin farklı akışların yeniden eşlenmesi, konu dizgileri için kullanılan önekleri, bir konu kümesini diğerinden ayırmak için kullanılan önekleri kullanır. Eşlemenin çalışması için rivayla uyulması gereken evrensel bir konu adlandırma kuralı tanımlamanız gerekir.

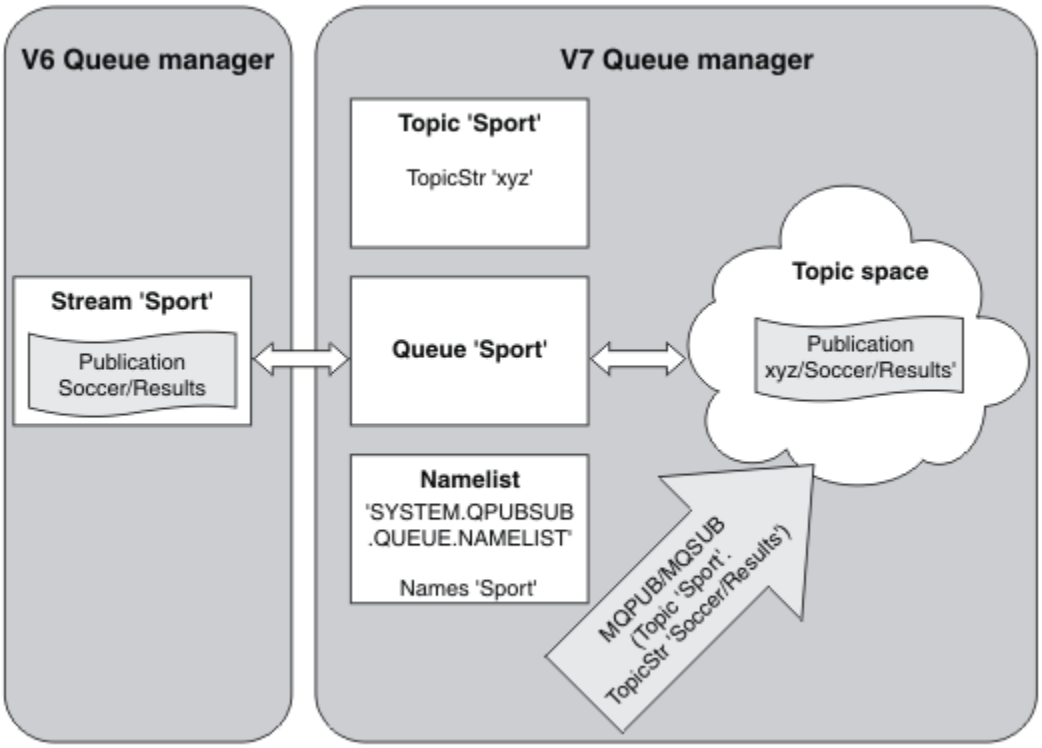
IBM WebSphere MQ 7.0' da, konu dizgileri çarpıştıysa, konu alanlarını ayırmak için akışlar kullanabilirsiniz.

IBM WebSphere MQ 7.0 ' den başlayarak, bir konu dizgisini konu alanındaki başka bir yere yeniden eşlemek için önceden düzeltme düzeneğini kullanıyorsunuz.

Not: Bir akışı sildiğinizde, önce akıştaki tüm abonelikleri silin. Aboneliklerin herhangi biri, aracı sıradüzenindeki diğer aracılardan kaynaklanırsa, bu işlem en önemlidir.

Örnek

In Şekil 31 sayfa 84, topic 'Sport' has the topic string 'xyz' resulting in publications that originate from stream 'Sport' being prefixed with the string 'xyz' in the version 7 queue manager topic space. Publishing or subscribing in version 7 to the topic 'Sport' prefixes 'xyz' to the topic string. Yayın bir IBM WebSphere MQ 6 abonesine akıyorsa, 'xyz' öneki yayından kaldırılır ve 'Sport' akımında yerleştirilir. Bunun tersine, bir yayın IBM WebSphere MQ 6 'dan IBM WebSphere MQ 7' a, 'Sport' akışından 'Sport' konusuna geldiğinde, konu dizgisine 'xyz' öneki eklenir.



Şekil 31. IBM WebSphere MQ 7 konuları ile birlikte var olanIBM WebSphere MQ 6 akışları

Abonelik noktaları ve konular

Adlandırılmış abonelik noktaları, konular ve konu nesneleri tarafından öykünülmektedir.

Abonelik noktalarını el ile eklemek için Abonelik noktası eklenmesibaşlıklı konuya bakın.

Subscription points in IBM MQ

IBM MQ , abonelik noktalarını IBM MQ konu ağacındaki farklı konu alanlarına eşler. Bir abonelik noktası olmayan komut iletilerindeki konular, değiştirilmeden IBM MQ konu ağacının köküne eşlenir ve özellikleri `SYSTEM.BASE.TOPIC`' den devralır.

Bir abonelik noktası içeren komut iletileri, `SYSTEM.QPUBSUB.SUBPOINT.NAMELIST`içindeki konu nesnelerinin listesini kullanarak işleme devam eder. Komut iletisinde abonelik noktası adı, listedeki konu nesnelerinin her biri için konu dizgisine göre eşleştirilir. Bir eşleşme bulunursa, bir konu düğümü olarak, abonelik noktası adı konu dizgisine önceden işaretlenmiş olur. Konu, özelliklerini `SYSTEM.QPUBSUB.SUBPOINT.NAMELIST`içinde bulunan ilişkili konu nesnesinden edinir.

Abonelik noktalarını kullanmanın etkisi, her bir abonelik noktası için ayrı bir konu alanı yaratmasıdır. Konu alanı, abonelik noktasıyla aynı adı taşıyan bir konuya kök saldı. Her konu alanındaki konular, özelliklerini, abonelik noktasıyla aynı adı taşıyan konu nesnesinden devralır.

Any properties not set in the matching topic object are inherited, in the normal fashion, from `SYSTEM.BASE.TOPIC`.

Var olan kuyruğa alınmış yayınlama/abone olma uygulamaları, `MQRFH2` ileti üstbilgilerini kullanarak, `Publish` ya da `Register subscriber` komut iletilerinde **`SubPoint`** özelliğini ayarlayarak çalışmaya devam eder. Abonelik noktası, komut iletisinde konu dizgisiyle birleştirilir ve sonuçta ortaya çıkan konu herhangi bir diğer konu gibi işlenir.

IBM WebSphere MQ 7.0ya da sonraki bir sürümü, uygulamalar abonelik noktalarından etkilenmez. Bir uygulama, bilgileri eşleşen konu nesnelerinden birinden edinen bir konu kullanıyorsa, bu uygulama eşleşen abonelik noktasını kullanarak kuyruğa alınan bir uygulamayla birlikte çalışır.

Örnek

An existing WebSphere Message Broker (now known as IBM Integration Bus) publish/subscribe application that was migrated to IBM MQ created two topic objects, GBP and USD, with the corresponding topic strings `'GBP'` and `'USD'`.

Existing publishers to the topic `NYSE/IBM/SPOT`, migrated to run on IBM MQ, that use the subscription point USD create publications on the topic `USD/NYSE/IBM/SPOT`. Similarly existing subscribers to `NYSE/IBM/SPOT`, using the subscription point USD create subscriptions to `USD/NYSE/IBM/SPOT`.

Subscribe to the dollar spot price in a IBM WebSphere MQ 7.0, or later, publish/subscribe program by calling MQSUB. Create a subscription using the USD topic object and the topic string `'NYSE/IBM/SPOT'`, as illustrated in the 'C' code fragment.

```
strncpy(sd.ObjectName, "USD", MQ_TOPIC_NAME_LENGTH);
sd.ObjectString.VSPtr = "NYSE/IBM/SPOT";
sd.ObjectString.VSLength = MQVS_NULL_TERMINATED;
MQSUB(Hconn, &sd, &Hobj, &Hsub, &CompCode, &Reason);
```

1. Küme konusu ana makinesinde USD ve GBP konu nesnelerinin CLUSTER özniteliğini ayarlayın.
2. Kümedeki diğer kuyruk yöneticilerindeki USD ve GBP konu nesnelerinin tüm kopyalarını silin.
3. Make sure that USD and GBP are defined in `SYSTEM.QPUBSUB.SUBPOINT.NAMELIST` on every queue manager in the cluster.

Tek bir kuyruk yöneticisi yayınlaması/abone olma yapılandırması örneği

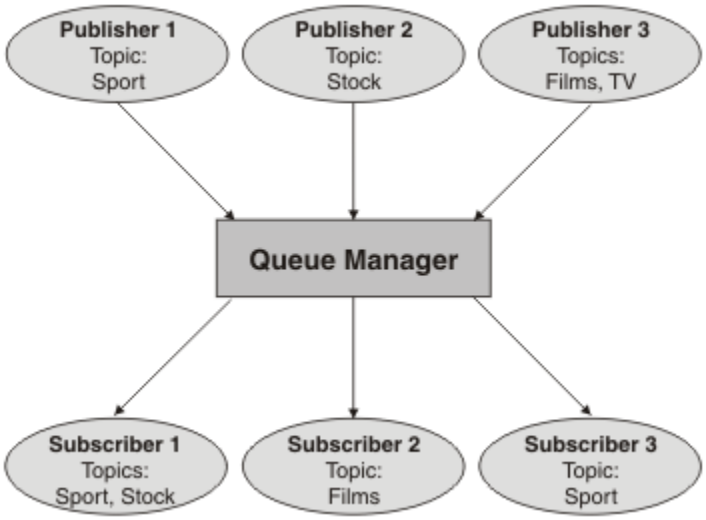
Şekil 32 sayfa 86 , temel tek bir kuyruk yöneticisi yayınlama/abone olma yapılandırmasını gösterir. Bu örnek, yayıncılardan birkaç konu hakkında bilgilerin bulunduğu bir haber hizmetine ilişkin yapılandırmayı gösterir:

- Yayıncı 1 Spor konusunu kullanarak spor sonuçlarıyla ilgili bilgileri yayınlıyor
- Yayıncı 2, hisse senedi fiyatlarıyla ilgili bilgileri Stoğun bir konusu kullanarak yayınlıyor
- Yayıncı 3, Filmler konusu kullanılarak film eleştirileri hakkında bilgi yayınlıyor ve televizyon listeleriyle ilgili bir konuyu yayınlıyor.

Üç abone farklı konulara ilgi gösterdiler, bu nedenle kuyruk yöneticisi ilgilendikleri bilgileri şu bilgileri gönderir:

- Abone 1 spor sonuçlarını ve hisse senedi fiyatlarını alır
- Abone 2, film incelemelerini alır
- Abone 3, spor sonuçlarını alır

Abonelerin hiçbiri televizyon listelerine ilgi göstermedi, bu yüzden bunlar dağıtılmadı.



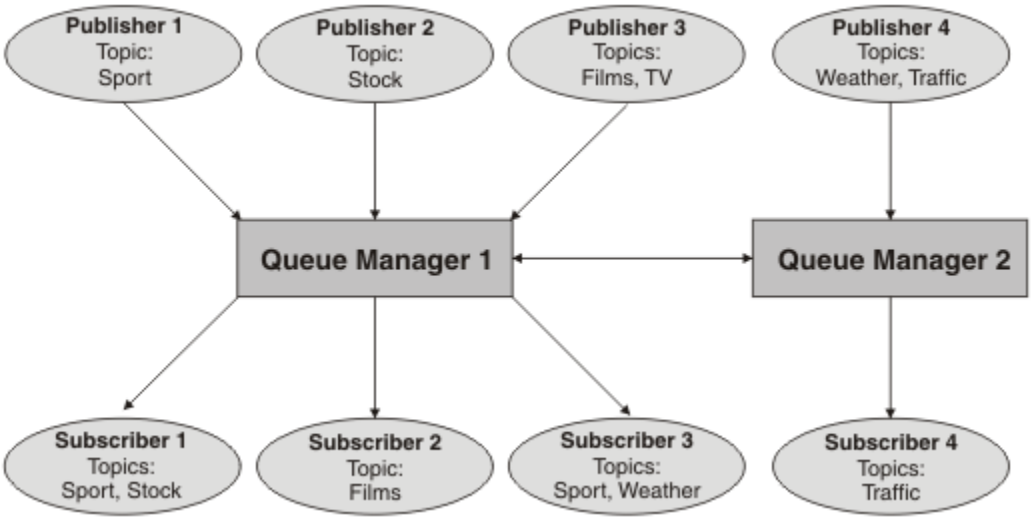
Şekil 32. Tek kuyruk yöneticisi yayınlama/abone olma örneği

Dağıtılmış yayınlama/abone olma ağları

Her kuyruk yöneticisi, bu konuya abone olan yerel olarak yaratılan aboneliklerle bir konuya yayınlanan iletilerle eşleşir. Bir kuyruk yöneticisi ağı yapılandırabilir; böylece, bir kuyruk yöneticisine bağlı bir uygulama tarafından yayınlanan iletiler ağdaki diğer kuyruk yöneticilerinde yaratılan aboneliklere uygun olarak teslim edilir. Bu, kuyruk yöneticileri arasında basit kanallar üzerinden ek yapılandırma gerektirir.

Dağıtılmış yayınlama/abone olma yapılanışı, birbirine bağlı bir kuyruk yöneticilerinden oluşan bir kümedir. Kuyruk yöneticilerinin tümü aynı fiziksel sistemde olabilir ya da birkaç fiziksel sistem üzerinden dağıtılabilirler. Kuyruk yöneticilerini bir araya bağlarken, aboneler bir kuyruk yöneticisine abone olabilir ve başlangıçta başka bir kuyruk yöneticisininde yayınlanan iletileri alabilirsiniz. Bunu göstermek için, aşağıdaki şekil, “Tek bir kuyruk yöneticisi yayınlaması/abone olma yapılandırması örneği” sayfa 85içinde açıklanan yapılandırmaya ikinci bir kuyruk yöneticisi ekler.

- Kuyruk yöneticisi 2, yayınlayıcı 4 tarafından hava durumu tahmini bilgilerini yayınlamak için, Hava durumu konusunu ve trafik durumu hakkında trafik durumunu kullanarak, trafik durumu hakkında bilgi yayınlayabilirsiniz.
- Abone 4, bu kuyruk yöneticisini kullanır ve trafik koşulları ile ilgili bilgileri konu trafiğini kullanarak bilgilendirir.
- Abone 3, yayınlayıcıdan farklı bir kuyruk yöneticisi kullansa da hava durumu koşullarına ilişkin bilgileri de abone eder. Bunun nedeni, kuyruk yöneticilerinin birbiriyle bağlantılı olması olabilir.

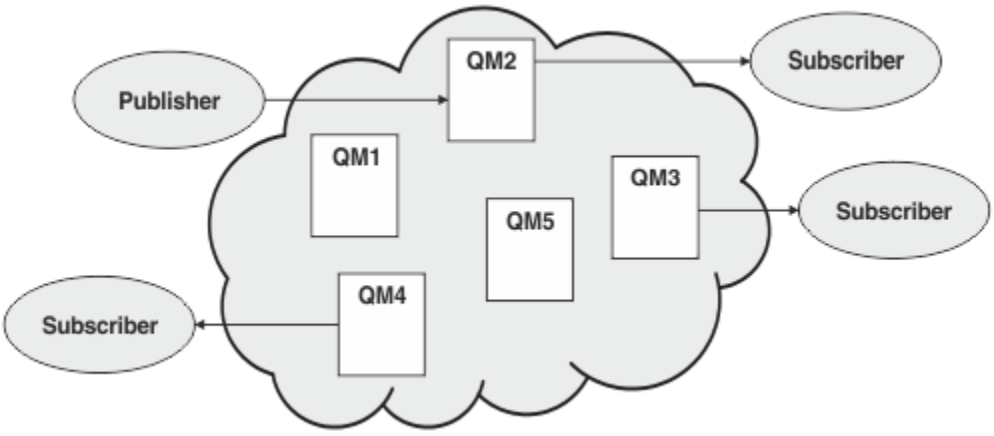


Şekil 33. İki kuyruk yöneticisi içeren yayınlama/abone olma örneği

Bir üst ve alt sıradüzende kuyruk yöneticilerini el ile bağlayabilir ya da bir yayınlama/abone olma kümesi yaratabilir ve IBM MQ , bağlantı ayrıntılarının büyük kısmını sizin için tanımlamanıza olanak sağlar. Örneğin, bir sıradüzende birden çok kümeyi birleştirerek, örneğin her iki topolojiyi de birlikte kullanabilirsiniz.

Yayınlama/abone olma kümelerine genel bakış

Bir yayınlama/abone olma kümesi, kümeye bir ya da daha fazla konu nesnesi eklenmiş olan standart bir kümedir. Bir kümedeki herhangi bir kuyruk yöneticisinde bir yönetim konusu nesnesi tanımladığınızda ve bir küme adı belirterek bu konu nesnesini kümelendiğinizde, bu konuya yayıncılar ve aboneler, kümedeki kuyruk yöneticilerinden herhangi birine bağlanabilir ve yayınlanan iletiler, kuyruk yöneticileri arasındaki küme kanallarının üzerinden abonelere yönlendirilir.



Şekil 34. Yayınlama/abone olma kümesi

Yayınlama/abone olma iletilerinin bir kümede nasıl yönlendirileceğini yapılandırmanın iki yolu vardır:

- doğrudan yönlendirme
- konu anasistem yöneltmesi

Doğrudan yönlendirilen kümelenmiş bir konuyu yapılandırdığınızda, bir kuyruk yöneticisinde yayınlanan iletiler, kümedeki diğer herhangi bir kuyruk yöneticisinde bulunan her aboneliğe doğrudan o kuyruk yöneticisinden gönderilir. Bu, yayınlar için en doğrudan yolu sağlayabilir, ancak bir kümedeki tüm kuyruk yöneticilerine diğer tüm kuyruk yöneticilerinden, bunların arasında kurulan küme kanallarının her birinin farkına varılmasına neden olur.

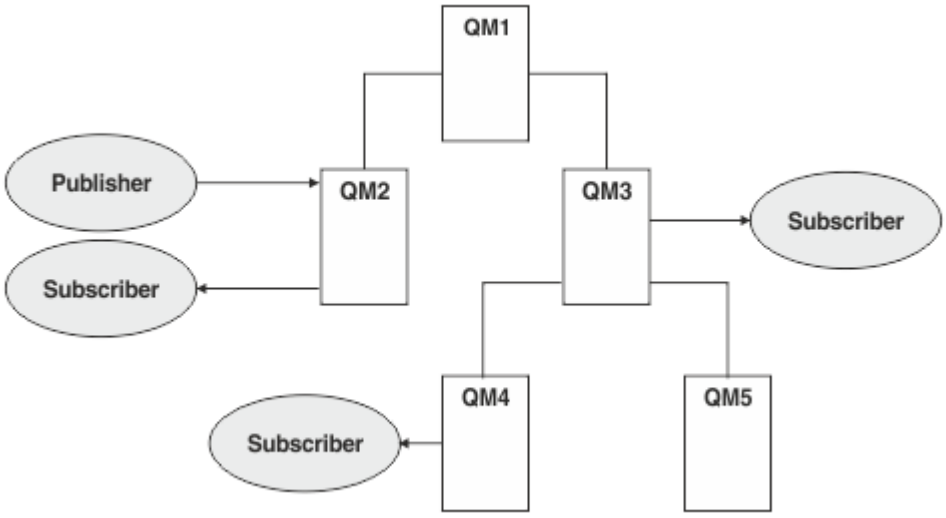
Konu anasistem yöneltmesi kullandığınızda, bir kuyruk yöneticisinden yayınlanan iletiler, denetlenen konu nesnesinin tanımlamasını barındıran bir kuyruk yöneticisine gönderilir. Bu *konu anasistem kuyruk yöneticisi* , iletiyi kümedeki diğer herhangi bir kuyruk yöneticisinde her aboneliğe yönelir. Yayıncılar ya da aboneler, konu ana makine kuyruk yöneticilerinde yer almıyorsa, bu, yayınlar için daha uzun bir rota ile sonuçlanır. Ancak, yalnızca konu anasistem kuyruk yöneticilerinin kümedeki diğer tüm kuyruk yöneticilerinden haberdar olmaları ve bunlarla birlikte oluşturulmuş küme kanallarının olması yararlı olur.

Daha fazla bilgi için "Kümeleri yayınla/abone ol" sayfa 89 başlıklı konuya bakın.

Yayınlama/abone olma sıradüzenlerine genel bakış

Yayınlama/abone olma sıradüzeni, kanallarla sıradüzensel bir yapıya bağlanan bir kuyruk yöneticisi kümesidir. Each queue manager identifies its *üst öğe* queue manager, as described in Bir kuyruk yöneticisini yayınlama/abone olma sıradüzenine bağlama.

Bir konuya ilişkin yayınlayıcılar ve aboneler, sıradüzendeki herhangi bir kuyruk yöneticisine ve bunların arasında sıradüzenli kuyruk yöneticisi bağlanırlığı kullanılarak ileti akışı arasında bağlantı kurabilirler.



Şekil 35. Yayınlama/abone olma sıradüzeni

Önceki şekilde, QM3 ve QM4 üzerindeki abonelere teslim edilen yayınlar QM2 'den QM1 ' e ve ardından QM3 ve son olarak da QM4' ye yönlendirildi.

Sıradüzenler, sıradüzendeki her kuyruk yöneticisi arasındaki ilişkiler üzerinde size doğrudan denetim sağlar. Bu, yayıncılardan abonelere iletilerin yönlendirilmesi üzerinde ince parçalı denetime olanak tanır ve özellikle kısıtlı bağlanırlık içeren kuyruk yöneticisi ağları arasında yönlendirme yaparken yararlı olur. Bir iletinin yayınlayıcıdan abonelere doğru yöneltildiği her kuyruk yöneticisinin kullanılabilirliğini ve yeteneğini dikkatli bir şekilde dikkate almalısınız.

Daha fazla bilgi için "Sıradüzenleri yayınlama/abone olma" sayfa 91 başlıklı konuya bakın.

Kuyruk yöneticileri arasında yayın dağıtımı

Yöneltme tercihlerine ek olarak, yayınları bir kuyruk yöneticisi ağı boyunca dağıtmaya ilişkin iki yaklaşım vardır:

- Yayınları yalnızca bir kuyruk yöneticisinden, o yayın için abone olan kuyruk yöneticilerine gönderin.
- Her yayını tüm kuyruk yöneticilerine gönderin ve aboneliklerine göre eşleştirmelerine izin verin.

Eski sonuçlar yayın iletilerinde yalnızca gerekli olduğu durumlarda gönderilir, ancak kuyruk yöneticileri arasında bir abonelik bilgisi düzeyinin paylaşılmasını gerektirir. İkincisi, abonelik bilgisinin paylaşılmasını gerektirmez, ancak kuyruk yöneticileri arasında gönderilmekte olan gereksiz yayın iletilerine neden olabilir.

Varsayılan olarak, IBM MQ eski yöntemi kullanır; bu yöntem, yayınların yalnızca kendileri için abonelikleri olan kuyruk yöneticilerine gönderilmektedir. The subscription knowledge is propagated between queue managers in the form of *yetkili abonelikler*. Bu, dağıtılmış bir yayınlama/abone olma topolojisinde kullanmak için en verimli olan, aboneliklerin dağılımına ve yaşam sürmesine ve yayınların sıklığına bağlıdır. Bkz. Yayınlama/abone olma ağlarındaki abonelik performansı.

İlgili kavramlar

"Konu ağaçları" sayfa 72
Tanımladığınız her konu, konu ağacında bir öğedir ya da düğümdür. Konu ağacı, önceden MQSC ya da PCF komutlarını kullanarak önceden tanımlanmış konuları içeren ya da içerebilecek boş olabilir. Yeni bir konuyu, konu yaratma komutlarını kullanarak ya da bir yayınlama ya da abonelikte ilk kez konuyu belirterek tanımlayabilirsiniz.

İlgili görevler

Yayınlama/abone olma kümelerini tasarlama

İlgili başvurular

Sıradüzeni senaryolarını yayınla/abone ol

Kümeleri yayınla/abone ol

Bir yayınlama/abone olma kümesi, birbirine bağlı kuyruk yöneticilerinden oluşan standart bir kümedir; bu, yayınların otomatik olarak yayınlama uygulamalarından kümedeki herhangi bir kuyruk yöneticisinde var olan aboneliklere taşındığı bir kümedir. Yayınlama/abone olma kümesinde yayın yönlendirmesi için iki seçenek vardır: *doğrudan yönlendirme* ve *konu anasistem yöneltmesi*. Seçtiğiniz yönlendirme, kümenize ilişkin boyut ve beklenen etkinlik kalıplarına bağlıdır.

Yayınlama/abone olma ileti sistemi için kullanılan bir küme, standart bir IBM MQ kümesinden farklı değildir. Bu nedenle, yayınlama/abone olma kümesi içindeki kuyruk yöneticileri fiziksel olarak ayrı bilgisayarlarda var olabilir ve her kuyruk yöneticisi çifti gerektiğinde otomatik olarak küme kanallarıyla birbirine bağlanır. Ek bilgi için Kümelerbaşlıklı konuya bakın.

Yayınlama/abone olma ileti alışverişi için kuyruk yöneticisi standart bir kümesini yapılandırmak için, kümedeki bir kuyruk yöneticisinde bir ya da daha çok yönetilen konu nesnesi tanımlayabilirsiniz. Konuyu bir küme konusu yapmak için, **CLUSTER** özelliğini kümenin adıyla yapılandırırdınız. Bunu yaparken, konu ağacı ' un o noktasında ya da altında bir yayıncı ya da abone tarafından kullanılan herhangi bir konu kümedeki tüm kuyruk yöneticilerinde paylaşılır ve konu ağacının kümelenmiş bir dalına yayınlanan iletiler otomatik olarak kümedeki diğer kuyruk yöneticilerindeki aboneliklere yönlendirilir.

Her iletinin tek bir kopyası, yayıncı kuyruk yöneticisi ile hedef kuyruk yöneticiliklerindeki iletinin abone sayısından bağımsız olarak, diğer kuyruk yöneticilerinden her biri arasında gönderilir. Bir ya da daha çok abonelik içeren bir kuyruk yöneticisine varışta, ileti tüm aboneliklerde yineleniyor.

Kümeye katılan herhangi bir kuyruk yöneticisi, kümelenmiş konulardan otomatik olarak haberdar olur ve o kuyruk yöneticisinde bulunan yayıncılar ve aboneler otomatik olarak kümeye katılır.

Kümeli olmayan yayınlama/abone olma etkinliği, kümelenmiş bir konu nesnesinin altına düşmeyen konu dizgileriyle çalışılarak yayınlama/abone olma kümesinde de yer alabilir.

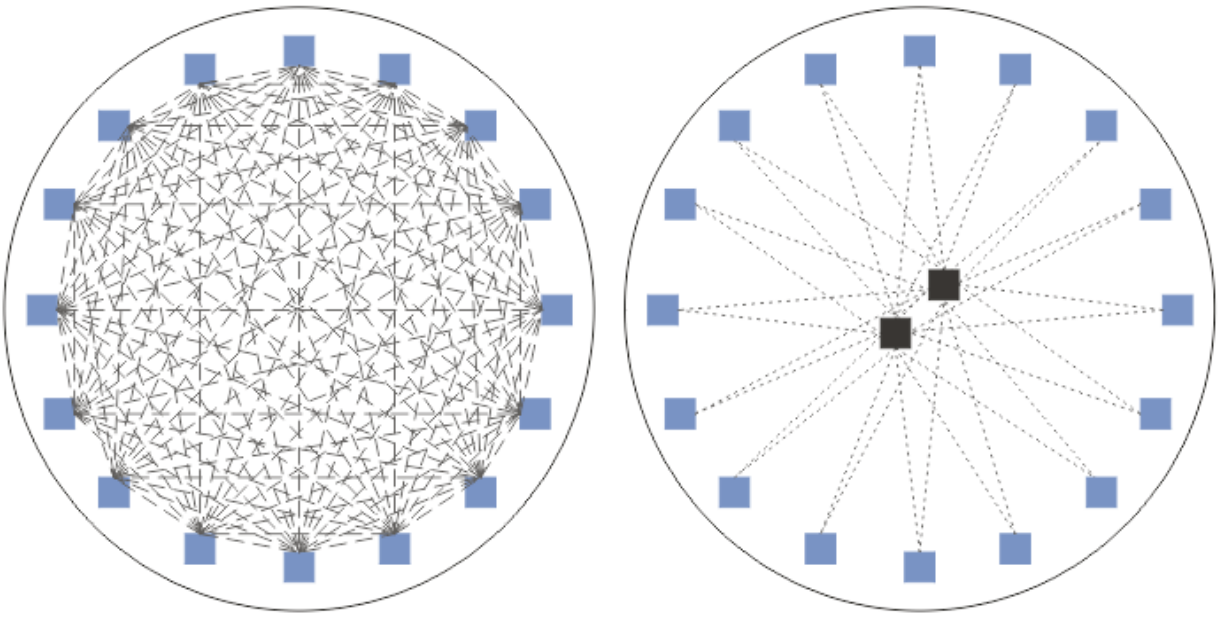
Yayınlama/abone olma kümesinde yayın yönlendirmesi için iki seçenek vardır: *doğrudan yönlendirme* ve *konu anasistem yöneltmesi*. Küme içinde kullanılacak ileti yönlendirmesini seçmek için, denetlenen konu nesnesindeki **CLROUTE** özelliğini aşağıdaki değerlerden birine ayarladınız:

- **DIRECT**
- **TOPICHOST**

Varsayılan olarak, konu yöneltme **DIRECT**' dir. Bu, IBM MQ 8.0' den önceki tek seçenekti. Bir kuyruk yöneticisinde doğrudan yöneltilen kümelenmiş bir konu yapılandırdığınızda, kümedeki tüm kuyruk yöneticileri kümedeki diğer tüm kuyruk yöneticilerini bilir. Yayınlama ve abone olma işlemleri gerçekleştirirken, her kuyruk yöneticisi kümedeki başka bir kuyruk yöneticisine doğrudan bağlanabilir.

IBM MQ 8.0' dan bunun yerine konu yönlendirmesini **TOPICHOST**olarak yapılandırabilirsiniz. Konu anasistemi yöneltme özelliğini kullandığınızda, kümedeki tüm kuyruk yöneticileri, yöneltilen konu

tanımlamasını barındıran küme kuyruğu yöneticilerini (yani, konu nesnesini tanımladığınız kuyruk yöneticilerini) tanır. Yayınlama ve abone olma işlemleri gerçekleştirirken, kümedeki kuyruk yöneticileri doğrudan birbirine değil, yalnızca bu konu anasistem kuyruk yöneticilerine bağlanır. Konu anasistem kuyruk yöneticileri, yayınların yayınlandığı kuyruk yöneticilerinden, eşleşen aboneliklere sahip kuyruk yöneticilerine yönlendirmekten sorumludur.



Şekil 36. Doğrudan yöneltme ve konu anasistem yöneltmesi

Doğrudan yönlendirmeye genel bakış

Yönetilen bir konu nesnesi doğrudan yöneltme için yapılandırıldığında, konu nesnesinin yalnızca, tüm kuyruk yöneticileri için kümedeki kuyruk yöneticilerinden birinde tanımlanabilmesinin gerekli olması gerekir. Bu konunun tanımlandığı kuyruk yöneticisi seçimi, konu için yayınlama/abone olma ileti alışverişi davranışını etkilemez.

Her ileti doğrudan yayınlayıcı kuyruk yöneticisinden, herhangi bir ara kuyruk yöneticisinden geçmeyen, kümedeki diğer kuyruk yöneticilerindeki her bir aboneliğe akar.

Varsayılan olarak, iletiler yalnızca, kümedeki bir ya da daha çok aboneliği barındıran kümedeki diğer kuyruk yöneticilerine gönderilir.

- Bu, her bir kuyruk yöneticisine dayalı olarak, bir ya da daha fazla aboneliği olan tüm konuların kümesinde bulunan diğer tüm kuyruk yöneticilerini doğrudan bilgilendirir. Bu, kümedeki tüm kuyruk yöneticilerinde, abone olunan tüm konuların farkında olması ve diğer tüm kuyruk yöneticilerine bir kanal kurma aboneliği barındıran kuyruk yöneticilerinden haberdar olması gerekir. Bu, her kuyruk yöneticisinin yayıncısı olup olmadığını bağımsız olarak içerir.
- Tüm kuyruk yöneticilerine abone olan her bir konuya ilişkin bilgi, abonelikleri olup olmamalarından bağımsız olarak, tüm yayınları kümedeki tüm kuyruk yöneticilerine göndermenin bir modeline çevrilerek kaldırılabilir. Bu, abonelik bilgisi trafiğini azaltır, ancak yayın trafiğini ve her bir kuyruk yöneticisinin oluşturduğu kanal sayısını artırması olası. Bkz. Yayınlama/abone olma ağlarındaki abonelik performansı.

Doğrudan yöneltilen kümelenmiş konuları kullanarak yayınlama/abone olma ileti akışları, her kümeden bir yayınlama/abone olma sıradüzenine bir kuyruk yöneticisi ekleyerek birden çok yayınlama/abone olma kümesine yayılabilir. Bkz. Birden çok kümenin konu alanlarının birleştirmesi.

Doğrudan yönlendirmeye ilişkin daha ayrıntılı bir keşif için bkz. Yayınlama/abone olma kümelerine doğrudan yönlendirme.

Konu anasistem yöneltmesine genel bakış

Konu anasistemi yöneltmesi için denetlenen bir konu nesnesi yapılandırıldığında, kümedeki bir kuyruk yöneticisinden gelen yayınlar, konu nesnesinin yapılandırıldığı bir kuyruk yöneticisinden ("konu anasistemi") ve buradan, aboneliklerin bulunduğu kuyruk yöneticilerine yönlendirilir.

- Bu, her bir kuyruk yöneticisine bir ya da daha fazla aboneliği olan her konunun ana anasistemlerini bildiren her bir kuyruk yöneticisine dayanır. Bir aboneliğin bulunduğu kuyruk yöneticisi, aboneliğin ilişkilendirdiği konu için her konu ana makineye bir kanal kurar.
- Konu dışı barındırma kuyruğu yöneticileri, yayınlama/abone olma amaçları için kümedeki diğer konu dışı barındırma kuyruğu yöneticilerinden haberdar değildir ve bu amaçla kanallar arasında kanal oluşturulmaz.
- Yayınlama uygulaması, konuyu barındıran bir kuyruk yöneticisine bağlıysa, yayınlanan iletiler, ek bir 'sekme' gerekmeden, eşleşen aboneliklerin oluşturulduğu kuyruk yöneticilerine doğrudan yöneltilir. Benzer şekilde, eşleşen abonelikler tek kuyruk yöneticisininde yaratıldıysa, o konuya ilişkin yayınlanan iletiler ek bir sekme gerektirmeden, doğrudan o kuyruk yöneticisine yöneltilir.
- Yayınlayıcı ile aynı kuyruk yöneticisine ilişkin abonelikler, ilk yayınları konu nesnesinin anasistemlerine yöneltmeden memnun olur.

Kümelenmiş kuyruklar için olduğu gibi, birden çok kuyruk yöneticisi aynı denetim konusu nesnesini yapılandırabilir. Bu, ileti yönlendirmesinin daha yüksek düzeyde kullanılabilirliğini ve iş yükü dengelemesi yoluyla yatay ölçeklemeyi sağlar. Konu ana makinesi konu nesnelerine ilişkin olarak, birden çok kuyruk yöneticisi aynı adlı konuyu konu ağacının aynı dalı için yapılandırırken, her konu anasistemi abone olunan her kuyruk yöneticisi tarafından abone olunan konulardan haberdar olur.

- Bir ileti yayınlandığında, abonelik barındırma kuyruğu yöneticilerine iletilecek anasistem kuyruğu yöneticilerinden biri için bu ileti gönderilir. Konu anasistem kuyruk yöneticisinin seçimi, kümelenmiş noktadan noktaya kuyruklar için aynı varsayılan iş yükü dengeleme kurallarını izler.
- Bir ya da daha çok konu anasistem kuyruğu yöneticisine bir yayınlama kuyruk yöneticisi ile iletişim kurulamazsa, iletiler geri kalan kullanılabilir konu barındıran kuyruk yöneticilerine yöneltilir.

Kümenin yönlendirilmiş bir dalındaki bir konuya ilişkin her yayın, kümenin herhangi bir yerinde herhangi bir abonelik olmasa da, konu anasistemlerinden birine iletilir. Varsayılan olarak, iletiler buradan yalnızca bir ya da daha çok aboneliği barındıran kümedeki diğer kuyruk yöneticilerine gönderilir.

- Bu, her bir konu anasistemi kuyruk yöneticisinin, kümedeki her kuyruk yöneticisinde abone olunan tüm konu dizgilerinden haberdar olması gerekir.
- Abone olunan her konuya ilişkin bilgi, abonelikleri olup olmamalarından bağımsız olarak, kümedeki tüm kuyruk yöneticilerine yöneltilen tüm yayınları bir konu anasistemine yöneltilen bir modele çevrilerek kaldırılabilir. Bu, abonelik bilgisi trafiğini azaltır, ancak yayın trafiğini artırır ve kuyruk yöneticisini barındıran her bir konu ile oluşturulan kanal sayısını artırır. Bkz. Yayınlama/abone olma ağlarındaki abonelik performansı.

Publish/subscribe message flows using topic host routed clustered topics **olamaz** span multiple publish/subscribe clusters through the use of a publish/subscribe hierarchy.

Konu anasistem yöneltmesi hakkında daha ayrıntılı bir keşif için Yayınlama/abone olma kümelerinde konu anasistem yönlendirmesibaşlıklı konuya bakın.

Sıradüzenleri yayınlama/abone olma

Kuyruk yöneticilerini kanallar kullanarak birbirine bağlayarak bir yayınlama/abone olma hiyerarşisi oluşturun ve daha sonra, kuyruk yöneticisi çiftleri arasında bir alt-üst öğe ilişkisi tanımlayın. Bir yayıncıdan aboneliklere, bir sıradüzendeki doğrudan ilişkiler üzerinden bir ileti akışları. Bunun, birden çok "sekmenin" oraya varması anlamına gelebilir.

Hedef kuyruk yöneticisinindeki iletinin abone sayısından bağımsız olarak, bir kuyruk yöneticisi çifti arasında iletinin yalnızca bir kopyası gönderilir. Bir ya da daha çok abonelik içeren bir kuyruk yöneticisine varışta, ileti tüm aboneliklerde yineleniyor.

Varsayılan olarak, iletiler yalnızca, başka bir kuyruk yöneticisinde bulunan bir aboneliğe ilişkin sıradüzenindeki diğer kuyruk yöneticilerine gönderilir:

- Bu, her kuyruk yöneticisine, şu anda bu kuyruk yöneticisinde ya da diğer ilişkilerinden birinde bir ya da daha fazla aboneliği olan tüm konularla ilgili olarak her bir doğrudan ilişki hakkında bilgi veren bir yöneticiyle ilgilidir. Bu, sıradüzendeki tüm kuyruk yöneticilerinde, abone olunmakta olan tüm konuların farkında olma sonucu ortaya çıktı.
- Bu davranış, her zaman yayınların sıradüzendeki tüm kuyruk yöneticilerine gönderilmesi, var olan aboneliklerden bağımsız olarak değiştirilebilir. Bu, abonelik bilgilerini sıradüzen boyunca yayma gerisini kaldırır, ancak yayın trafiğini artırabilir.

Bir küme yarattığınızda, iletilerin ağ içinde sonsuza kadar çevrilmesine neden olan bir döngü oluşturmamaya özen göstermeniz gerekir. Bir sıradüzende böyle bir döngü yaratılamaz.

Her kuyruk yöneticisinin benzersiz bir kuyruk yöneticisi adı olmalıdır.

Yayınlama/abone olma ileti akışları birden çok yayınlama/abone olma kümesine yayılabilir. Bunu yapmak için, bir yayınlama/abone olma sıradüzenine her bir kümeden bir kuyruk yöneticisi ekleyin.

Daha ayrıntılı bir keşif için bkz. Yayınlama/abone olma sıradüzenlerinde yöneltme.

Yayınlama/abone olma ağındaki yetkili sunucu abonelikleri

Yetkili sunucu aboneliği, başka bir kuyruk yöneticisininde yayınlanan konular için bir kuyruk yöneticisi tarafından yapılan bir aboneliktir. Yetkili abonelik, bir abonelik tarafından abone olunan her bir konu dizesine ilişkin kuyruk yöneticileri arasında akışları içerir. Yetkili sunucu aboneliklerini belirtik olarak yaratmadığınızda, kuyruk yöneticisi sizin adınıza bunu yapar.

Kuyruk yöneticilerini bir yayınlama/abone olma kümesine ya da bir yayınlama/abone olma sıradüzenine bağlayabilirsiniz. Yetkili sunucu abonelikleri, bağlı kuyruk yöneticileri arasında akar. Yetkili sunucu abonelikleri, abonelerin, diğer kuyruk yöneticilerine bağlı olan bu konuya abone olarak bir kuyruk yöneticisine bağlı bir yayıncı tarafından yaratılmış bir konuya neden olur. Bkz. “Dağıtılmış yayınlama/abone olma ağları” sayfa 86.

Bireysel konu dizgilerine binlerce aboneliğin bulunduğu ya da bu aboneliklerin varlığının hızla değişebileceği bir yayınlama/abone olma/abone olma topolojilerinde, yetkili abonelik yayılımın ek yükü göz önünde bulundurulmalıdır. Bu konunun geri kalanında açıklanan otomatik toplamanın yanı sıra, bağlı kuyruk yöneticileri arasında yetkili sunucu abonelikleri ve yayınlarının akışını daha da kısıtlayan ve yetkili sunucu aboneliğinin tüm bağlı kuyruk yöneticilerine yayılmasını engelleyen gecikme süresini azaltan el ile yapılandırma değişiklikleri yapabilirsiniz. Bkz. Yayınlama/abone olma ağlarındaki abonelik performansı.

Yetkili sunucu abonelikleri, yerel abonelikler tarafından kullanılan hiçbir seçiciyi ve genel arama karakterleri içeren abonelik konu dizgileri basitleştirilebilir. Bu, gerçek aboneliklerin olmadığı yetkili sunucu abonelikleriyle eşleşen yayınlara neden olabilir; bu, kuyruk yöneticileri arasında ek yayın akışı sağlar. Abonelikleri barındıran kuyruk yöneticisi, aboneliklere ek yayınların döndürülmemesi için bu tür tutarsızlıkları ortaya çıkarmaz.

Yetkili sunucu aboneliği toplaması

Yetkili sunucu abonelikleri, yinelenen bir eleme sistemi kullanılarak toplanır. Çözümlenen bir konu dizgisi için, ilk yerel abonelik ya da alınan yetkili abonelik için bir yetkili sunucu aboneliği gönderilir. Aynı konu dizgisine ilişkin sonraki abonelikler, bu var olan yetkili aboneliğin kullanımını sağlar.

Son yerel abonelik ya da alınan yetkili abonelik iptal edildikten sonra yetkili sunucu aboneliği iptal edilir.

Yayın toplaması

Bir kuyruk yöneticisinde aynı konu dizgisine birden fazla abonelik varsa, yayınlama/abone olma topolojisindeki diğer kuyruk yöneticilerinden bu konu dizgisiyle eşleşen her bir yayın ile eşleşen her bir yayının tek bir kopyası gönderilir. İletinin gelmesiyle, yerel kuyruk yöneticisi iletinin bir kopyasını her bir eşleşen aboneliğe aktarır.

Yetkili abonelikler joker karakterler içerdiğinde tek bir yayının konu dizesiyle eşleşmesi için birden çok yetkili sunucu aboneliği olabilir. Bir ileti, bağlı tek bir kuyruk yöneticisi tarafından yaratılan iki ya da daha fazla yetkili sunucu aboneliğine uyan bir kuyruk yöneticisininde yayınlandıysa, birden çok yetkili sunucu aboneliğini karşılamak için uzak kuyruk yöneticisine yayının yalnızca bir kopyası uzak kuyruk yöneticisine iletilir.

İlgili görevler

Dağıtılmış yayınlama/abone olma ağındaki döngü algılaması

Yetkili abonelikte genel arama karakterleri

Abonelikler, yayınlardaki birden çok konu dizgileriyle eşleşmek için konu dizgilerinde genel arama karakterleri kullanabilir.

Bir aboneliğin kullanabileceği iki genel arama karakteri şeması vardır: *konu tabanlı* ve *karakter tabanlı*. Bkz."Joker şemalar" sayfa 66.

IBM WebSphere MQ 7.0 ve sonraki sürümlerde, genel arama abonelikleri için tüm yetkili sunucu abonelikleri konuya dayalı genel arama karakterlerini kullanacak şekilde dönüştürülür. Karakter tabanlı bir genel arama karakteri bulunursa, bu karakter bir # karakteriyle değiştirilir, en yakın /karakterine geri döner. Örneğin, `/aaa/bbb/c*d` , `/aaa/bbb/#`olarak dönüştürülür. Uzak kuyruk yöneticilerindeki dönüştürme sonuçları, belirtik olarak abone olunandan biraz daha fazla yayın göndermektedir. Bu ek yayınlar, yayınları yerel abonelerine teslim ettiğinde, yerel kuyruk yöneticisi tarafından dışarı süzülmüş olur.

Controlling wildcard usage with the JOKER property

Yayınların genel arama karakteri konu dizgisi adlarını kullanan abone uygulamalarına teslim edilmesini denetlemek için MQSC **Topic** JOKER özelliğini ya da eşdeğer PCF Topic `WildcardOperation` özelliğini kullanın. JOKER özelliği, iki olası değerden birine sahip olabilir:

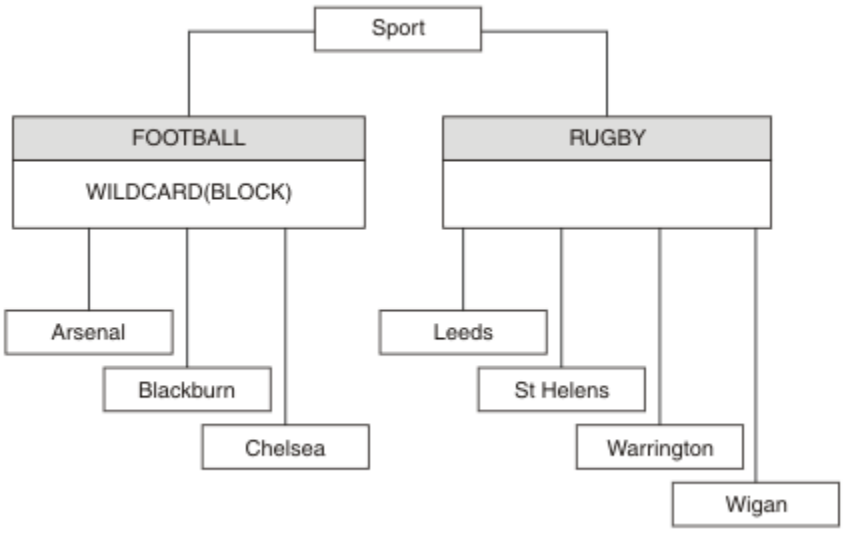
Genel arama karakteri
Bu konuya ilişkin genel arama karakteri aboneliklerinin davranışı.

Passthru
Bu konu nesnesindeki konu dizgisine göre daha az genel lengüzel bir konuya yapılan abonelikler, bu konu ve konu dizgilerine bu konudan daha özel yayınlar alır.

Öbek
Bu konu nesnesindeki konu dizgisinden daha az özel bir konuya yapılan abonelikler, bu konu ya da konu dizgilerine bu konudan daha özel yayınlar almaz.

Abonelikler tanımlandığında bu özniteliğin değeri kullanılır. Bu özniteliği değiştirirseniz, var olan aboneliklerin kapsadığı konular kümesi değişiklikten etkilenmez. Bu senaryo, konu nesneleri yaratıldığında ya da silindiğinde topoloji değiştirildiğinde de geçerlidir; WILDCARD özniteliğinin değiştirilmesinden sonra yaratılan aboneliklerle eşleşen konular kümesi, değiştirilen topoloji kullanılarak yaratılır. Eşleşen konu kümesini var olan abonelikler için yeniden değerlendirilmeye zorlamak istiyorsanız, kuyruk yöneticisini yeniden başlatmanız gerekir.

In the example, "Örnek: Sport yayınlama/abone olma kümesini yaratın" sayfa 79, you can follow the steps to create the topic tree structure shown in Şekil 23 sayfa 76.



Şekil 37. JOKER özelliğini kullanan bir konu ağacı, BLOCK

A subscriber using the wildcard topic string # receives all publications to the `Sport` topic and the `Sport/Rugby` subtree. The subscriber receives no publications to the `Sport/Football` subtree, because the JOKER property value of the `Sport/Football` topic is BLOCK.

Varsayılan değerPASSTHRU ' dir. `Sport` ağacındaki düğümlere JOKER özellik değerini PASSTHRU olarak ayarlayabilirsiniz. Düğümlerde BLOCK JOKER özellik değeri yoksa, PASSTHRU ayarı abonelerin `Sports` ağacındaki düğümlere göre gözlemlediği davranışı değiştirmez.

Örnekte, genel arama karakterinin, teslim edilen yayınları nasıl etkilediğini görmek için abonelikler oluşturun; bkz. Şekil 27 sayfa 81. Bazı yayınlar yaratmak için Şekil 30 sayfa 82 içinde yayınlama komutunu çalıştırın.

```
pub QMA
```

Şekil 38. QMA ' ya Yayınla

Sonuçlar Çizelge 3 sayfa 76içinde gösterilir. JOKER propertyözellik değerinin BLOCK' in genel arama karakteri kapsamında, genel arama karakterlerinin genel arama karakterlerine sahip aboneliklerini nasıl önlediğini fark edin.

Çizelge 6. QMAtarihinde alınan yayınlar

Abonelik	Konu dizesi	Alınan yayınlar	Notlar

SPORTS	`Sports/#`	`Sports`
		`Sports/Rugby`
		`Sports/Rugby/Leeds`
			All publications to Football subtree blocked by `WILDCARD(BLOCK)` on `Sports/Football`

SARSENAL	`Sports/#/Arsenal`	-	`Sports/Football` üzerinde`WILDCARD(BLOCK)` , `Arsenal`üzerinde genel arama karakteri aboneliğini önler
SLEEDS	`Sports/#/Leeds`	`Sports/Rugby/Leeds`	Sports/Rugby için varsayılan `WILDCARD` , `Leeds`üzerinde genel arama karakteri aboneliğini engellememektedir.

Not:

Suppose a subscription has a wildcard that matches a topic object with the JOKER property value BLOCK. Aboneliğin aynı zamanda eşleşen genel arama karakterinin sağında bir konu dizgisi varsa, abonelik hiçbir zaman bir yayını almaz. Engellenen yayınların kümesi, engellenen genel arama karakterinin ebeveynleri olan konular için yayınlardır. Publications to topics that are children of the topic with the BLOCK property value are blocked by the wildcard. Bu nedenle, genel arama karakterinin sağına bir konu içeren abonelik konu dizgileri, hiçbir yayın için hiçbir yayın almayacak.

WILDCARD özellik değerinin BLOCK olarak ayarlanması, genel arama karakterleri içeren bir konu dizgisini kullanarak abone olamayacağınız anlamına gelmez. Böyle bir abonelik normal. The subscription has an explicit topic that matches the topic with a topic object having a JOKER property value BLOCK. It uses wildcards for topics that are parents or children of the topic with the JOKER property value BLOCK. Şekil 23 sayfa 76örneğinde, Sports/Football/# gibi bir abonelik yayınları alabilir.

Genel arama karakterleri ve küme konuları

Küme konu tanımları, bir kümedeki her kuyruk yöneticisine dağıtılır. Kuyruk yöneticisinde bir küme yöneticisinde bulunan bir küme konusuna abonelik, yetkili abonelikleri oluşturan kuyruk yöneticisinde yer alan kuyruk yöneticisinde yer alan bir sonuçla sonuçlanır. Bir yetkili sunucu aboneliği, kümedeki diğer her kuyruk yöneticisinde yaratılır. Genel arama karakterleri içeren, küme konularıyla birleştirilen konuları kullanan abonelikler, davranışı tahmin etmek için zor bir yol gösterebilirler. Bu davranış, aşağıdaki örnekteki açıklamadır.

In the cluster set up for the example, “Örnek: Sport yayınlama/abone olma kümesini yaratın” sayfa 79, QMB has the same set of subscriptions as QMA, yet QMB received no publications after the publisher published to QMA, see Şekil 24 sayfa 76. Sports/Football ve Sports/Rugby konuları küme başlıklarına rağmen, fullsubs.tst içinde tanımlanan abonelikler bir küme konusuna gönderme yapmamış olur. Yetkili sunucu aboneliği, QMB 'tan QMA' a yayılmaz. Yetkili abonelikler olmadan, QMA hiçbir yayın QMB' a iletilmedi.

Some of the subscriptions, such as Sports/#/Leeds, might seem to reference a cluster topic, in this case Sports/Rugby. The Sports/#/Leeds subscription actually resolves to the topic object SYSTEM.BASE.TOPIC.

Sports/#/Leeds gibi bir aboneliğin gönderme yaptığı konu nesnesinin çözülmesine ilişkin kural şu şekildir. Konu dizesini ilk genel arama karakterine kes. İlişkili bir denetim konusu nesnesi olan ilk konuyu aramak için konu dizgisinden sola doğru tarayın. Konu nesnesi bir küme adı belirtebilir ya da bir yerel konu nesnesi tanımlayabilir. In the example, Sports/#/Leeds, the topic string after truncation is Sports, which has no topic object, and so Sports/#/Leeds inherits from SYSTEM.BASE.TOPIC, which is a local topic object.

Kümelenmiş konulara abone olunmanın, genel arama karakterinin yayılmasını nasıl değiştirebileceğini görmek için, upsubs.battoplu iş komut dosyasını çalıştırın. Komut dosyası abonelik kuyruklarını temizler ve fullsubs.tstiçinde küme başlığı aboneliklerini ekler. Bir dizi yayın yaratmak için puba.bat komutunu yeniden çalıştırın; bkz. Şekil 24 sayfa 76.

Çizelge 4 sayfa 78 , yayınların yayınlandığı kuyruk yöneticisine iki yeni abonelik eklenmesinin sonucunu gösterir. Sonuç beklendiği gibi, yeni abonelikler her biri bir yayın alır ve diğer abonelikler tarafından alınan yayınların sayıları değişmez. Diğer küme kuyruk yöneticisininde beklenmeyen sonuçlar ortaya çıkar; bkz. Çizelge 5 sayfa 78.

Çizelge 7. QMAtarihinde alınan yayınlar

Abonelik	Konu dizesi	Alınan yayınlar	Notlar
SPORTS	Sports/#	Sports Sports/Rugby Sports/Rugby/Leeds	All publications to Football subtree blocked by WILDCARD(BLOCK) on Sports/Football

Çizelge 7. QMAtarihinde alınan yayınlar (devamı var)			
Abonelik	**Konu dizesi**	**Alınan yayınlar**	**Notlar**
SARSENAL	Sports/#/Arsenal	-	Sports/Football üzerindeWILDCARD(BLOCK) , Arsenalüzerinde genel arama karakteri aboneliğini önler
SLEEDS	Sports/#/Leeds	Sports/Rugby/Leeds	Sports/Rugby için varsayılan WILDCARD , Leedsüzerinde genel arama karakteri aboneliğini engellememektedir.
FARSENAL	Sports/Football/ Arsenal	Sports/Football/ Arsenal	Arsenal , aboneliğin bir genel arama karakterinin olmadığı için bir yayın alır.
FLEEDS	Sports/Rugby/Leeds	Sports/Rugby/Leeds	Leeds , herhangi bir etkinlikte bir yayın alır.

Çizelge 5 sayfa 78 shows the results of adding the two new subscriptions on QMB and publishing on QMA. QMB ' in bu iki yeni aboneliği olmadan hiçbir yayın almadığını geri çağırın. As expected, the two new subscriptions receive publications, because Sports/FootBall and Sports/Rugby are both cluster topics. QMB forwarded proxy subscriptions for Sports/Football/Arsenal and Sports/Rugby/Leeds to QMA, which then sent the publications to QMB.

Beklenmeyen sonuç, daha önce yayın içermeyen Sports/# ve Sports/#/Leeds adlı aboneliklerin artık yayınları aldığından kaynaklanır. Bunun nedeni, diğer abonelikler için QMB 'e iletilen Sports/Football/Arsenal ve Sports/Rugby/Leeds yayınlarının artık QMB' e bağlı herhangi bir abone için kullanılabilmesinin nedeni. Consequently the subscriptions to the local topics Sports/# and Sports/#/Leeds receive the Sports/Rugby/Leeds publication. Spor/Futbol, JOKER özellik değeri BLOCKolarak ayarlanmış olduğundan,Sports/#/Arsenal bir yayın almamaya devam eder.

Çizelge 8. QMBtarihinde alınan yayınlar			
Abonelik	**Konu dizesi**	**Alınan yayınlar**	**Notlar**
SPORTS	*Sports/#*	*Sports/Rugby/ Leeds*	*All publications to Football subtree blocked by WILDCARD(BLOCK) on Sports/Football*
SARSENAL	Sports/#/Arsenal	-	Sports/Football üzerindeWILDCARD(BLOCK) , Arsenalüzerinde genel arama karakteri aboneliğini önler
SLEEDS	*Sports/#/Leeds*	*Sports/Rugby/ Leeds*	*Sports/Rugby için varsayılan WILDCARD , Leedsüzerinde genel arama karakteri aboneliğini engellememektedir.*
FARSENAL	Sports/Football/ Arsenal	Sports/Football/ Arsenal	Arsenal , aboneliğin bir genel arama karakterinin olmadığı için bir yayın alır.
FLEEDS	Sports/Rugby/Leeds	Sports/Rugby/Leeds	Leeds , herhangi bir etkinlikte bir yayın alır.

Çoğu uygulamada, bir aboneliğin başka bir aboneliğin davranışını etkilemesi istenmeyen bir davranışa neden olur. One important use of the JOKER property with the value BLOCK is to make the subscriptions

to the same topic string containing wildcards behave uniformly. Aboneliğin yayıncı ile aynı kuyruk yöneticisine mi, yoksa farklı bir aboneliğe mi ait olduğunu, aboneliğin sonuçlarının aynı olduğunu mu?

Joker karakterler ve akışlar

Yayınlama/abone olma API ' ya yazılan yeni bir uygulama için geçerli olan etki, * aboneliğinin yayınsız olarak alınmamasını sağlar. To receive all the Sports publications you must subscribe to Sports/*, or Sports/#, and similarly for Business publications.

Yayınlama/abone olma aracısı IBM WebSphere MQ 7 ' a ve sonraki sürümlere geçirildiğinde, kuyruğa alınan var olan bir yayınlama/abone olma uygulamasının davranışı değişmez. The **StreamName** property in the **Publish**, **Register Publisher**, or **Subscriber** commands is mapped to the name of the topic the stream has been migrated to.

Genel arama karakterleri ve abonelik noktaları

Yayınlama/abone olma API ' ya yazılan yeni bir uygulama için, geçişin etkisi, * aboneliğinin yayınsız olarak alınmamasını sağlar. To receive all the Sports publications you must subscribe to Sports/*, or Sports/#, and similarly for Business publications.

Yayınlama/abone olma aracısı IBM WebSphere MQ 7 ' a ve sonraki sürümlere geçirildiğinde, kuyruğa alınan var olan bir yayınlama/abone olma uygulamasının davranışı değişmez. **Publish**, **Register Publisher**ya da **Subscriber** komutlarındaki **SubPoint** özelliği, aboneliğin yeni düzeye geçirilmiş olduğu konunun adıyla eşlenir.

Örnek: Sport yayınlama/abone olma kümesini yaratın

Aşağıdaki adımlar, dört kuyruk yöneticisi içeren bir küme (CL1) oluşturur: iki tam havuz, CL1A ve CL1B, ve iki kısmi havuz, QMA ve QMB. Tüm havuzlar yalnızca küme tanımlamalarını tutmak için kullanılır. QMA , küme konusu anasistemine atanır. Sürekli abonelikler hem QMA hem de QMBüzerinde tanımlanır.

Not: Örnek, Windowsiçin kodlanmıştır. Örneğin, diğer platformlardaki örneği yapılandırmak ve sınamak için Create qmgrs.bat ve pub.batoluşturma kodunu yeniden kodlamanız gerekir.

1. Komut dosyalarını oluşturun.
   a. topics.tstyarat
   b. Create wildsubs.tst
   c. Create fullsubs.tst
   d. Create qmgrs.bat
   e. oluştur pub.bat
2. Yapılandırmayı oluşturmak için Create qmgrs.bat komutunu çalıştırın.

```
qmgrs
```

Create the topics in Şekil 23 sayfa 76. The script in figure 5 creates the cluster topics Sports/Football and Sports/Rugby.

Not: REPLACE seçeneği, bir konunun TOPICSTR özelliklerini değiştirmez. TOPICSTR , örnekte farklı konu ağaçlarını test etmek için kullanılan bir özeldir. Konuları değiştirmek için önce konuyu silin.

```
DELETE TOPIC ('Sports')
DELETE TOPIC ('Football')
DELETE TOPIC ('Arsenal')
DELETE TOPIC ('Blackburn')
DELETE TOPIC ('Chelsea')
DELETE TOPIC ('Rugby')
DELETE TOPIC ('Leeds')
DELETE TOPIC ('Wigan')
DELETE TOPIC ('Warrington')
DELETE TOPIC ('St. Helens')

DEFINE TOPIC ('Sports')   TOPICSTR('Sports')
DEFINE TOPIC ('Football')  TOPICSTR('Sports/Football') CLUSTER(CL1) WILDCARD(BLOCK)
DEFINE TOPIC ('Arsenal')  TOPICSTR('Sports/Football/Arsenal')
DEFINE TOPIC ('Blackburn') TOPICSTR('Sports/Football/Blackburn')
DEFINE TOPIC ('Chelsea')  TOPICSTR('Sports/Football/Chelsea')
DEFINE TOPIC ('Rugby')   TOPICSTR('Sports/Rugby')  CLUSTER(CL1)
DEFINE TOPIC ('Leeds')   TOPICSTR('Sports/Rugby/Leeds')
DEFINE TOPIC ('Wigan')   TOPICSTR('Sports/Rugby/Wigan')
DEFINE TOPIC ('Warrington') TOPICSTR('Sports/Rugby/Warrington')
DEFINE TOPIC ('St. Helens') TOPICSTR('Sports/Rugby/St. Helens')
```

Şekil 39. Konuları silin ve yaratın: topics.tst

Not: REPLACE konu dizgilerinin yerini değiştirmediği için konuları silin.

Joker karakterler içeren abonelikler oluşturun. Genel arama karakterleri, Şekil 23 sayfa 76içindeki konu nesneleriyle ilgili konulara karşılık gelir. Her abonelik için bir kuyruk yaratın. Kuyruklar temizlenir ve komut dosyası çalıştırıldığında ya da yeniden çalıştırıldığında abonelikler silinir.

Not: REPLACE seçeneği, bir aboneliğin TOPICOBJ ya da TOPICSTR özelliklerini değiştirmez. TOPICOBJ or TOPICSTR are the properties that are usefully varied in the example to test different subscriptions. Bunları değiştirmek için önce aboneliği silin.

```
DEFINE QLOCAL(QSPORTS)  REPLACE
DEFINE QLOCAL(QSARSENAL) REPLACE
DEFINE QLOCAL(QSLEEDS)  REPLACE
CLEAR QLOCAL(QSPORTS)
CLEAR QLOCAL(QSARSENAL)
CLEAR QLOCAL(QSLEEDS)

DELETE SUB  (SPORTS)
DELETE SUB  (SARSENAL)
DELETE SUB  (SLEEDS)
DEFINE SUB  (SPORTS)  TOPICSTR('Sports/#')     DEST(QSPORTS)
DEFINE SUB  (SARSENAL) TOPICSTR('Sports/+/Arsenal') DEST(QSARSENAL)
DEFINE SUB  (SLEEDS)  TOPICSTR('Sports/+/Leeds')  DEST(QSLEEDS)
```

Şekil 40. Genel arama karakteri abonelikleri yarat: wildsubs.tst

Küme konusu nesnelerine başvuruda bulunan abonelikler oluşturun.

Not:

The delimiter, /, is automatically inserted between the topic string referenced by TOPICOBJ, and the topic string defined by TOPICSTR.

DEFINE SUB(FARSENAL) TOPICSTR('Sports/Football/Arsenal') DEST(QFARSENAL) , aynı aboneliği oluşturur. TOPICOBJ , önceden tanımladığınız konu dizgisine hızlı bir şekilde gönderme yapmak için kullanılır. Abonelik, yaratıldığında, artık konu nesnesini ifade etmimez.

```
DEFINE QLOCAL(QFARSENAL) REPLACE
DEFINE QLOCAL(QRLEEDS)  REPLACE
CLEAR QLOCAL(QFARSENAL)
CLEAR QLOCAL(QRLEEDS)

DELETE SUB  (FARSENAL)
DELETE SUB  (RLEEDS)
DEFINE SUB  (FARSENAL) TOPICOBJ('Football') TOPICSTR('Arsenal') DEST(QFARSENAL)
DEFINE SUB  (RLEEDS)  TOPICOBJ('Rugby')  TOPICSTR('Leeds')  DEST(QRLEEDS)
```

Şekil 41. Abonelikleri sil ve yarat: fullsubs.tst

İki havuzu olan bir küme oluşturun. Yayınlama ve abone olma için iki kısmi havuz oluşturun. Her şeyi silmek ve yeniden başlamak için komut dosyasını yeniden çalıştırın. Komut dosyası aynı zamanda konu sıradüzenini ve ilk genel arama karakteri aboneliklerini de yaratır.

Not:

Diğer platformlarda, benzer bir komut dosyası yazın ya da tüm komutları yazın. Bir komut dosyasının kullanılması, her şeyi silmesini ve aynı bir yapılandırmayla yeniden başlatılmasını hızlı bir şekilde sağlar.

```
@echo off
set port.CL1B=1421
set port.CLIA=1420
for %%A in (CL1A CL1B QMA QMB) do call :createQM %%A
call :configureQM CL1A CL1B %port.CL1B% full
call :configureQM CL1B CL1A %port.CL1A% full
for %%A in (QMA QMB) do call :configureQM %%A CL1A %port.CL1A% partial
for %%A in (topics.tst wildsubs.tst) do runmqsc QMA < %%A
for %%A in (wildsubs.tst) do runmqsc QMB < %%A
goto:eof

:createQM
echo Configure Queue manager %1
endmqm -p %1
for %%B in (dlt crt str) do %%Bmqm %1
goto:eof

:configureQM
if %1==CL1A set p=1420
if %1==CL1B set p=1421
if %1==QMA set p=1422
if %1==QMB set p=1423
echo configure %1 on port %p% connected to repository %2 on port %3 as %4 repository
echo DEFINE LISTENER(LST%1) TRPTYPE(TCP) PORT(%p%) CONTROL(QMGR) REPLACE | runmqsc %1
echo START LISTENER(LST%1) | runmqsc %1
if full==%4 echo ALTER QMGR REPOS(CL1) DEADQ(SYSTEM.DEAD.LETTER.QUEUE) | runmqsc %1
echo DEFINE CHANNEL(TO.%2) CHLTYPE(CLUSSDR) TRPTYPE(TCP) CONNAME('LOCALHOST(%3)') CLUSTER(CL1)
REPLACE | runmqsc %1
echo DEFINE CHANNEL(TO.%1) CHLTYPE(CLUSRCVR) TRPTYPE(TCP) CONNAME('LOCALHOST(%p%)')
CLUSTER(CL1) REPLACE | runmqsc %1
goto:eof
```

Şekil 42. Kuyruk yöneticileri yarat: qmgrs.bat

Küme konularına abonelikleri ekleyerek yapılandırmayı güncelleyin.

```
@echo off
for %%A in (QMA QMB) do runmqsc %%A < wildsubs.tst
for %%A in (QMA QMB) do runmqsc %%A < upsubs.tst
```

Şekil 43. Abonelikleri güncelleştir: upsubs.bat

Yayın konusu dizgisini içeren iletileri yayınlamak için, parametre olarak kuyruk yöneticisiyle `pub.bat`komutunu çalıştırın. `Pub.bat` uses the sample program **amqspub**.

```
@echo off
@rem Provide queue manager name as a parameter
set S=Sports
set S=6 Sports/Football Sports/Football/Arsenal
set S=6 Sports/Rugby Sports/Rugby/Leeds
for %%B in (6) do echo %%B | amqspub %%B %1
```

Şekil 44. Yayınla: pub.bat

İlgili kavramlar

Genel arama abonelikleri ve alıkonan yayınlar

Yayın kapsamı

Bir yayınlama/abone olma kümesi ya da sıradüzeni yapılandırdığınızda, bir yayının kapsamı, kuyruk yöneticilerinin bir yayını uzak kuyruk yöneticilerine iletip iletmeyeceğini daha fazla denetler. Yayınların kapsamını yönetmek için **PUBSCOPE** konu özniteliğini kullanın.

Bir yayın uzak kuyruk yöneticilerine iletilmezse, yayını yalnızca yerel aboneler alır.

Bir yayınlama/abone olma kümesi kullandığınızda, yayınların kapsamı öncelikle konu ağacındaki belirli noktalardaki kümelenmiş konu nesnelerinin tanımlamasıyla denetlenir. Yayın kapsamı, yayınların kümedeki diğer kuyruk yöneticilerine akılmasına izin verecek şekilde ayarlanmalıdır. Belirli kuyruk yöneticilerindeki belirli konuların ince taneli denetimlerine gerek duyarken, kümelenmiş bir konu için yayın kapsamını kısıtlamalısınız.

Yayınlama/abone olma sıradüzeni kullandığınızda, yayınların kapsamı öncelikle abonelik kapsamı öznitelikle birlikte bu öznitelik tarafından denetlenir.

PUBSCOPE özniteliği, belirli bir konuda yapılan yayınların kapsamını belirlemek için kullanılır. Özniteliği aşağıdaki değerlerden birine ayarlayabilirsiniz:

MMGR
: Yayın yalnızca yerel abonelere teslim edilir. Bu yayınlara *yerel yayınlar*adı verilir. Yerel yayınlar, uzak kuyruk yöneticilerine iletilmez ve bu nedenle uzak kuyruk yöneticilerine bağlı olan aboneler tarafından alınmaz.

TÜMÜ
: Yayın, yerel abonelere ve bir yayınlama/abone olma kümesi ya da sıradüzenindeki uzak kuyruk yöneticilerine bağlı olan yerel abonelere ve abonelere teslim edilir. Bu yayınlara *genel yayınlar*adı verilir.

ASPARENT
: Konu ağacındaki üst konunun **PUBSCOPE** ayarını kullanın.

Yayıncılar, MQPMO_SCOPE_QMGR put iletisi seçeneğini kullanarak bir yayının yerel mi, yoksa genel mi olduğunu da belirleyebilir. Bu seçenek kullanılırsa, **PUBSCOPE** konu özniteliği kullanılarak ayarlanan herhangi bir davranışı geçersiz kılar.

İlgili kavramlar

"Yönetimle ilgili konu nesneleri" sayfa 73
Yönetimle ilgili bir konu nesnesini kullanarak, konulara varsayılan, varsayılan olmayan öznitelikler atayabilirsiniz.

İlgili görevler

Dağıtılmış yayınlama/abone olma ağlarının yapılandırılması

Abonelik kapsamı

Abonelik kapsamındaki bir aboneliğin kapsamı, bir kuyruk yöneticisinde aboneliğin olup olmadığını, yayınlama/abone olma kümesi ya da sıradüzeninde başka bir kuyruk yöneticisinde yayınlanan yayınların mı, yoksa yalnızca yerel yayıncılardaki yayınların yayınlanıp alınmayacağını denetler.

Abonelik kapsamını bir kuyruk yöneticisiyle sınırlandırma, yetkili sunucu aboneliklerini yayınlama/abone olma topolojisindeki diğer kuyruk yöneticilerine iletilmekten durdurur. Bu, kuyruklar arası yönetici yayınlama/abone olma ileti alışverişi trafiğini azaltır.

Bir yayınlama/abone olma kümesi kullandığınızda, aboneliklerin kapsamı öncelikle konu ağacındaki belirli noktalardaki kümelenmiş konu nesnelerinin tanımlamasıyla denetlenir. Abonelik kapsamı, yetkili sunucu aboneliklerinin kümedeki diğer kuyruk yöneticilerine akışının yapılmasına izin verecek şekilde ayarlanmalıdır. Belirli kuyruk yöneticilerindeki belirli konuların ince taneli denetime gereksinim duyarken, kümelenmiş bir konu için abonelik kapsamını kısıtlamalısınız.

Bir yayınlama/abone olma sıradüzeni kullandığınızda, aboneliklerin kapsamı öncelikle yayın kapsamı öznitelikle birlikte bu öznitelik tarafından denetlenir.

The **SUBSCOPE** topic attribute is used to determine the scope of subscriptions made to a specific topic. Özniteliği aşağıdaki değerlerden birine ayarlayabilirsiniz:

MMGR
Bir abonelik yalnızca yerel yayınları alır ve yetkili sunucu abonelikleri uzak kuyruk yöneticilerine yayılmaz.

TÜMÜ
Yetkili abonelik, yayınlama/abone olma kümesi ya da sıradüzeninde uzak kuyruk yöneticilerine yayılır ve abonenin yerel ve uzak yayınları alır.

ASPARENT
Konu ağacındaki üst konunun **SUBSCOPE** ayarını kullanın.

When subscription scope for a topic is set to TüM, either directly or resolved through ASPARENT, individual subscriptions to that topic can restrict their scope to MMGR by specifying MQSO_SCOPE_QMGR when creating the subscription. Kapsamı QMGR kapsamı olan bir konuya ilişkin abonelik, kapsamı ALLolarak genişletemez.

İlgili kavramlar

“Yönetimle ilgili konu nesneleri” sayfa 73
Yönetimle ilgili bir konu nesnesini kullanarak, konulara varsayılan, varsayılan olmayan öznitelikler atayabilirsiniz.

İlgili görevler

Dağıtılmış yayınlama/abone olma ağlarının yapılandırılması

Konu alanları

Konu alanı, abone olabileceğiniz ve yayınlayabileceğiniz başlıklara ilişkin kümedir. Dağıtılmış bir yayınlama/abone olma topolojisindeki bir kuyruk yöneticisinin, bu topolojideki bağlı kuyruk yöneticilerine abone olunan ve yayınlanmış olan konuları içeren bir konu alanı vardır.

Not: Bir kuyruk yöneticisinden, denetim konusu nesneleri, konu dizgileri ve konu ağaçları gibi konulara genel bakış için bkz.“Konular” sayfa 64. Aksi belirtilmediği sürece, yürürlükteki makaledeki *konulara* ilişkin ek başvurular *konu dizgileri* konusuna bakın.

Konular başlangıçta aşağıdaki yöntemlerden biriyle oluşturulur:

- Bir konu nesnesini ya da kalıcı aboneliği tanımladığınızda, yönetimsel olarak yönetimsel olarak.
- dinamik olarak, bir uygulama yeni bir konu için dinamik olarak bir yayın veya abonelik oluşturduğunda.

Konular diğer kuyruk yöneticilerine hem yetkili sunucu abonelikleri yoluyla, hem de denetim kümesi konu nesneleri yaratılarak yayılır. Yetkili sunucu abonelikleri, yayınlayıcının bağlı olduğu kuyruk yöneticisinden, abonelerin kuyruk yöneticilerine iletilmekte olan yayınlarla sonuçlanır.

Yetkili sunucu abonelikleri, bir kuyruk yöneticisi sıradüzenindeki üst-alt öğe ilişkileri tarafından birbirine bağlanan tüm kuyruk yöneticileri arasında geçirilir. Sonuç olarak, bir kuyruk yöneticisinde, sıradüzendeki başka bir kuyruk yöneticisinde tanımlanan bir konuya abone olabilirsiniz. Kuyruk yöneticileri arasında bağlı bir yol olduğu sürece, kuyruk yöneticilerinin nasıl bağlandığılarının bir önemi yoktur.

Yetkili sunucu abonelikleri, yayınlama/abone olma kümesindeki küme konularına abonelikler de yayılır. Küme konusu, **CLUSTER** özniteliğine sahip bir konu nesnesine eklenen ya da üst öğesinden özniteliği devralan bir konudur. Küme konuları olmayan konular yerel konular olarak bilinir ve kümede eşlenmez. Bir yetkili abonelik aboneliği, yerel konulara aboneliklerden kümeye geçirilir.

Özetlemek için, iki durumda aboneler için yetkili sunucu abonelikleri oluşturulur.

1. Kuyruk yöneticisi, bir sıradüzeninin üyesidir ve bir yetkili sunucu aboneliği kuyruk yöneticisinin üst ve alt öğelerinden iletilir.
2. Kuyruk yöneticisi bir kümenin üyesidir ve abonelik konu dizgisi, bir küme konusu nesnesiyle ilişkili bir konuya çözülüyor. Konu *doğrudan yönlendirilmiş* bir küme konusu olduğunda, yetkili abonelikler kümenin tüm üyelerine iletilir. Konu bir *konu anasistemi yönelttiği* küme konusu olduğunda, yetkili sunucu abonelikleri yalnızca kümelenmiş konu nesnesini tanımlamış olan kümedeki kuyruk yöneticilerine iletilir. Daha fazla bilgi için bkz.“Kümeleri yayınla/abone ol” sayfa 89.

Bir kuyruk yöneticisi, bir kümenin ve sıradüzeninin üyesiyse, yetkili abonelikler her iki mekanizma tarafından da aboneye yinelenen yayınlar sağlanmadan yayılır.

Üç yayınlama/abone olma topolojisine ilişkin konu alanları aşağıdaki listede açıklanmıştır:

- “Dava 1. Kümeleri yayınla/abone ol” sayfa 102.
- “Dava 2. Sürüm 7 ya da sonraki sürümlerde sıradüzenleri yayınla/abone ol” sayfa 103.
- “Dava 3. IBM WebSphere MQ 6' ta sıradüzenleri ve akışları yayınla/abone ol” sayfa 103.

Ayrı konularda, aşağıdaki yapılandırma görevleri konu alanlarının nasıl birleştirileceğini açıklar.

- Yayınlama/abone olma kümesinde tek bir konu alanı yaratılması.
- Var olan IBM WebSphere MQ 6 konu alanlarına IBM WebSphere MQ 7 ya da daha sonraki bir kuyruk yöneticisi eklenmesi.
- Birden çok küme konu alanlarının birleştirilmesi.
- Konu alanlarının birden çok kümede birleştirilmesi ve yalıtılması.
- Birden çok kümede konu boşluklara yayınlama ve abone olma.

Dava 1. Kümeleri yayınla/abone ol

Örnekte, kuyruk yöneticisinin bir yayınlama/abone olma sıradüzenine bağlı olmadığını varsayın.

Bir kuyruk yöneticisi yayınlama/abone olma kümesinin üyesiyse, konu alanı yerel konulardan ve küme konularından oluşur. Yerel konular, **CLUSTER** özniteliği olmayan konu nesneleriyle ilişkilendirilir. Bir kuyruk yöneticisinin yerel konu nesnesi tanımlamaları varsa, bu alanın konu alanı, yerel olarak tanımlanmış bir konu nesnelerine de sahip olan kümedeki başka bir kuyruk yöneticisinden farklıdır.

Bir yayınlama/abone olma kümesinde, abone olduğunuz konu bir küme konusu nesnesine çözümlemiyorsa, başka bir kuyruk yöneticisinde tanımlanan bir konuya abone olamazsınız.

Birden çok kuyruk yöneticisine bir küme konu nesnesinin aynı adı (örneğin, *konu anasistem yöneltmesi*kullanılırken) gerekli olduğunda, tüm tanımlamaların gerekli olduğu yerlerde eşleşmesi önemlidir. Daha fazla bilgi için bakınız: Creating a single topic space in a publish/subscreen cluster.

Tanımın bir küme konusu ya da yerel konu için olup olmadığı, bir konu nesnesinin yerel tanımlaması, kümenin başka bir yerinde tanımlanan aynı konu nesnesinden önceliklidir. Yerel olarak tanımlanmış konu, başka bir yerde tanımlanan nesne daha yeni olsa bile kullanılır.

Küme konusu nesnesinin, kümedeki her yerde aynı konu dizgisiyle ilişkilendirilmesi önemlidir. Bir konu nesnesinin ilişkilendirildiği konu dizesini değiştiremezsiniz. Aynı konu nesnesini farklı bir konu dizgisiyle ilişkilendirmek için, konu nesnesini silmeniz ve yeni konu dizesiyle yeniden yaratmanız gerekir. Konu kümelenmişse, etki, kümenin diğer üyelerinde saklanan konu nesnesinin kopyalarını silmek ve daha sonra, yeni konu nesnesinin kopyalarının küme içinde her yerde yaratılabilmesinden etkilenir. Konu nesnesinin kopyalarının tümü aynı konu dizgisine gönderme yapıyor.

Yanlışlıkla farklı konu dizgileriyle kümedeki farklı kuyruk yöneticilerinde aynı adı taşıyan konu nesnesinin iki tanımını yanlışlıkla yaratmanız mümkündür. Bu, farklı konu dizgilerine sahip aynı konu nesnesinin

birden çok tanımı, konuya nasıl ve konuya gönderme yapıldığından bağlı olarak farklı sonuçlar üretebileceğinden, kafa karıştırıcı davranışlara neden olabilir. Bu önemli noktaya ilişkin daha fazla bilgi için Aynı ada sahip birden çok küme konusu tanımlaması başlıklı konuya bakın.

Dava 2. Sürüm 7 ya da sonraki sürümlerde sıradüzenleri yayınla/abone ol

Örnekte, kuyruk yöneticisinin bir yayınlama/abone olma kümesinin üyesi olmadığını varsayın.

IBM WebSphere MQ 7 ya da sonraki bir yayın düzeyiyle, bir kuyruk yöneticisi yayınlama/abone olma sıradüzeninin bir üyesiyse, konu alanı yerel olarak ve bağlı kuyruk yöneticilerindeki tüm konulardan oluşur. Bir sıradüzendeki tüm kuyruk yöneticilerinin konu alanı aynı. Yerel konular ve genel konular arasında bir bölüm bölünmesi yoktur.

Bir yayıncıdan, sıradüzendeki farklı kuyruk yöneticilerine bağlı bir aboneye akan bir konuda bir yayının yayınlanmasını önlemek için, **PUBSCOPE** ve **SUBSCOPE** seçeneklerinden birini QMGRolarak ayarlayın.

Kuyruk yöneticisi QMA' daki USA/Alabama konu dizgisiyle Alabama bir konu nesnesi tanımladığınızı varsayalım. Sonuç aşağıdaki gibidir:

1. Artık QMA konumundaki konu alanı Alabama konu nesnesini ve USA/Alabamakonu dizgisini içerir.
2. An application or administrator can create a subscription at QMA using the topic object name Alabama.
3. Bir uygulama, sıradüzendeki herhangi bir kuyruk yöneticisinde USA/Alabamadahil olmak üzere herhangi bir konuya abonelik oluşturabilir. If QMA has not been defined locally, the topic USA/Alabama resolves to the topic object SYSTEM.BASE.TOPIC.

Dava 3. IBM WebSphere MQ 6' ta sıradüzenleri ve akışları yayınla/abone ol

IBM WebSphere MQ 7öncesinde, konu alanı, tüm kuyruk yöneticilerindeki varsayılan akışın yer aldığı ayrı akışlar olarak bölündü. Yayınlar, farklı akışlar arasında akışa geçemez. Adlandırılmış akışlar kullanılırsa, farklı kuyruk yöneticilerindeki konu alanları farklı olabilir. Konular, varsayılan akıştaki konulara ve farklı adlandırılmış akışlardaki konulara bölünmektedir.

Not: Her adlandırılmış akış ayrı bir konu alanı oluşturur. Bağlı bir topoloji oluşturmak için bağlı kuyruk yöneticilerindeki her bir adlandırılan akış var olmalıdır. X akımı X, QMA ve QMC' de tanımlıdır, ancak QMBüzerinde tanımlanmaz. QMA , QMB'nin üst öğesiyse ve QMB , QMC' nin üst öğesiyse, XX akışındaki hiçbir konu QMA ile QMCarasında akamaz.

Hem **PUBSCOPE** hem de **SUBSCOPE** seçeneklerinin QMGR ya da ALL olarak ayarlanması, yalnızca yerel tüketime ilişkin yayınların değiş tokuş edilmesi ya da yalnızca genel kullanım için yayınların değiş tokuş edilmesi için bir yayıncı ve abone olmasını gerektirir.

IBM WebSphere MQ 7' tan, yayınlama/abone olma API 'si kullanılarak akışlar kullanılamaz. Bir IBM WebSphere MQ 7 kuyruk yöneticisine kuyruklanmış yayınlama/abone olma olanağını kullanırsanız, akışlar akışların etkisinin benzetimini yapan farklı konu nesnelerine eşlenir. Akıştaki tüm konular için kök konu olan bir konu nesnesi yaratılarak bir akış benzetimi oluşturur. Kuyruk yöneticisi, akışın ve her bir ağacın ilgili kök konusu arasındaki yayınları ve abonelikleri eşliyor.

İlgili kavramlar

“Yayın kapsamı” sayfa 100
Bir yayınlama/abone olma kümesi ya da sıradüzeni yapılandırdığınızda, bir yayının kapsamı, kuyruk yöneticilerinin bir yayını uzak kuyruk yöneticilerine iletip iletmeyeceğini daha fazla denetler. Yayınların kapsamını yönetmek için **PUBSCOPE** konu özniteliğini kullanın.

“Abonelik kapsamı” sayfa 100
Abonelik kapsamındaki bir aboneliğin kapsamı, bir kuyruk yöneticisinde aboneliğin olup olmadığını, yayınlama/abone olma kümesi ya da sıradüzeninde başka bir kuyruk yöneticisinde yayınlanan yayınların mı, yoksa yalnızca yerel yayıncılardaki yayınların yayınlanıp alınmayacağını denetler.

İlgili görevler

Dağıtılmış yayınlama/abone olma ağlarının yapılandırılması

IBM MQ Çok Yaylı

IBM MQ Multicast, düşük gecikme süresi, yüksek fan dışında, güvenilir çoklu yayın ileti sistemi sunar.

Çoklu yayın, yüksek sayıda aboneye ölçeklendirilebileceği gibi, performansa zarar vermeksizin, yayınlama/abone olma mesajlarının verimli bir biçimidir. IBM MQ , yüksek fan çıkışıyla düşük gecikme süreli ileti sistemi sağlamak için onayları, negatif onayları ve sıra numaralarını kullanarak güvenilir Multicast ileti alışverişlerinden olanak sağlar.

IBM MQ Multicast 'ın adil teslimatı, hiçbir alıcının bir avantaj elde edilmemesinin sağlanması için eşzamanlı teslimat olanaklarına olanak sağlar. IBM MQ Multicast, iletileri teslim etmek için ağı kullandıkça, çıkış verilerine bir yayınlama/abone olma altyapısı gerekmez. Bir konu bir grup adresiyle eşlendikten sonra, bir kuyruk yöneticisine gerek yoktur; yayıncılar ve aboneler eşdüzeyler arası kipte çalışabilirler. Bu, kuyruk yöneticisi sunucularında yükleme azaltılmasına izin verir ve kuyruk yöneticisi sunucusu artık olası bir hata noktası değildir.

İlk çok noktaya yayın kavramları

IBM MQ Multicast, İletişim Bilgileri (COMMINFO) nesnesi kullanılarak, var olan sistemlere ve uygulamalara kolayca tümleştirilebilir. İki KONU nesnesi alanı, çoklu yayın trafiğini desteklemek ya da yoksaymak için var olan KONU nesnelerinin hızlı bir şekilde yapılandırılabilmesini sağlar.

Çok hedefli nesneler için gereken nesneler

Aşağıdaki bilgiler, IBM MQ Multicast için gerekli olan iki nesneye ilişkin kısa bir genel bakış sağlar:

COMMINFO nesnesi
COMMINFO nesnesi, çok hedefli iletimle ilişkili öznitelikleri içerir. COMMINFO nesne değiştirgeleriyle ilgili ek bilgi için DEFINE COMMINFObaşlıklı konuya bakın.

Ayarlanması gereken tek COMMINFO alanı, COMMINFO nesnesinin adıdır. Daha sonra bu ad, COMMINFO nesnesini bir konuya tanıtmak için kullanılır. Değerin geçerli bir çoklu yayın grubu adresi olduğundan emin olmak için COMMINFO nesnesinin **GRPADDR** alanı denetlenmelidir.

KONU NESNESI
Konu, bir yayınlama/abone olma iletisinde yayınlanan bilgilerin konusu ve bir konu TOPIC nesnesi yaratılarak tanımlanmıştır. KONU nesnesi parametreleri hakkında daha fazla bilgi için bkz. KONUYU TANıMLAMA.

Var olan konular, şu KONU nesnesi parametrelerinin değerlerini değiştirerek çok hedefli olarak kullanılabilir: **COMMINFO** ve **MCAST**.

- **COMMINFO** Bu parametre, çok hedefli iletişim bilgileri nesnesinin adını belirtir.
- **MCAST** Bu parametre, konu ağacındaki bu konumda çoklu yayının izin verilip verilmeyeceğini belirtir. Varsayılan olarak **MCAST** , ASPARENT değerine ayarlıdır; bu, konunun çoklu yayın özniteliğinin üst öğeden devralınacağı anlamına gelir. **MCAST** ayarı ENABLE olarak ayarlanıyor, bu düğümde çok noktaya yayın trafiği var.

Çok noktaya yayın ağları ve konular

Aşağıdaki bilgiler, farklı abonelik tiplerine ve konu tanımlamasına sahip aboneliklere ilişkin bir genel bakış sunar. Bu örnekler, KONU nesnesi **COMMINFO** parametresinin geçerli bir COMMINFO nesnesinin adına ayarlandığını varsayar:

Konu çok noktaya yayın etkinleştirilmiş olarak ayarlandı
Konu dizgisi **MCAST** parametresi ENABETLEolarak ayarlandıysa, çok hedefli yetenekli istemcilere abonelikler izin verilir ve aşağıdaki durumlar dışında birden çok noktaya gönderim aboneliği yapılır:

- Bu, çok hedefli yetenekli bir istemciden gelen dayanıklı bir aboneliktir.
- Bu, çok hedefli yetenekli bir istemciden yönetilmeyen bir aboneliktir.
- Bu, çok hedefli olmayan bir istemciden gelen bir aboneliktir.

Bu durumda, çok noktaya gönderim aboneliği olmayan bir abonelik yapılır ve abonelikler normal yayınlama/abone olma düzeyine indirgenir.

Konu çok noktaya gönderim geçersiz kılındı olarak ayarlandı
Konu dizgisi **MCAST** parametresi DISABLEolarak ayarlandıysa, çok hedefli olmayan bir abonelik her zaman yapılır ve abonelikler normal yayınlama/abone olma düzeyine indirgenir.

Konu yalnızca çok hedefli olarak ayarlandı
Konu dizgisi **MCAST** parametresi YALNIZCAolarak ayarlandıysa, çok hedefli olan istemcilerden abonelikler izin verilir ve aşağıdaki durumlar dışında birden çok noktaya gönderim aboneliği yapılır:

- Sürekli abonelik: Durable abonelikleri neden kodu 2436 (0984) (RC2436) ile reddedilir: MQRC_DURABILITY_NOT_ALLOWALIZE
- Bu, yönetilmeyen bir aboneliktir: Yönetilmeyen abonelikler, 2046 (07FE) (RC2046): MQRC_OPTIONS_ERRORneden koduyla reddedilir.
- Çok hedefli olmayan bir istemciden abonelik: Bu abonelikler neden kodu 2560 (0A00) (RC2560): MQRC_MULTICAST_ONLYile reddedilir.
- Yerel bir uygulamanın aboneliğidir: Bu abonelikler neden kodu 2560 (0A00) (RC2560): MQRC_MULTICAST_ONLYile reddedilir.

Linux Windows AIX MQ Telemetry'e genel bakış

MQ Telemetry , bir kuyruk yöneticisinin parçası olan bir telemetri (MQXR) hizmetidir. Kendinizi yazabileceğiniz ya da ücretsiz olarak yükleyebileceğiniz telemetrik istemcilerden ve komut satırı ve gezgin yönetici arabirimleri içerir. Telemetri, geniş bir uzak aygıt yelpazesinden veri toplamak ve bu çeşitli uzak aygıtların denetlenmesi anlamına gelir. MQ Telemetry ile, web uygulamaları ile aygıtların veri toplamasını ve denetimini bütünleştirebilirsiniz.

IBM WebSphere MQ 7.1' dan MQ Telemetry , IBM MQbileşeninin bir bileşenidir. Upgrading for these versions is essentially installing a later version of IBM MQ.

Ancak, IBM MQ 8.0 ' den başlayarak, Client Software Development Kit artık ürünün bir parçası olarak sağlanmamaktadır. Benzer örnek uygulamalar, Eclipse Paho ve MQTT.org' da serbestçe kullanılabilir olmaya devam eder. Bkz. IBM MQ Telemetry Transport örnek programları.

MQ Telemetry , IBM WebSphere MQ 7.1 ve daha sonraki bir bileşendir; MQ Telemetry , ana ürünle birlikte kurulabilir ya da ana ürün kurulduktan sonra kurulabilir. Geçiş bilgileri için bkz. Migrating MQ Telemetry on Windows ve Migrating MQ Telemetry on Linux.

Included in MQ Telemetry are the following components:

Telemetri kanalları
MQTT istemcilerinin IBM MQ ' e bağlanmasını yönetmek için telemetri kanallarını kullanın. Telemetri kanalları, IBM MQgibi yeni IBM MQ nesnelerini (SYSTEM.MQTT.TRANSMIT.QUEUEile etkileşim kurmak için) kullanır.

Telemetri (MQXR) hizmeti
MQTT istemcileri, telemetri kanallarına bağlanmak için SYSTEM.MQXR.SERVICE telemetri hizmetini kullanır.

MQ TelemetryiçinIBM MQ Explorer desteği
MQ Telemetry , IBM MQ Explorerkullanılarak yönetilebilir.

Documentation
MQ Telemetry belgeleri, IBM WebSphere MQ 7.1 standardındaki standart IBM MQ ürün belgelerine eklenmiştir. Java ve C istemcilerine ilişkin SDK belgeleri, ürün belgelerinde ve Javadoc ve HTML biçiminde sağlanır.

Telemetri kavramları

Ne yapacağınıza karar vermek için etraftaki ortamlardan bilgi topluyorsunuz. Bir tüketici olarak, hangi yiyeceklerin satın alacağına karar vermeden önce, mağazada ne olduğunu kontrol edin. Bir bağlantı ayarlamadan önce, şimdi ayrılırsanız, yolculuğun ne kadar süreceğini bilmek istiyorsunuz. Doktorla

görüşmeden önce belirtilerini kontrol et. Bir otobüs ne zaman varacağınızı kontrol edin, bekleyip beklememeye karar vermeden önce. Bu kararların bilgileri doğrudan sayaçlardan ve cihazlardan, yazılı kelimeden kağıda ya da bir ekrandan geliyor, ve senden. Nerede olduğunuz ve ne zaman ihtiyaç duyarsanız, bilgi toplayabilir, bir araya getirmeniz, analiz etmek ve üzerinde işlem yapmak gerekir.

Bilgi kaynakları yaygın olarak dağılırsa veya ulaşılamazsa, en doğru bilgiyi toplamak zor ve maliyetli hale gelir. Yapmak istediğiniz çok sayıda değişiklik varsa ya da değişiklikleri yapmak zorlaşmışsa, değişiklikler yapılmaz ya da daha az etkililince yapılır.

peki ya bilgi toplama maliyetleri ve kontrol eden, yaygın olarak dağılan cihazlar dijital teknolojiyle cihazları internete bağlayarak büyük ölçüde azalsa ne olur? Bu bilgiler, İnternet ve kurumsal kaynaklar kullanılarak analiz edilebilir. Size bilinçli kararlar vermek ve onları harekete geçirmeniz için daha fazla fırsatınız var.

Teknolojik eğilimler, çevresel ve ekonomik baskılar, bu değişikliklerin gerçekleşmesini sağlar:

1. Sensörlerin ve çalıştırıcıların bağlanma ve kontrol etme maliyeti, standardizasyon ve düşük maliyetli dijital işlemcilere bağlantı nedeniyle azaltılır.
2. internet ve internet teknolojileri, cihazları birbirine bağlamak için giderek daha fazla kullanılıyor. bazı ülkelerde cep telefonları, internet uygulamalarına yapılan bağlantı sayısındaki kişisel bilgisayarları aşıyor. Diğer aygıtlar mutlaka takip edilir.
3. Internet ve internet teknolojileri, bir uygulamanın veri alabilmek için çok daha kolay hale getirmektedir. Verilere kolay erişim, verileri algılayıcılardan elde edilen verileri daha çok çözümde yararlı olan bilgilere dönüştürmek için veri analitiğinin kullanımını yönlendiriyor.
4. Kaynakların akıllı kullanımı genellikle karbon emisyonlarını ve maliyetlerini azaltmanın daha hızlı ve daha ucuz bir yoludur. Alternatifler: yeni kaynaklar bulmak veya mevcut kaynakları kullanmak için yeni teknolojiler geliştirmek, uzun vadeli çözüm olabilir. Yeni teknolojilerin geliştirilmesi ya da yeni kaynakların bulunması, var olan çözümlerin iyileştirilmesinden daha yavaş, daha yavaş ve daha yüksek maliyetli olmak üzere, genellikle daha yavaş ve daha yüksek maliyetli bir biçimde ortaya çıkmaktadır.

Örnek

Bir örnek, bu eğilimlerin ortamla etkileşim kurmak için yeni fırsatları nasıl yaratacağını akıllı bir şekilde gösterir.

Uluslararası Yaşam Güvenliği Konvansiyonu (SOLAS), bir çok gemilerde Otomatik Tanımlama Sistemi 'nin (AIS) görevlendirilmesini gerektirir. 300 ton ve yolcu gemisinde ticaret gemilerine ihtiyaç var. AIS, kıyı taşımacılığı için öncelikle bir çarpışma önleme sistemidir. Kıyı sularını izlemek ve kontrol etmek için deniz yetkilileri tarafından kullanılır.

Dünyanın dört bir yanındaki meraklıları düşük maliyetli AIS izleme istasyonlarını devreye sokuyor ve kıyı sevkiyat bilgilerini internete yerleştiriyor. Diğer meraklılar, AIS ' in bilgilerini internetteki diğer bilgilerle birleştiren uygulamalar yazıyorlar. Sonuçlar Web sitelerine bağlı ve Twitter ve SMS kullanılarak yayınlanmıştır.

Bir uygulamada, Southampton yakınlarındaki AIS istasyonlarından gelen bilgiler, gemi sahipliği ve coğrafi bilgi ile birleştirilir. Uygulama, feribot gelenlemeleri ve gidenleri hakkında canlı bilgileri Twitter' a besler. Southampton ve Wight Adası arasındaki feriboları kullanan düzenli yolcuların Twitter ya da SMS kullanarak haber kaynağına abone olması. Eğer beste, feribotlarını geç saatte çalıştırıyorsa, yolcuların kalkış ve varış zamanından sonra limanda feribota binmelerini geciktirebilir.

Daha fazla örnek için bkz. “Telemetri kullanım senaryoları” sayfa 108.

İlgili görevler

kurmaMQ Telemetry
YönetmeMQ Telemetry
Windowsüzerinde MQ Telemetry geçiriliyor
Linuxüzerinde MQ Telemetry geçiriliyor
MQ Telemetryiçin uygulama geliştirilmesi
MQ Telemetry sorun giderme

İlgili başvurular
MQ Telemetry Başvuru

Linux Windows AIX MQ Telemetry' a giriş

İnsanlar, işletmeler ve hükümetler giderek daha akıllı bir şekilde yaşayıp içinde çalışacağımız çevreyle daha akıllı bir şekilde etkileşim kurmak için MQ Telemetry ' i kullanmayı istiyorlar. MQ Telemetry , her türlü aygıtı İnternet 'e ve işletmeye bağlar ve akıllı aygıtlar için uygulama oluşturma maliyetlerini azaltır.

MQ Telemetry nedir?

- IBM MQ tarafından sağlanan evrensel ileti sistemi omurgasını, çok çeşitli uzak algılayıcılar, çalıştırıcılar ve telemetri aygıtlarına genişleten IBM MQ özelliğidir. MQ Telemetry , akıllı kurumsal uygulamaları, hizmetleri ve karar vericileri donanımlı aygıt ağlarıyla birbirine bağlayan akıllı kurumsal uygulamaları, hizmetleri ve karar alıcıları birbirine bağlayabilmesi için IBM MQ ' ı genişletir.
- MQ Telemetry ' in temel parçaları şunlardır:

  MQ Telemetry (MQXR) hizmeti.
  Bu hizmet IBM MQ Server içinde çalışır ve telemetri aygıtlarıyla iletişim kurmak için IBM MQ Telemetry Transport (MQTT) iletişim kuralını kullanır.

  YazdığınızMQTT uygulamaları.
  Bu uygulamalar, telemetri aygıtları ile IBM MQ kuyruk yöneticisi arasında taşınan bilgileri ve bu bilgilere yanıt olarak alınan tüm işlemleri denetler. Bu uygulamaların yaratılmasına yardımcı olmak için MQTT istemci kitaplıklarını kullanıyorsunuz.

Benim için ne yapabilir?

- MQTT is an open messaging transport that allows MQTT implementations to be created for a wide variety of devices.
- MQTT istemcileri, sınırlı kaynaklara sahip küçük ayak izi aygıtlarında çalıştırılabilir.
- MQTT , manyetik bant genişliğinin düşük olduğu, verilerin gönderilmesinin maliyetinin pahalı olduğu ya da kırılgan olabilecek ağlarda verimli bir şekilde çalışır.
- İleti teslimi, uygulamadan güvenli bir şekilde sağlanır ve bu uygulamadan ayrılır.
- Uygulama programcılarının iletişim programlama bilgisine sahip olması gerekmez.
- İletiler diğer ileti sistemi uygulamalarıyla değiş tokuş edilebilir. Bunlar başka bir telemetri uygulaması ya da bir MQI, JMS ya da kurumsal ileti sistemi uygulaması olabilir.

Nasıl kullanırım?

- Örnek komut dosyaları, örnek bir IBM MQ Telemetry Transport v3 istemci uygulaması (`mqttv3app.jar`) ile birlikte çalışılır. Bkz. IBM MQ Telemetry Transport örnek programları.
- Use the IBM MQ Explorer and its associated tools to administer the telemetry feature of IBM MQ.
- Bir kuyruk yöneticisine bağlanan ve yayınlama/abone olma ileti alışverişi kullanan MQTT uygulamaları yaratmanıza yardımcı olması için istemci kitaplıklarını kullanın.
- Uygulamanızı ve istemci kitaplığınızı uygulamanızın çalışacağı aygıta dağıtın.

Nasıl çalışır?

- MQTT , bir yayınlama aboneliği iletişim kuralıdır. An MQTT client application can publish messages to an MQTT server, or subscribe for messages that are sent by applications that connect to an MQTT server.
- MQTT istemci uygulamaları, MQTT ileti aktarımsını gerçekleştiren istemci kitaplıklarını kullanır.
- Temel bir MQTT istemci uygulaması, standart bir MQ istemcisi gibi çalışır, ancak çok daha geniş bir platform ve ağ üzerinde çalışabilir.
- MQ Telemetry (MQXR) hizmeti, bir IBM MQ kuyruk yöneticisini MQTT sunucusuna dönüştürür.

- When an IBM MQ queue manager acts as the MQTT server, other applications that connect to the queue manager can subscribe for and receive the messages from the MQTT client.
- Kuyruk yöneticisi, uygulamaları yayınlamak için yayınlama uygulamalarından iletileri dağıtma yönelticisi olarak işlev görür.
- İletiler, farklı istemci uygulamaları tipleri arasında dağıtılabilir. Örneğin, Telemetry istemcileri ile JMS istemcileri arasında.

Not: MQ Telemetry replaces the SCADA nodes that were withdrawn in version 7 of WebSphere Message Broker (now known as IBM Integration Bus) and runs on Windows, Linux, and AIX.

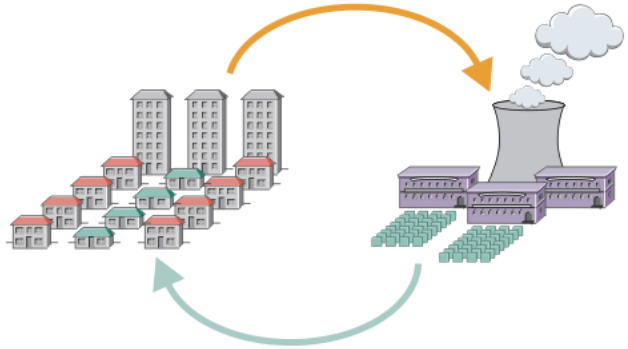
Linux Windows AIX

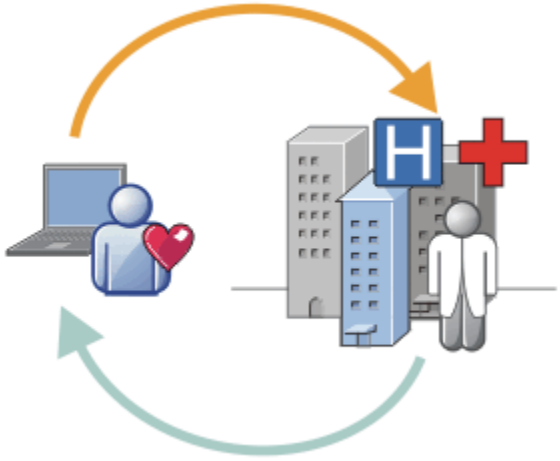
Telemetri kullanım senaryoları

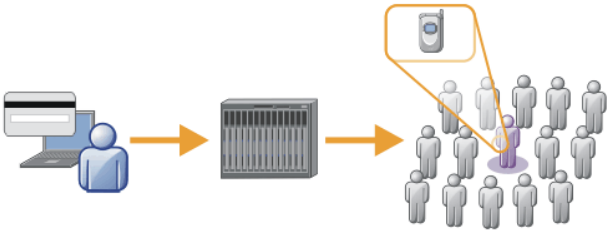
Telemetry, verilerin otomatik algılanması, verilerin ölçülmesi ve uzaktan aygıtların denetlenmesini sağlar. Bu vurgu, verilerin aygıtlardan merkezi bir denetim noktasına iletilmesinin üzerinde yer alıyor. Telemetri ayrıca, aygıtlara yapılandırma ve denetim bilgilerinin gönderilmesine de dahildir.

MQ Telemetry connects small devices by using the MQTT protocol, and connects the devices to other applications by using IBM MQ. MQ Telemetry , aygıtlar ve İnternet arasında bir boşluk oluşturmanın "akıllı çözümler" oluşturulmasına daha kolay yol açılmasını sağlar. Akıllı çözümler, aygıtları izleyen ve denetleyen uygulamalar için İnternet 'te ve kurumsal uygulamalarda kullanılabilir olan bilgilerin zenginliğini açığa çıkarlar.

Aşağıdaki şemalarda MQ Telemetry' un bazı tipik kullanımları gösterilmektedir:

Teletext: Smart Electricity	
	• Hizmet sağlayıcıya gönderilen enerji kullanımı verilerini içerenMQTT iletisi. • MQ Telemetry , enerji kullanımı verilerinin analizine dayalı olarak CONTROL COMMANS komutunu gönderir. • Daha fazla bilgi için aşağıdaki kullanım senaryolarına bakın: “Telemetri kullanım senaryosu: Home enerji izleme ve denetimi” sayfa 110

Teletext: Akıllı Sağlık Hizmetleri	
• MQ Telemetry , Health Data 'yı Hastanız ve Doctor 'a gönderir. • MQTT message alerts or feedback can be sent based on analysis of Health Data. • Daha fazla bilgi için aşağıdaki kullanım senaryolarına bakın: “Telemetri kullanım senaryosu: Ev hasta izleme” sayfa 109	

Telemetri: Bir Crowd 'da bir	
	• Banka sunucusuna basit bir kart işlemi gönderilir. • MQ Telemetry , kartvizitinin kullanıldığı müşteriyi uyarlayan, binlerciden bir kişiyi tanımlar. • MQ Telemetry , en basit bilgi girişini kullanabilir ve bu kişinin yerini bulabilir.

Alt konularda açıklanan kullanım senaryoları gerçek örneklerden çizilir. Telemetreyi kullanmanın yollarını ve telemetri teknolojisinin çözmesi gereken bazı ortak sorunları gösteriyorlar.

Linux Windows AIX Telemetri kullanım senaryosu: Ev hasta izleme

Bir kardiyak hasta bakım sisteminde IBM ile bir sağlık hizmeti sağlayıcısı arasındaki işbirliği içinde, implant edilen bir kardiyoter defibrilatör bir hastaneyle iletişim kurar. Hasta ve implant cihazına ilişkin veriler, bir hastanın evindeki MQTT cihazına RF telemetrisi kullanılarak aktarılır.

Genellikle aktarım gece boyunca yatak başucunda bulunan bir vericiye alınır. Verici, verileri telefon sistemi üzerinden güvenli bir şekilde hastaneye aktarır ve veriler analiz edilir.

Sistem, bir hastanın hekime yapması gereken ziyaretlerin sayısını azaltır. Hasta ya da cihazın ne zaman dikkat edilmesi gerektiğini algılıyor ve acil bir durum halinde, nöbetçi hekime uyarıyor.

IBM ile healthcare sağlayıcısı arasındaki işbirliği, telemetri kullanım senaryolarında ortak olan özelliklere sahiptir:

Görünmezlik

Aygıt, güç sağlamak, telefon hattı ve günün bir parçası için aygıta yakın olmak dışında herhangi bir kullanıcı müdahalesini gerektirmez. Bunun kullanımı güvenilir ve kullanımı basittir.

Aygıtı aygıtı ayarlamaya ilişkin gereksinimi kaldırmak için aygıt tedarikçisi aygıtı önceden yapılandırır. Hasta sadece onu bağlamalıdır. Yapılandırmanın hasta tarafından ortadan kaldırılması aygıtın çalışmasını basitleştirir ve aygıtın yanlış bir şekilde yapılandırılmasını azaltır.

MQTT istemcisi, aygıtın bir parçası olarak gömüledir. Aygıt geliştiricisi, aygıt ve geliştirici ya da sağlayıcındaki MQTT istemcisi uygulamasını yerleştirir, ön yapılandırmanın bir parçası olarak MQTT istemcisini yapılandırır.

MQTT istemcisi, geliştiricinin Java uygulaması içinde bulunduğu bir Java SE JAR dosyası olarak verilir. Bu gibiJava dışı ortamlar için, aygıt geliştirici, yayınlanan MQTT biçimlerini ve iletişim kuralını kullanarak farklı bir dilde bir istemci uygulayabilir. Diğer bir seçenek olarak, geliştirici, Windows, Linux ve ARM platformları için paylaşılan kitaplıklar olarak teslim edilen C istemcilerinden birini kullanabilir.

Çift olmayan bağlantı

Defibrilatör ile hastane arasındaki iletişim ağ karakteristiğinden bile uzak. İki farklı ağ, hastadan veri toplamanın ve verileri hastaneye göndermekte kullanılan farklı problemleri çözmek için kullanılır. Patent ve MQTT aygıtı arasında, kısa menzilli bir düşük güç RF ağı kullanılır. Verici, düşük bant genişliğine sahip telefon hattı üzerinden bir VPN TCP/IP bağlantısı kullanarak hastaneye bağlanır.

Her aygıtı doğrudan bir Internet Protocol ağına bağlamanın bir yolunu bulmak genellikle kullanışsız olur. Bir göbeğe bağlı iki ağın kullanılması ortak bir çözümdür. MQTT aygıtı basit bir merkezdir, bilgileri hastadan depolar ve hastaneye iletir.

Güvenlik

hekim, hasta verilerinin özgünlüğünü güvenebilmelidir. hasta, verilerinin gizliliğini ise, saygı duyulmasını ister.

Bazı durumlarda, VPN ya da TLS ' yi kullanarak bağlantıyı şifrelemek için yeterlidir. Diğer durumlarda, verilerin depolandıktan sonra bile saklanmasının arzu edilir.

Bazen telemetri cihazı güvenli değildir. Örneğin, paylaşılan bir konuta olabilir. Verilerin doğru hastadan olduğundan emin olmak için aygıtın kullanıcısının kimliği doğrulanmalıdır. Aygıtın kendisi, TLS ' yi kullanan sunucuya doğrulanabilir ve sunucu, aygıta doğrulanır.

The telemetry channel between the device and the queue manager supports JAAS for user authentication and TLS for communication encryption, and device authentication. Bir yayına erişim, IBM MQiçindeki nesne yetkili yöneticisi tarafından denetlenir.

Kullanıcının kimliğini doğrulamak için kullanılan tanıtıcı, ortak bir hasta kimliği gibi farklı bir tanıtıcıyla (örneğin, farklı bir tanıtıcı) eşlenebilir. Ortak bir tanıtıcı, IBM MQiçindeki konuları yayımlamak için gereken yetkiyi yapılandırmayı basitleştirir.

Bağlanırlık

MQTT aygıtı ile hastane arasındaki bağlantı, çevirmeli arama kullanır ve 300 baud kadar düşük bir bant genişliğine sahip çalışır.

300 baud 'da etkili bir şekilde çalışmak için, MQTT protocol , TCP/IP üstbilgilerine ek olarak bir iletiye yalnızca birkaç fazla bayt ekler.

MQTT protocol , gecikmeleri düşük tutan tek iletim *fire and forget* ileti sistemini sağlar. Ayrıca, garantili teslimat yanıt süresinden daha önemli olduğunda *en az bir kez* ve *tam olarak bir kez* teslimat için birden çok iletim olanağı da kullanılabilir. Teslimatı garanti altına almak için iletiler, başarıyla teslim edilinceye kadar aygıtta depolanır. Bir aygıt kablosuz olarak bağlıysa, garantili teslimat özellikle yararlı olur.

Ölçeklenebilirlik

Telemetri cihazları genellikle on binlerce ile milyonlara kadar büyük sayılarda konuşlandırılır.

Birçok aygıtın bir sisteme bağlanması, bir çözüm üzerinde büyük taleplerin yer almasını sağlar. Aygıtların ve bunların yazılımlarının maliyeti, lisansları, aygıtları ve kullanıcıları yönetmeyi gerektiren iş talepleri vardır. Teknik talepler, ağ üzerindeki yükü ve sunuculara ilişkin bilgileri içerir.

Bağlantıların açılması, açık bağlantıları korumaktan daha fazla sunucu kaynağı kullanır. Ancak, telefon hatlarını kullanan bu tür bir kullanım durumunda, bağlantıların giderilmesi, bağlantıların gerekli olmadığı kadar açık bırakıldığı anlamına gelir. Veri aktarımları büyük ölçüde batmış bir tabiat. Gece boyunca bağlantıların ani bir şekilde doruk noktasına çıkmasını önlemek için bağlantılar gece boyunca zamanlanabilir.

İstemcide, istemci yapılanışlarının ölçeklenebilirliğinin en alt düzeyde istemci konfigürasyonuna yardımcı olması gerekir. The MQTT client is embedded in the device. Aygıtların hastalara yerleştirilmesi için bir yapılandırma ya da MQTT istemci lisansı kabul adımının oluşturulmasına ilişkin bir gereksinim yoktur.

Sunucuda, MQ Telemetry kuyruk yöneticisi başına ilk 50.000 açık bağlantı hedefine sahiptir.

Bağlantılar IBM MQ Explorerkullanılarak yönetilir. IBM MQ Explorer , bağlantıları yönetilebilir bir sayıya göre süzer. Uygun bir şekilde seçilen bir şema ile istemcilere, coğrafi bölgelere göre veya hasta adına göre alfabetik olarak süzgeç uygulayabilir.

Linux Windows AIX Telemetri kullanım senaryosu: Home enerji izleme ve denetimi

Akıllı sayaçlar, enerji tüketimi hakkında geleneksel sayaçlardan daha fazla detay topluyor.

Akıllı sayaçlar genellikle bir evde tek tek aletleri izlemek ve kontrol etmek için yerel telemetri ağı ile birleşmektedir. Bazıları uzaktan izleme ve denetim için uzaktan da bağlanır.

Uzak bağlantı bir kişi tarafından, bir güç yardımcı programı tarafından ya da merkezi bir denetim noktası tarafından ayarlanabilir. Uzaktan denetim noktası, güç kullanımını okuyabilir ve kullanım verilerini

sağlayabilir. Sürekli fiyatlandırma ve hava durumu bilgileri gibi kullanımı etkilemek için veri sağlayabilir. Bu, genel güç oluşturma verimliliğini artırmak için yükü sınırlayabilir.

Akıllı sayaçlar yaygın olarak devreye alınmaya başlıyor. Örneğin Birleşik Krallık hükümeti, 2020 yılına kadar her Birleşik Krallık 'a akıllı sayaçların devreye alınması konusunda istişare halinde.

Ev ölçümü kullanım durumlarının bir dizi ortak özellikleri vardır:

Görünmezlik

Kullanıcı ölçümü kullanarak enerji tasarrufu için yer almak istemezse, ölçüm kullanıcı müdahalesi gerektirmemelidir. Bu, tek tek aygıtlara enerji arzının güvenilirliğini azaltmamalıdır.

Bir MQTT istemcisi, ölçümle birlikte konuşlandırılan yazılıma yerleştirilebilir ve ayrı bir kuruluş ya da yapılandırma gerektirmez.

Çift olmayan bağlantı

Aygıtlar ve akıllı ölçüm arasındaki iletişim, ölçüm ve uzak bağlantı noktası arasında olduğundan farklı bağlanırlık standartları gerektirir.

Akıllı ölçüm aygıtından aygıtlara yönelik bağlantı yüksek düzeyde kullanılabilir olmalı ve bir ev alanı ağı için ağ standartlarına uygun olmalıdır.

Uzak ağın çeşitli fiziksel bağlantıları kullanması olasılığı vardır. Bunlardan bazıları, cep telefonu gibi yüksek bir iletim maliyetine sahip ve aralıklı olarak da olabilir. MQTT v3 belirtimi, uzak bağlantıları ve yerel bağdaştırıcılar ile akıllı ölçüm bağlantıları arasındaki bağlantıları temel almaktadır.

Güç prizleri ile uygulanabilirliği arasındaki bağlantı ve ölçüm, bir ev alanı ağı (Zigbee gibi) kullanın. Algılayıcı ağları (MQTT-S) içinMQTT , Zigbee ve diğer düşük bant genişliği ağ protokolleriyle çalışmak üzere tasarlanmıştır. MQ Telemetry doğrudan MQTT-S ' yi desteklemez. MQTT-S ile MQTT v3arasında bağlantı kurmak için bir ağ geçidi gerektirir.

ev hasta izleme gibi, ev enerji izleme ve kontrol için çözümler, akıllı sayacı bir göbek olarak kullanarak birbirine bağlı birden fazla ağı gerektirir.

Güvenlik

akıllı sayaçlarla ilişkili bir takım güvenlik sorunları var. Bu sorunlar, işlemlerin saygınlığı, başlatılan herhangi bir denetim eyleminin yetkilendirilmesi ve güç tüketimi verilerinin gizliliğidir.

Gizliliği sağlamak için, ölçüm ve uzak denetim noktası arasında MQTT tarafından aktarılan verilerin TLS kullanılarak şifrelenmesi gerekir. Denetim eylemlerinin yetkilendirilmesini sağlamak için, ölçüm ile uzaktan denetim noktası arasındaki MQTT bağlantısı, TLS kullanılarak karşılıklı olarak doğrulanabilir.

Bağlanırlık

Uzak ağın fiziksel yapısı oldukça farklı olabilir. Bu, var olan bir geniş bant bağlantısını kullanabilir ya da yüksek çağrı maliyetleri ve aralıklı kullanılabilirlik ile bir mobil ağ kullanır. Yüksek maliyetli, kesintili bağlantılar için MQTT , verimli ve güvenilir bir protokoldür; bkz. “Telemetri kullanım senaryosu: Ev hasta izleme” sayfa 109.

Ölçeklenebilirlik

Sonunda güç şirketleri ya da merkezi kontrol noktaları, on milyonlarca akıllı sayacı devreye almayı planlıyor. Başlangıçta, her bir devreye alma başına sayaç sayısı on binlere kadar binlerdir. Bu sayı, kuyruk yöneticisi başına 50.000 açık istemci bağlantısının ilk MQTT hedefi ile karşılaştırılabilir.

ev enerji izleme ve kontrol için mimarinin kritik bir yönü, akıllı sayacı bir ağ yoğunlaştırıcı olarak kullanmaktadır. Her bir aygıt bağdaştırıcısı ayrı bir algılayıcıdır. Merkezi MQTTkullanarak bir yerel göbeğe bağlayarak, göbek, veri akışlarını merkezi denetim noktasıyla tek bir TCP/IP oturumuna yoğunlaştırabilir ve aynı zamanda oturum kesintilerini aşmak için kısa bir süre için iletileri saklayabilir.

Uzak bağlantılar, iki nedenden dolayı ev enerjisi kullanım senaryolarında açık bırakılmalıdır. İlk olarak, bağlantıların açılması, isteklerin gönderilmesine göre uzun bir göreli yol alır. Kısa bir aralıktaki "yük sınırlaması" isteklerini göndermek için birçok bağlantının açılacağı süre çok uzun. İkinci olarak, güç şirketinden yükleme sınırlaması istekleri almak için, önce bağlantı istemci tarafından açılmalıdır.

MQTTile bağlantılar her zaman istemci tarafından başlatılır ve güç şirketinden yükleme sınırlaması istekleri almak için bağlantının açık bırakılması gerekir.

Bağlantı açma hızı kritik önem gösteriyorsa ya da sunucu, zaman açısından kritik öneme sahip istekleri başlattıysa, çözüm genellikle birçok açık bağlantının sürdürülmesi için kullanılır.

Linux Windows AIX Telemetri kullanım durumları: Radyo Frekansı Tanımlama (RFID)

RFID, bir nesneyi kablosuz olarak tanımlamak ve izlemek için katıştırılmış RFID etiketi kullanımlıdır. RFID etiketleri, birkaç metre aralığa kadar okunabilir ve RFID okuyucunun görüş satırından dışarı çıkabilirler. Edilgen etiketler, bir RFID okuyucusu tarafından etkinleştirilir. Etkin etiketler, dış etkinleştirme olmadan iletir. Etkin etiketler bir güç kaynağına sahip olmalıdır. Pasif etiketler, aralıklarını artırmak için bir güç kaynağı içerebilir.

RFID birçok uygulamada kullanılır ve kullanım senaryosu tipleri büyük ölçüde farklılık gösterir. RFID kullanım senaryoları, evde hasta izleme ve ev enerji izleme ve kontrol kullanım senaryoları, bazı benzerlikler ve farklılıklar vardır.

Görünmezlik

Birçok kullanım senaryolarında, RFID okuyucu büyük sayılarda devreye alınır ve kullanıcı müdahalesi olmadan çalışmalıdır. Okuyucu, merkezi bir denetim noktasıyla iletişim kurmak için yerleşik bir MQTT istemcisi içerir.

Örneğin, bir dağıtım deposunda, okuyucu bir paleti algılamak için bir hareket algılayıcısı kullanır. Bu, paletteki öğelerin RFID etiketlerini etkinleştirir ve merkezi uygulamalara veri ve istekler gönderir. Veriler, stokun konumunu güncellemek için kullanılır. İstekler, belirli bir bölmeye taşınması gibi daha sonra palete ne olacağını denetler. Havayolları, ve havaalanı bagaj sistemleri, bu şekilde RFID kullanıyor.

Bazı RFID kullanım senaryolarında, okuyucunun Java Platform, Micro Edition (Java ME) gibi standart bir bilgi işlem ortamına sahip olması gerekir. Bu durumlarda MQTT istemcisi, üretim işleminden sonra ayrı bir yapılandırma adımında konuşlandırılabilir.

Çift olmayan bağlantı

RFID okuyucuları, bir MQTT istemcisi içeren yerel denetim aygıtından ayrılabilir ya da her bir okuyucu bir MQTT istemcisi yerleştirebilir. Genellikle coğrafi ya da iletişim katsayıları topoloji seçimini belirtir.

Güvenlik

Gizlilik ve özgünlük, RFID etiketlerinin ekinde güvenlikle ilgili sorunlardır. RFID etiketleri dikkat çekemez ve gizli olarak izlenebilir, aldatılabilir ya da kurcalanabilir.

RFID güvenlik sorunlarının çözümü, yeni RFID çözümlerinin devreye alınması için fırsat artırır. Güvenlik açığı RFID etiketinde yer alsa da, merkezi bilgi işleme kullanılarak yerel okuyucu, farklı tehditlere karşı yaklaşıma ilişkin yaklaşımlar önerir. Örneğin, etiket kurcalama, teslimatlara ve dağıtıma karşı stok düzeylerinin dinamik olarak ilişkilendirilerek saptanması olabilir.

Bağlanırlık

RFID uygulamaları genellikle, RFID okuyucularından ve anında sorgulardan toplanan bilgilerin hem toplu depolarına hem de iletisine dahil olur. Dağıtım ambarı kullanım senaryosında, RFID okuyucusu her zaman bağlanır. Bir etiket okunduğunda, okuyucuya ilişkin bilgiler de yayınlanır. Depolama uygulaması, yanıtı okuyucuya geri yayınlar.

Veri ambarı uygulamasında, ağ genellikle güvenilir olur ve anlık istekler düşük gecikme süresi performansı için *yangın ve unut* iletilerini kullanabilir. Toplu depo ve iletme verileri, veri denetleme ile ilişkili yönetim maliyetlerini en aza indirmek için *aynen bir kez* ileti sistemini kullanabilir.

Ölçeklenebilirlik

RFID uygulaması hemen yanıt gerektiriyorsa, ikinci ya da iki saniye sırasıyla, RFID okuyucuları bağlı kalmalıdır.

Linux Windows AIX Telemetri kullanım senaryoları: Ortam algılama

Çevre algılama, nehir suyu düzeyleri ve kalitesi, atmosferik kirleticiler ve diğer çevresel veriler hakkında bilgi toplamak için telemetri kullanır.

Algılayıcılar, kablolu iletişime erişmeden uzak yerlerde sık sık yer alır. Kablosuz bant genişliği pahalı ve güvenilirliği düşük olabilir. Genellikle, küçük bir coğrafi bölgede yer alan bir dizi ortam algılayıcısı, güvenli bir yerde yerel izleme aygıtına bağlanır. Yerel bağlantılar kablolu ya da kablosuz olarak kullanılabilir.

Görünmezlik

Algılayıcı aygıtları, merkezi izleme aygıtından daha az erişilebilir, daha düşük güçlü ve daha fazla sayıda devreye alınmış olabilir. Algılayıcılar bazen "aptal", ve yerel izleme aygıtı, algılayıcı verilerini dönüştürmek ve depolamak için bağdaştırıcıları içerir. İzleme aygıtı büyük olasılıkla, Java Platform, Standard Edition (Java SE) ya da Java Platform, Micro Edition (Java ME) özelliğini destekleyen bir genel amaçlı bilgisayar birleştirecektir. Görünmezlik, MQTT istemcisi yapılandırılırken önemli bir gereksinim olma olasılığının düşük olduğunu ifade eder.

Çift olmayan bağlantı

Algılayıcıların ve uzak bağlantının maliyeti ve bant genişliğinin yetenekleri, genellikle merkezi bir sunucuya bağlı bir yerel izleme göbeğiyle sonuçlanır.

Güvenlik

Çözüm, bir askeri veya savunma kullanım durumunda kullanılmadığı sürece, güvenlik önemli bir gereksinim değildir.

Bağlanırlık

Pek çok kullanım, sürekli izleme ya da anında veri kullanılabilirliği gerektirmiyor. Sel düzeyi uyarısı gibi özel durum verileri, hemen iletilmeli. Algılayıcı verileri, bağlantı ve iletişim maliyetlerini azaltmak için yerel izleme programında toplanır ve zamanlanan bağlantılar kullanılarak aktarılır. Kural dışı durum verileri, monitörde algılanır algılanmaz iletilir.

Ölçeklenebilirlik

Algılayıcılar, yerel göbeklerin etrafında yoğunlaştırılır ve algılayıcı verileri, bir zamanlamaya göre iletilen paketlere toplanır. Her iki faktör de, doğrudan bağlı algılayıcılar kullanılarak empoze edilecek olan merkezi sunucudaki yükü azaltır.

Linux Windows AIX Telemetri kullanım senaryoları: Mobil uygulamalar

Mobil uygulamalar, kablosuz aygıtlar üzerinde çalışan uygulamalardır. Aygıtlar soysal uygulama altyapıları ya da özel aygıtlar.

Genel platformlarda telefon ve kişisel veri asistanları gibi taşınabilir aygıtlar ve dizüstü bilgisayarlar gibi taşınabilir aygıtlar yer alır. Özel aygıtlar, belirli uygulamalara göre uyarlanan özel amaçlı donanımı kullanır. "Signed-for" parcel teslimi kaydetmek için kullanılan bir aygıt, özel bir mobil aygıt örneğidir. Özel mobil aygıtlardaki uygulamalar genellikle bir genel yazılım platformu üzerinde oluşturulabiliyor.

Görünmezlik

Özel mobil uygulamaların devreye alınması yönetiliyor ve MQTT istemci uygulamasının yapılandırılmasını içerebilir. Görünmezlik, MQTT istemcisi yapılandırılırken önemli bir gereksinim olma olasılığının düşük olduğunu ifade eder.

Çift olmayan bağlantı

Önceki kullanım senaryolarının yerel göbek topolojisinden farklı olarak, mobil istemciler uzaktan bağlanır. İstemci uygulama katmanı doğrudan merkezi göbekteki bir uygulamaya bağlanır.

Güvenlik

Küçük fiziksel güvenlik, mobil aygıt ve mobil kullanıcının kimliği doğrulanmalıdır. TLS, aygıtın kimliğini doğrulamak için kullanılır ve kullanıcının kimliğini doğrulamak için JAAS kullanılır.

Bağlanırlık

Mobil uygulama kablosuz kapsama bağlıysa, çevrimdışı çalışabilmeli ve kesintiye uğramış bir bağlantıyla verimli bir şekilde başa çıkabilmelidir. Bu ortamda amaç bağlı kalmak, ancak uygulamanın iletileri depolayabilmesi ve iletebilmesi gerekir. Genellikle iletiler, siparişler ya da teslim onaylarıdır ve önemli iş değerine sahiptir. Depolanmış ve güvenilir bir şekilde iletilmeleri gerekiyor.

Ölçeklenebilirlik

Ölçeklenebilirlik önemli bir sorun değildir. Uygulama istemcilerinin sayıları, özel mobil uygulama kullanım senaryolarında, binlik ya da onbinleri aşmayacak şekilde bulunur.

Linux Windows AIX Telemetri aygıtları kuyruk yöneticisine bağlama

Telemetri aygıtları, bir MQTT v3 istemcisi kullanarak bir kuyruk yöneticisine bağlanır. MQTT v3 istemcisi, telemetri (MQXR) hizmeti adı verilen bir TCP/IP dinleyicisine bağlanmak için TCP/IP ' yi kullanır.

Bir telemetri aygıtını kuyruk yöneticisine bağladığınızda, MQTT istemcisi `MqttClient.connect` yöntemini kullanarak bir TCP/IP bağlantısı başlatır. Like IBM MQ clients, an MQTT client must be connected to the queue manager to send and receive messages. The connection is made at the server using a TCP/IP listener, installed with MQ Telemetry, called the telemetry (MQXR) service. Her kuyruk yöneticisi, en çok bir telemetri (MQXR) hizmeti çalıştırır.

Telemetri (MQXR) hizmeti, telemetri kanalına bağlantı ayırmak için `MqttClient.connect` yöntemindeki her istemci tarafından ayarlanan uzak yuva adresini kullanır. Yuva adresi, TCP/IP anasistem adı ve kapı numarasının birleşimidir. Aynı uzaktan yuva adresini kullanan birden çok istemci, telemetri (MQXR) hizmetiyla aynı telemetri kanalına bağlıdır.

Bir sunucuda birden çok kuyruk yöneticisi varsa, telemetri kanallarını kuyruk yöneticileri arasında bölün. Kuyruk yöneticileri arasında uzak yuva adreslerini ayırın. Her telemetri kanalını benzersiz bir uzak yuva adresi ile tanımlayın. İki telemetri kanalı aynı yuva adresini kullanmamalıdır.

Aynı uzak yuva adresi, birden çok kuyruk yöneticisindeki telemetri kanalları için yapılandırıldıysa, bağlantı kurmak için ilk telemetri kanalı kazanır. Aynı adresle bağlantı kuran sonraki kanallar başarısız olur.

Sunucuda birden çok ağ bağdaştırıcısı varsa, uzaktan yuva adreslerini telemetri kanalları arasında bölün. Yalnızca tek bir telemetri kanalında herhangi bir özel yuva adresi yapılandırıldığı sürece, yuva adreslerinin ayrılması tamamen gelişigüzel bir şekilde ortaya konmaktadır.

Configure IBM MQ to connect MQTT clients using the wizards provided in the MQ Telemetry supplement for IBM MQ Explorer. Diğer bir seçenek olarak, telemetreyi el ile yapılandırmak için Linux ve AIXüzerinde telemetri için kuyruk yöneticisi yapılandırılması ve Windowsüzerinde telemetri kuyruk yöneticisi yapılandırılıyor içindeki yönergeleri izleyin.

İlgili başvurular

MQXR özellikleri

Linux Windows AIX Telemetri bağlantısı iletişim kuralları

MQ Telemetry , TCP/IP IPv4 ve IPv6ve TLS ' yi destekler.

Linux Windows AIX Telemetri (MQXR) hizmeti

Telemetri (MQXR) hizmeti bir TCP/IP dinleyicidir ve IBM MQ hizmeti olarak yönetilir. Create the service using an IBM MQ Explorer wizard, or with a **runmqsc** command.

MQ Telemetry (MQXR) hizmeti SYSTEM.MQXR.SERVICE olarak adlandırılır.

IBM MQ Explorericin MQ Telemetry işlevinde sağlanan Telemetry örnek yapılandırma sihirbazı, telemetri hizmetini ve bir örnek telemetri kanalı oluşturur; bkz. Verifying the installation of MQ Telemetry by using IBM MQ Explorer .

Komut satırından örnek yapılanışı yaratın; bkz. Komut satırını kullanarak MQ Telemetry kuruluşunun doğrulanması.

Telemetri (MQXR) hizmeti kuyruk yöneticisiyle otomatik olarak başlatılır ve durdurulur. IBM MQ Explorericindeki hizmetler klasörünü kullanarak hizmeti denetleyin. To see the service, you must click the icon to stop IBM MQ Explorer filtering out SYSTEM objects from the display.

Hizmetin el ile nasıl yaratılabilmesinin bir örneği için bkz.

- Linux AIX Linuxüzerinde SYSTEM.MQXR.SERVICE oluşturma.
- Windows Windowsüzerinde SYSTEM.MQXR.SERVICE oluşturma.

V 9.2.4 IBM MQ 9.2.4 , Linuxüzerinde SYSTEM.MQXR.SERVICE oluşturmave Windowsüzerinde SYSTEM.MQXR.SERVICE oluşturma tarihleri arasında, MQTT TLS kanallarının şifrelenmesi için geçiş tümcelerinden gerekli olan varsayılan anahtarı belirtebilmek için güncellenir. Daha fazla bilgi için bakınız: Encryption of passprsases for MQTT TLS kanals.

Linux Windows AIX Telemetri kanalları

Java Authentication and Authorization Service (JAAS) ya da TLS kimlik doğrulaması gibi farklı özelliklerle bağlantı oluşturmak ya da istemci gruplarını yönetmek için telemetri kanalları yaratın.

Create Telemetry channels using the **New Telemetry Channel** wizard, supplied in the MQ Telemetry function for IBM MQ Explorer. Belirli bir TCP/IP kapısındaki MQTT istemcilerinden gelen bağlantıları kabul etmek için sihirbazı kullanarak bir kanal yapılandırın. Since IBM WebSphere MQ 7.1, you can configure MQ Telemetry using the command line program, **runmqsc**.

İstemcileri gruplara ayırarak, çok sayıda istemci bağlantısının daha kolay yönetmesini sağlamak için farklı kapılarda birden çok telemetri kanalı oluşturun. Her telemetri kanalının farklı bir adı vardır.

Farklı bağlantı tipleri yaratmak için farklı güvenlik öznitelikleri içeren telemetri kanallarını yapılandırabilirsiniz. Farklı TCP/IP adreslerindeki istemci bağlantılarını kabul etmek için birden çok kanal yaratın. İletileri şifrelemek ve telemetri kanalını ve istemciyi doğrulamak için TLS ' yi kullanın; bkz. MQTT istemcilerinin ve telemetri kanallarının TLS yapılandırması. IBM MQ nesnelerine erişim yetkilendirmesini basitleştirmek için kullanıcı kimliğini belirtin. MQTT kullanıcısının kimliğini JAASile doğrulamak için bir JAAS yapılandırması belirtin; bkz. MQTT istemci tanıtıcısı, yetkilendirme ve kimlik doğrulaması.

Linux Windows AIX IBM MQ Telemetry Transport iletişim kuralı

IBM MQ Telemetry Transport (MQTT) v3 iletişim kuralı, küçük aygıtlar arasında düşük bant genişliğine ya da pahalı bağlantılara ilişkin iletilerin değiş tokuş edilmesine ve iletilerin güvenilir bir şekilde gönderilmesine yönelik olarak tasarlanmıştır. TCP/IP kullanır.

MQTT protocol yayınlanmıştır; bkz. IBM MQ Telemetry Transport biçimi ve iletişim kuralı. Protokolün 3. sürümü yayınlama/abone olma özelliğini kullanır ve şu üç hizmet niteliklerini destekler: *fire and forget*, *en az bir kez*ve *tam olarak bir kez*.

İletişim kuralı üstbilgilerinin küçük boyutu ve byte dizisi iletisi bilgi yükü, iletileri küçük tutar. Üstbilgiler, 2 baytlık sabit bir üstbilgiyi oluşturur ve en çok 12 bayt ek değişken üstbilgisi içerir. Bu protokol, abone olmak ve bağlanmak için 12 byte değişken üstbilgisi kullanır ve çoğu yayın için yalnızca 2 baytlık değişken üstbilgileri kullanır.

Üç hizmet niteliğiyle, düşük gecikme süresi ve güvenilirlik arasında ticaret yapabilirsiniz; bkz. MQTT istemcisi tarafından sağlanan hizmet nitelikleri. *Yangın ve unutun* , kalıcı aygıt depolaması kullanmaz ve bir yayını göndermek ya da almak için yalnızca bir iletim kullanır. *En az bir kez*ve *aynen bir kez* , iletişim kuralı durumunu korumak ve kabul edilinceye kadar bir iletiyi kaydetmek için aygıtta kalıcı depolama gerektirir.

Linux Windows AIX MQTT müşterileri

Bir MQTT istemci uygulaması, telemetri aygıtından sunucuya bağlanmaktan ve bilgileri sunucuya yayınlamadan sorumlu olur. ayrıca konulara abone olabilir, yayınları alabilir ve telemetri cihazını kontrol edebilir.

IBM MQ istemci uygulamalarından farklı olarak, MQTT istemci uygulamaları IBM MQ uygulamalarıdır. Bağlantı kurmak için kuyruk yöneticisi belirtmiyorlar. Bunlar belirli IBM MQ programlama arabirimlerini kullanmaları için sınırlı değildir. Bunun yerine, MQTT istemcileri MQTT 3 iletişim kuralını uygular. Kendi istemci kitaplığınızı, programlama dilindeki MQTT protocol 'a ve tercihinize göreplatform' e yazabilirsiniz. Bkz. IBM MQ Telemetry Transport biçimi ve iletişim kuralı.

To simplify writing MQTT client apps, use the C, Java, and JavaScript client libraries that encapsulate the MQTT protocol for a number of platforms. Bu kitaplıkları MQTT uygulamalarınıza dahil etseniz, tam işlevli bir MQTT istemcisi en az 15 satır kodu kadar kısa olabilir. MQTT istemci kitaplıkları, Eclipse Paho ve MQTT.org' da serbestçe kullanılabilir. Bkz. IBM MQ Telemetry Transport örnek programları.

The MQTT client app is always responsible for initiating a connection with a telemetry channel. Bağlantı bağlandıktan sonra, MQTT istemci uygulaması ya da bir IBM MQ uygulaması ileti değiş tokacı başlatabilir.

MQTT istemci uygulamaları ve IBM MQ uygulamaları, aynı konu kümesini yayınlar ve bunlara abone olur. Bir IBM MQ uygulaması, istemci uygulaması önce bir abonelik yaratmadan doğrudan bir MQTT istemcisi uygulamasına ileti gönderebilir. Bkz. MQTT istemcilerine ileti göndermek için dağıtımlı kuyruğa alma yapılandırılması.

MQTT istemci uygulamaları, telemetri kanalı kullanarak IBM MQ ' a bağlanır. Telemetri kanalı, MQTT ve IBM MQtarafından kullanılan farklı türde ileti türleri arasında bir köprü görevi görür. It creates publications and subscriptions in the queue manager on behalf of the MQTT client app. teletext kanalı, bir MQTT istemci uygulamasının abonelikleriyle eşleşen yayınları kuyruk yöneticisinden MQTT istemci uygulamasına gönderir.

Linux Windows AIX MQTT istemcisine ileti gönderme

IBM MQ applications can send MQTT v3 clients messages by publishing to subscriptions created by clients, or by sending messages directly. MQTT istemcileri, diğer istemciler tarafından abone olunan konulara yayınlanarak bir başkasına ileti gönderebilirler.

Bir MQTT istemcisi, IBM MQolanağından aldığı bir yayına abone olur.

Do the task, “IBM MQ Exploreristemcisinden MQTT istemci yardımcı programına bir ileti yayınlanıyor” sayfa 118 to send a publication from IBM MQ to an MQTT client.

Bir MQTT v3 istemcisinin ileti alması için standart yol, bir konuya ya da konu kümesine ilişkin bir abonelik yaratmasını sağlar. In the example code snippet, Şekil 45 sayfa 117, the MQTT client subscribes using the topic string `"MQTT Examples"`. An IBM MQ C application, Şekil 46 sayfa 117, publishes to the topic using the topic string `"MQTT Examples"`. Şekil 47 sayfa 118kod parçacısında, MQTT istemcisi geri çağırma yönteminde yayını alır: `messageArrived`.

IBM MQ ' in yayınları MQTT istemcilerinden gelen aboneliklere yanıt olarak göndermek için nasıl yapılandırılacağı hakkında daha fazla bilgi için bkz. Publishing a message in response to an MQTT client subscription.

Bir IBM MQ uygulaması, doğrudan bir MQTT istemcisine ileti gönderir

“IBM MQ Explorerkomutunu kullanarak bir MQTT istemcisine ileti gönderme” sayfa 122 içinden bir iletiyi doğrudan IBM MQ istemcisinden MQTT istemcisine göndermek içintaskgörevini yapın.

Bir MQTT istemcisine bu şekilde gönderilen bir iletinin istenmemiş ileti olarak çağrılır. MQTT v3 istemcileri, bir konu adı kümesiyle birlikte yayınlar olarak istenmeyen iletiler alırlar. Telemetri (MQXR) hizmeti, konu adını uzak kuyruk adına ayarlar.

For further information about how to configure IBM MQ to send messages directly to MQTT clients, see İstemciye doğrudan ileti gönderme.

Bir MQTT istemcisi bir ileti yayınlar

Bir MQTT v3 istemcisi, başka bir MQTT v3 istemcisi tarafından alınan bir iletiyi yayınlayabilir, ancak istenmemiş bir ileti gönderemez. Şekil 48 sayfa 118 kod parçacığı, Java' ta bir MQTT v3 istemcisinin bir iletiyi nasıl yayınlatacağını gösterir.

Belirli bir MQTT v3 istemcisine ileti göndermek için tipik bir örüntü, her istemci için kendi `ClientIdentifier`aboneliği yaratmaktadır. Do the task "Belirli bir MQTT v3 istemcisine ileti yayınlama" sayfa 124 to publish a message from one MQTT client to another MQTT client using `ClientIdentifier` as a topic string.

Örnek kod parçacıklar

Şekil 45 sayfa 117 içindeki kod parçacığı, Java ' ta yazılan bir MQTT istemcisinin bir abonelik yarattısını göstermektedir. Ayrıca, abonelikle ilgili yayınları almak için `messageArrived` geri bildirme yöntemine de gereksinim duyar.

```
String    clientId = String.format("%-23.23s",
                     System.getProperty("user.name") + "_" +
                     (UUID.randomUUID().toString())).trim()).replace('-', '_');
MqttClient  client = new MqttClient("localhost", clientId);
String topicString = "MQTT Examples";
int            QoS = 1;
client.subscribe(topicString, QoS);
```

Şekil 45. MQTT v3 istemci abonesi

Şekil 46 sayfa 117 içindeki kod parçacığı, C içinde yazılan bir IBM MQ uygulamasının bir yayını nasıl gönderdiğini gösterir. Kod parçacığı görevden alınır, Bir değişken konusuna yayınlayıcı yarat

```
/* Define and set variables to.defaults */
/* Omitted lines declaring variables    */
char * topicName   = ""
char * topicString = "MQTT Examples"
char * publication = "Hello world!";
do {
  MQCONN(qMgrName, &Hconn, &CompCode, &Reason);
  if (CompCode != MQCC_OK) break;
  td.ObjectType = MQOT_TOPIC;    /* Object is a topic             */
  td.Version = MQOD_VERSION_4;   /* Descriptor needs to be V4     */
  strncpy(td.ObjectName, topicName,  MQ_TOPIC_NAME_LENGTH);
  td.ObjectString.VSPtr = topicString;
  td.ObjectString.VSLength = (MQLONG)strlen(topicString);
  MQOPEN(Hconn, &td, MQOO_OUTPUT | MQOO_FAIL_IF_QUIESCING, &Hobj, &CompCode, &Reason);
  if (CompCode != MQCC_OK) break;
  pmo.Options = MQPMO_FAIL_IF_QUIESCING | MQPMO_RETAIN;
  MQPUT(Hconn, Hobj, &md, &pmo, (MQLONG)strlen(publication)+1, publication, &CompCode, &Reason);
  if (CompCode != MQCC_OK) break;
  MQCLOSE(Hconn, &Hobj, MQCO_NONE, &CompCode, &Reason);
  if (CompCode != MQCC_OK) break;
  MQDISC(&Hconn, &CompCode, &Reason);
} while (0);
```

Şekil 46. IBM MQ publisher

Yayın geldiğinde, MQTT istemcisi MQTT uygulama istemcisi `MqttCallback` sınıfına ilişkin `messageArrived` yöntemini çağırır.

```
public class CallBack implements MqttCallback {
  public void messageArrived(MqttTopic topic, MqttMessage message) {
    try {
      System.out.println("Message arrived: \"" + message.toString()
          + "\" on topic \"" + topic.toString() + "\"");
    } catch (Exception e) {
      e.printStackTrace();
    }
  }
// ... Other callback methods
}
```

Şekil 47. messageArrived yöntemi

Şekil 48 sayfa 118 , Şekil 45 sayfa 117içinde oluşturulan aboneliğe bir ileti yayınlayarak MQTT v3 yayınını gösterir.

```
      String          address = "localhost";
      String         clientId = String.format("%-23.23s",
                                System.getProperty("user.name") + "_" +
                                (UUID.randomUUID().toString())).trim()).replace('-', '_');
      MqttClient       client = new MqttClient(address, clientId);
      String      topicString = "MQTT Examples";
      MqttTopic         topic = client.getTopic(Example.topicString);
      String     publication  = "Hello world";
      MqttMessage     message = new MqttMessage(publication.getBytes());
      MqttDeliveryToken token = topic.publish(message);
```

Şekil 48. MQTT v3 istemcisi yayınlayıcısı

Linux Windows AIX IBM MQ Exploreristemcisinden MQTT istemci yardımcı programına bir ileti yayınlanıyor

Follow the steps in this task to publish a message using IBM MQ Explorer, and subscribe to it with the MQTT client utility. Ek bir görev, varsayılan iletim kuyruğunu SYSTEM.MQTT.TRANSMIT.QUEUEolarak ayarlamak yerine, kuyruk yöneticisi diğer adını nasıl yapılandıracağını gösterir.

Başlamadan önce

Bu görev, IBM MQ ve IBM MQ Explorerile ilgili bilgi sahibi olduğunuzu ve IBM MQ ve MQ Telemetry özelliğinin kurulu olduğunu varsayar.

Bu görev için kuyruk yöneticisi kaynaklarını yaratan kullanıcının yeterli yetkiye sahip olması gerekir. Gösteri amaçlı olarak, IBM MQ Explorer kullanıcı kimliğinin mqm grubunun üyesi olduğu varsayılır.

Bu görev hakkında

Bu görevde, IBM MQ ' ta bir konu oluşturun ve MQTT istemcisi yardımcı programını kullanarak konuya abone olun. When you publish to the topic using IBM MQ Explorer, the MQTT client receives the publication.

Yordam

Aşağıdaki görevlerden birini gerçekleştirin:

- MQ Telemetry' u kurdun, ancak henüz başlatmadınız. Şu görevi yapın: “Henüz tanımlı bir telemetri (MQXR) hizmeti ile görev başlat” sayfa 119.
- Daha önce IBM MQ telemetresi çalıştırdınız, ancak gösteriyi yapmak için yeni bir kuyruk yöneticisi kullanmak istiyorsunuz. Şu görevi yapın: “Henüz tanımlı bir telemetri (MQXR) hizmeti ile görev başlat” sayfa 119.

- Görevi, tanımlanmış telemetri kaynakları olmayan, var olan bir kuyruk yöneticisini kullanarak yapmak istiyorsunuz. **Örnek yapılanışı tanımla** sihirbazını çalıştırmak istemezseniz.

  a. Telemetreyi ayarlamak için aşağıdaki görevlerden birini gerçekleştirin:

  – Linux ve AIXüzerinde telemetri için kuyruk yöneticisi yapılandırılması
  – Windowsüzerinde telemetri kuyruk yöneticisi yapılandırılıyor

  b. Görevi yapın: “Görevi çalışan telemetri (MQXR) hizmetiyle başlat” sayfa 120
- Görevi önceden tanımlanmış telemetri kaynakları olan var olan bir kuyruk yöneticisini kullanarak yapmak istiyorsanız, şu görevi yapın: “Görevi çalışan telemetri (MQXR) hizmetiyle başlat” sayfa 120.

Sonraki adım

Bir iletiyi doğrudan istemci yardımcı programına göndermek için “IBM MQ Explorerkomutunu kullanarak bir MQTT istemcisine ileti gönderme” sayfa 122 yapın.

Henüz tanımlı bir telemetri (MQXR) hizmeti ile görev başlat

Kuyruk yöneticisi için örnek telemetri kaynakları tanımlamak üzere bir kuyruk yöneticisi yaratın ve **Örnek yapılanışı tanımla** komutunu çalıştırın. IBM MQ Explorerkullanarak bir ileti yayınlayın ve bu iletiyi MQTT istemci yardımcı programıyla abone olun.

Bu görev hakkında

When you set up sample telemetry resources using the **Örnek yapılanışı tanımla**, the wizard sets the guest user ID permissions. Konuk kullanıcı kimliğinin bu şekilde yetkilendirilmesi için dikkatli bir şekilde göz önünde bulundurun. `guest` on Windows, and `nobody` on Linux, are given permission to publish and subscribe to the root of the topic tree, and to put messages onto `SYSTEM.MQTT.TRANSMIT.QUEUE`.

Sihirbaz ayrıca, varsayılan iletim kuyruğunu `SYSTEM.MQTT.TRANSMIT.QUEUE`değerine ayarlar; bu da varolan bir kuyruk yöneticisiyle çalışan uygulamaları etkileyebilir. Telemetreyi yapılandırmak ve varsayılan iletim kuyruğunu kullanmamak için zahmetli, ancak zahmetli bir işlem olabilir; şu görevle ilgili takip yapın: “Kuyruk yöneticisi diğer adının kullanılması” sayfa 121. Bu görevde, var olan varsayılan iletim kuyruğuna müdahale olasılığının önüne geçebilmek için bir kuyruk yöneticisi yaratasınız.

Yordam

1. IBM MQ Explorerkomutunu kullanarak yeni bir kuyruk yöneticisi yaratın ve başlatın.
   a) `Queue Managers` klasörünü farenin sağ düğmesiyle tıklatın > **Yeni** > **Kuyruk yöneticisi ...**öğelerini seçin. Bir kuyruk yöneticisi adı yazın > **Son**.

   Bir kuyruk yöneticisi adı girin; örneğin, `MQTTQMGR`.
2. Telemetri (MQXR) hizmetini oluşturun ve başlatın ve örnek bir telemetri kanalı oluşturun.
   a) `Queue Managers\`*`QmgrName`*`\Telemetry` klasörünü açın.
   b) **Örnek yapılanışı tanımla ...** > **Son**düğmesini tıklatın.

   Launch MQTT Client Utility onay kutusunu imlenmiş olarak bırakın.
3. MQTT istemci yardımcı programını kullanarak `MQTT Example` için bir abonelik oluşturun.
   a) **Bağlan**'ı tıklayın.

   İstemci geçmişi bir `Connected` olayını kaydeder.
   b) **Abonelik \ Konu** alanına `MQTT Example` yazın > **Abone ol**.

   İstemci geçmişi bir `Subscribed` olayını kaydeder.
4. IBM MQiçinde `MQTTExampleTopic` oluşturun.
   a) **MQ Gezgini**> **Yeni** > **Konu**' da `Queue Managers\`*`QmgrName`*`\Topics` klasörünü sağ tıklatın.
   b) Type `MQTTExampleTopic` as the **Ad** > **Sonraki**.
   c) Type `MQTT Example` as the **Konu dizesi** > **Son**.

d) Alındı bildirimi penceresini kapatmak için **Tamam** düğmesini tıklatın.

5. Publish `Hello World!` to the topic `MQTT Example` using IBM MQ Explorer.
   a) IBM MQ Exploreriçindeki `Queue Managers\`*`QmgrName`*`\Topics` klasörünü tıklatın.
   b) `MQTTExampleTopic` > **Test yayını ...**seçeneğini sağ tıklatın.
   c) **İleti verileri** (.) > **İletiyi yayınla** > Switch to the MQTT Client Utility penceresine `Hello World!` yazın.

   İstemci geçmişi bir `Received` olayını kaydeder.

Görevi çalışan telemetri (MQXR) hizmetiyle başlat

Bir telemetri kanalı ve bir konu oluşturun. Kullanıcıya konuyu ve telemetri iletim kuyruğunu kullanması için yetki verin. IBM MQ Explorerkullanarak bir ileti yayınlayın ve bu iletiyi MQTT istemci yardımcı programıyla abone olun.

Başlamadan önce

Görevin bu sürümünde, bir kuyruk yöneticisi (*QmgrName*) tanımlanır ve çalışır. Telemetri (MQXR) hizmeti tanımlandı ve çalışıyor. Telemetri (MQXR) hizmeti el ile yaratılmış ya da **Örnek yapılanışı tanımla** sihirbazı çalıştırılarak yaratılmış olabilir.

Bu görev hakkında

In this task you configure an existing queue manager to send a publication to the MQTT client utility.

Görevin “1” sayfa 120 . adımı, varsayılan iletim kuyruğunu `SYSTEM.MQTT.TRANSMIT.QUEUE`değerine ayarlar; bu kuyruk, varolan bir kuyruk yöneticisiyle çalışan uygulamaları kesintiye neden olabilir. Telemetreyi yapılandırmak ve varsayılan iletim kuyruğunu kullanmamak için zahmetli, ancak zahmetli bir işlem olabilir; şu görevle ilgili takip yapın: “Kuyruk yöneticisi diğer adının kullanılması” sayfa 121.

Yordam

1. `SYSTEM.MQTT.TRANSMIT.QUEUE` değerini varsayılan iletim kuyruğu olarak ayarlayın.
   a) `Queue Managers\`*`QmgrName`* `folder` > **Özellikler ...**öğelerini sağ tıklatın.
   b) Gezginde **İletişim** seçeneğini tıklatın.
   c) Click **Seç ...** > Select `SYSTEM.MQTT.TRANSMIT.QUEUE` > **Tamam** > **Tamam**.
2. MQTT istemci yardımcı programını IBM MQ' e bağlamak ve MQTT istemcisi yardımcı programını başlatmak için bir telemetri kanalı `MQTTExampleChannel` oluşturun.
   a) **MQ Gezgini**> **Yeni** > **Telemetri kanalı ...**içindeki `Queue Managers\`*`QmgrName`* `\Telemetry\Channels` klasörünü farenin sağ düğmesiyle tıklatın.
   b) **Kanal adı** alanında `MQTTExampleChannel` yazın > **İleri** > **İleri**.
   c) İstemci yetkilendirme panosundaki **Sabit kullanıcı kimliği** 'yi, yayınlayacağı ve `MQTTExample` > **İleri**' ye abone olacak kullanıcı kimliğine çevirin.
   d) Leave **İstemci Yardımcı Programını Başlat** checked> **Son**.
3. MQTT istemci yardımcı programını kullanarak `MQTT Example` için bir abonelik oluşturun.
   a) **Bağlan**'ı tıklayın.

      İstemci geçmişi bir `Connected` olayını kaydeder.
   b) **Abonelik \ Konu** alanına `MQTT Example` yazın > **Abone ol**.

   İstemci geçmişi bir `Subscribed` olayını kaydeder.
4. IBM MQiçinde `MQTTExampleTopic` oluşturun.
   a) **MQ Gezgini**> **Yeni** > **Konu**' da `Queue Managers\`*`QmgrName`*`\Topics` klasörünü sağ tıklatın.
   b) Type `MQTTExampleTopic` as the **Ad** > **Sonraki**.

c) Type `MQTT Example` as the **Konu dizesi** > **Son**.

d) Alındı bildirimi penceresini kapatmak için **Tamam** düğmesini tıklatın.

5. If you want a user, not in the `mqm` group, to publish and subscribe to the `MQTTExample` topic, do the following:

   a) Authorize the user to publish and subscribe to the topic `MQTTExampleTopic`:

   ```
   setmqaut -m qMgrName -t topic -n MQTTExampleTopic -p User ID -all +pub +sub
   ```

   b) Kullanıcıya, `SYSTEM.MQTT.TRANSMIT.QUEUE`' e bir ileti yerleştirmesi için yetki verin:

   ```
   setmqaut -m qMgrName -t q -n SYSTEM.MQTT.TRANSMIT.QUEUE -p User ID -all +put
   ```

6. Publish `Hello World!` to the topic `MQTT Example` using IBM MQ Explorer.

   a) IBM MQ Exploreriçindeki `Queue Managers\`*`QmgrName`*`\Topics` klasörünü tıklatın.

   b) `MQTTExampleTopic` > **Test yayını ...**seçeneğini sağ tıklatın.

   c) **İleti verileri** (.) > **İletiyi yayınla** > Switch to the MQTT Client Utility penceresine `Hello World!` yazın.

İstemci geçmişi bir `Received` olayını kaydeder.

Kuyruk yöneticisi diğer adının kullanılması

Varsayılan iletim kuyruğunu `SYSTEM.MQTT.TRANSMIT.QUEUE`olarak ayarlamadan IBM MQ Explorer komutunu kullanarak MQTT istemcisi yardımcı programına bir ileti yayınlayın.

Görev, önceki görevin devamı ve varsayılan iletim kuyruğunu `SYSTEM.MQTT.TRANSMIT.QUEUE`olarak ayarlamamak için bir kuyruk yöneticisi diğer adı kullanıyor.

Başlamadan önce

Görevi, “Henüz tanımlı bir telemetri (MQXR) hizmeti ile görev başlat” sayfa 119 ya da görevi (“Görevi çalışan telemetri (MQXR) hizmetiyle başlat” sayfa 120) tamamlayın.

Bu görev hakkında

Bir MQTT istemcisi bir abonelik yarattığında, IBM MQ yanıtı uzak kuyruk yöneticisi adı olarak `ClientIdentifier`komutunu kullanarak gönderir. Bu görevde, `ClientIdentifier`(`MyClient`) olanağını kullanır.

If there is no transmission queue or queue manager alias called `MyClient`, the response is placed on the default transmission queue. By setting default transmission queue to `SYSTEM.MQTT.TRANSMIT.QUEUE`, the MQTT client gets the response.

Kuyruk yöneticisi diğer adlarını kullanarak, varsayılan iletim kuyruğunu `SYSTEM.MQTT.TRANSMIT.QUEUE` ' e ayarlamaktan kaçının. Her `ClientIdentifier`için bir kuyruk yöneticisi diğer adı ayarlamalısınız. Genellikle, kuyruk yöneticisi diğer adlarını kullanabilmek için çok fazla istemci kullanılır. Genellikle `ClientIdentifier` öngörülemez, telemetriyi bu şekilde yapılandırmanın imkansız hale getirmektedir.

Yine de, bazı durumlarda varsayılan iletim kuyruğunu `SYSTEM.MQTT.TRANSMIT.QUEUE`dışında bir değere yapılandırmanız gerekebilir. Yordam ' daki adımlar, varsayılan iletim kuyruğunu `SYSTEM.MQTT.TRANSMIT.QUEUE`olarak ayarlamak yerine bir kuyruk yöneticisi diğer adını yapılandırır.

Yordam

1. `SYSTEM.MQTT.TRANSMIT.QUEUE` öğesini varsayılan iletim kuyruğu olarak kaldırın.

   a) `Queue Managers\`*`QmgrName`* `folder` > **Özellikler ...**öğelerini sağ tıklatın.

   b) Gezginde **İletişim** seçeneğini tıklatın.

   c) `SYSTEM.MQTT.TRANSMIT.QUEUE` 'u **Varsayılan iletim kuyruğu** alanından > **Tamam**' ı kaldırın.

2. Artık MQTT istemci yardımcı programıyla bir abonelik oluşturamadığınızı denetleyin:
   a) **Bağlan**'ı tıklayın.

      İstemci geçmişi bir `Connected` olayını kaydeder.

   b) **Abonelik \ Konu** alanına `MQTT Example` yazın > **Abone ol**.

   İstemci geçmişi , bir `Subscribe failed` ve bir `Connection lost` olayını kaydeder.
3. `ClientIdentifier`(`MyClient`) için bir kuyruk yöneticisi diğer adı yaratın.
   a) `Queue Managers\`*`QmgrName`*`\Queues` klasörünü farenin sağ düğmesiyle tıklatın > **Yeni** > **Uzak kuyruk tanımı**.
   b) Tanımın adını girin, `MyClient` > **İleri**.
   c) **Uzak kuyruk yöneticisi** alanına `MyClient` yazın.
   d) **İletim kuyruğu** alanına `SYSTEM.MQTT.TRANSMIT.QUEUE` yazın > **Son**.
4. MQTT istemci yardımcı programını yeniden bağlayın.
   a) **İstemci tanıtıcısı** ' nın `MyClient`olarak ayarlı olup olmadığını denetleyin.
   b) **Bağlan**

   İstemci geçmişi bir `Connected` olayını kaydeder.
5. MQTT istemci yardımcı programını kullanarak `MQTT Example` için bir abonelik oluşturun.
   a) **Bağlan**'ı tıklayın.

      İstemci geçmişi bir `Connected` olayını kaydeder.

   b) **Abonelik \ Konu** alanına `MQTT Example` yazın > **Abone ol**.

   İstemci geçmişi bir `Subscribed` olayını kaydeder.
6. Publish `Hello World!` to the topic `MQTT Example` using IBM MQ Explorer.
   a) IBM MQ Exploreriçindeki `Queue Managers\`*`QmgrName`*`\Topics` klasörünü tıklatın.
   b) `MQTTExampleTopic` > **Test yayını ...**seçeneğini sağ tıklatın.
   c) **İleti verileri** (.) > **İletiyi yayınla** > Switch to the MQTT Client Utility penceresine `Hello World!` yazın.

   İstemci geçmişi bir `Received` olayını kaydeder.

Linux Windows AIX IBM MQ Explorerkomutunu kullanarak bir MQTT istemcisine ileti gönderme

IBM MQ Explorerkomutunu kullanarak IBM MQ kuyruğuna bir ileti koyarak, MQTT istemcisi yardımcı programına bir ileti gönderin. Bu görev, bir iletiyi doğrudan bir MQTT istemcisine göndermek için uzak kuyruk tanımlamasının nasıl yapılandırılacağı gösterilmektedir.

Başlamadan önce

Do the task, “IBM MQ Exploreristemcisinden MQTT istemci yardımcı programına bir ileti yayınlanıyor” sayfa 118. MQTT istemci yardımcı programını bağlıyken bırakın.

Bu görev hakkında

Bu görev, bir konuya yayınlamak yerine, kuyruğu kullanarak bir MQTT istemcisine ileti göndermeyi gösterir. İstemcide bir abonelik yaratmasınız. Görevin “2” sayfa 123 adımı, önceki aboneliğin silindiğini gösterir.

Yordam

1. MQTT istemci yardımcı programını bağlantısını kesip yeniden bağlayarak var olan abonelikleri atın.

The subscription is discarded because, unless you change the defaults, the MQTT client utility connects with a clean session; see Temizleme oturumları.

To make it easier to do the task, type your own `ClientIdentifier`, rather than use the generated `ClientIdentifier` created by the MQTT client utility.

a) MQTT istemci yardımcı programını telemetri kanalından çıkarmak için **Disconnect** (Bağlantıyı Kes) seçeneğini tıklatın.

İstemci Geçmişi bir `Disconnected` olayını kaydeder

b) **İstemci Tanıtıcıları** ' yı `MyClient`olarak değiştirin.

c) **Bağlan**'ı tıklayın.

İstemci Geçmişi bir `Connected` olayını kaydeder

2. MQTT istemci yardımcı programının `MQTTExampleTopic`için yayın aldığından emin olun.

a) IBM MQ Explorericindeki `Queue Managers\`*`QmgrName`*`\Topics` klasörünü tıklatın.

b) `MQTTExampleTopic` > **Test yayını ...**seçeneğini sağ tıklatın.

c) **İleti verileri** (.) > **İletiyi yayınla** > Switch to the MQTT Client Utility penceresine `Hello World!` yazın.

İstemci geçmişi' nde hiçbir olay kaydedilmez.

3. İstemci için bir uzak kuyruk tanımlaması yaratın.

Uzak kuyruk tanımlamasındaki uzak kuyruk yöneticisi adı olarak `ClientIdentifier`(İstemci Tanıtıcısı), `MyClient`(MyClient) olarak ayarlayın. Uzak kuyruk adı olarak istediğiniz adı kullanın. Uzak kuyruk adı, konu adı olarak bir MQTT istemcisine geçirilir.

a) `Queue Managers\`*`QmgrName`*`\Queues` klasörünü farenin sağ düğmesiyle tıklatın > **Yeni** > **Uzak kuyruk tanımı**.

b) Tanımın adını girin, `MyClientRemoteQueue` > **İleri**.

c) **Uzak kuyruk** alanına `MQTTExampleQueue` yazın.

d) **Uzak kuyruk yöneticisi** alanına `MyClient` yazın.

e) **İletim kuyruğu** alanına `SYSTEM.MQTT.TRANSMIT.QUEUE` yazın > **Son**.

4. Put a test message onto `MyClientRemoteQueue`.

a) **MyClientRemoteQueue** > **Sınama iletisini ekle ...**öğelerini sağ tıklatın.

b) İleti veri alanına `Hello queue!` yazın > **İleti koy** > **Kapat**

İstemci geçmişi bir `Received` olayını kaydeder.

5. `SYSTEM.MQTT.TRANSMIT.QUEUE` öğesini varsayılan iletim kuyruğu olarak kaldırın.

a) `Queue Managers\`*`QmgrName`* `folder` > **Özellikler ...**öğelerini sağ tıklatın.

b) Gezginde **İletişim** seçeneğini tıklatın.

c) `SYSTEM.MQTT.TRANSMIT.QUEUE` 'u **Varsayılan iletim kuyruğu** alanından > **Tamam**' ı kaldırın.

6. “4” sayfa 123adımını yeniden yap.

`MyClientRemoteQueue` , iletim kuyruğunu belirtik olarak adlarına alan bir uzak kuyruk tanımlamasıdır. `MyClient`'e bir ileti göndermek için varsayılan iletim kuyruğunu tanımlamak için a' ya gerek yoktur.

Sonraki adım

Varsayılan iletim kuyruğu artık `SYSTEM.MQTT.TRANSMIT.QUEUE`olarak ayarlanmadıysa, `ClientIdentifier`, `MyClient`için bir kuyruk yöneticisi diğer adı tanımlanmadıkça, MQTT Client Utility yeni bir abonelik yaratamamaktadır. Varsayılan iletim kuyruğunu `SYSTEM.MQTT.TRANSMIT.QUEUE`olarak geri yükleyin.

Linux Windows AIX Belirli bir MQTT v3 istemcisine ileti yayınlama

Birmessagev3 istemcisinden başka bir MQTT istemcisinden bir ileti yayınlayın; yayınlama/abone olma aracısı olarak `ClientIdentifier` konu adı ve IBM MQ olarakusingkomutunu kullanın.

Başlamadan önce

Do the task, “IBM MQ Exploreristemcisinden MQTT istemci yardımcı programına bir ileti yayınlanıyor” sayfa 118. MQTT istemci yardımcı programını bağlıyken bırakın.

Bu görev hakkında

Görev iki şeyi gösterir:

1. Bir MQTT istemcisinden bir konuya abone olma ve başka bir MQTT istemcisinden bir yayın alma.
2. Konu dizgisi olarak `ClientIdentifier` kullanılarak "noktadan noktaya iletişim" abonelikleri ayarlanıyor.

Yordam

1. MQTT istemci yardımcı programını bağlantısını kesip yeniden bağlayarak var olan abonelikleri atın.

   The subscription is discarded because, unless you change the defaults, the MQTT client utility connects with a clean session; see Temizleme oturumları.

   To make it easier to do the task, type your own `ClientIdentifier`, rather than use the generated `ClientIdentifier` created by the MQTT client utility.

   a) MQTT istemci yardımcı programını telemetri kanalından çıkarmak için **Disconnect** (Bağlantıyı Kes) seçeneğini tıklatın.

   İstemci Geçmişi bir `Disconnected` olayını kaydeder

   b) **İstemci Tanıtıcıları** ' yı `MyClient`olarak değiştirin.

   c) **Bağlan**'ı tıklayın.

   İstemci Geçmişi bir `Connected` olayını kaydeder
2. Create a subscription to the topic, `MyClient`

   `MyClient` , bu istemcinin `ClientIdentifier` ' dır.

   a) **Abonelik \ Konu** alanına `MyClient` yazın > **Abone ol**.

   İstemci geçmişi bir `Subscribed` olayını kaydeder.
3. Başka bir MQTT istemcisi yardımcı programı başlatın.

   a) `Queue Managers\`*`QmgrName`*`\Telemetry\channels` klasörünü açın.

   b) **PlainText** kanalını farenin sağ düğmesiyle tıklatın > **MQTT Client Utility programını çalıştır ...**

   c) **Bağlan**'ı tıklayın.

   İstemci Geçmişi bir `Connected` olayını kaydeder
4. Publish `Hello MyClient!` to the topic `MyClient`.

   a) Copy the subscription topic, `MyClient`, from the MQTT client utility running with the `ClientIdentifier`, `MyClient`.

   b) `MyClient` öğesini, her bir MQTT istemci yardımcı programı eşgörünümünün **Publication \ Topic** (Yayın) alanına yapıştırın.

   c) **Yayın \ iletisi** alanına `Hello MyClient!` yazın.

   d) Her iki örnekte de **Yayınla** öğesini tıklatın.

Sonuçlar

`ClientIdentifier`, `MyClient`ile MQTT istemci yardımcı programındaki **İstemci geçmişi** , iki **Alındı** olayını ve bir **Yayınlandı** olayını kaydeder. Diğer MQTT istemcisi yardımcı program örneği bir **Yayınlandı** olayı kaydeder.

Yalnızca bir **Alındı** olayı görüyorsanız, aşağıdaki olası nedenleri denetleyin:

1. Kuyruk yöneticisi için varsayılan iletim kuyruğu `SYSTEM.MQTT.TRANSMIT.QUEUE` olarak ayarlanmış mı?
2. Diğer alıştırmaları yaparken, kuyruk yöneticisi diğer adlarını ya da `MyClient` ile ilgili uzak kuyruk tanımlarını yarattın mı? Bir yapılandırma sorununuz varsa, kuyruk yöneticisi diğer adları ya da iletim kuyrukları gibi `MyClient`' un başvuruda bulunduğu kaynakları silin. İstemci yardımcı programlarının bağlantısını kesin, telemetri (MQXR) hizmetini durdurun ve yeniden başlatın.

Linux Windows AIX MQTT istemcisinden bir IBM MQ uygulamasına ileti gönderme

Bir IBM MQ uygulaması, bir konuya abone olarak MQTT v3 istemcisinden bir ileti alabilir. MQTT istemcisi, bir telemetri kanalı kullanarak IBM MQ ' a bağlanır ve aynı konu üzerinde yayınlanarak IBM MQ uygulamasına bir ileti gönderir.

Do the task, “Publishing a message to IBM MQ from an MQTT client” sayfa 125, to learn how to send a publication from an MQTT client to a subscription defined in IBM MQ.

Konu kümelenmişse ya da bir yayınlama/abone olma sıradüzeni kullanılarak dağıtılmış ise, abonelik, MQTT istemcisinin bağlı olduğu kuyruk yöneticisine farklı bir kuyruk yöneticisinden olabilir.

Linux Windows AIX Publishing a message to IBM MQ from an MQTT client

Create a subscription to a topic using IBM MQ Explorer and publish to the topic using MQTT client utility.

Başlamadan önce

Do the task, “IBM MQ Exploreristemcisinden MQTT istemci yardımcı programına bir ileti yayınlanıyor” sayfa 118. MQTT istemci yardımcı programını bağlıyken bırakın.

Bu görev hakkında

Bu görev, MQTT istemcisiyle bir iletinin yayınlanmasını ve IBM MQ Explorerkullanılarak yaratılmış yönetilmeyen bir kalıcı aboneliği kullanarak yayını ele aldığını gösterir.

Yordam

1. Create a durable subscription to the topic string `MQTT Example`.

   Kuyruğu oluşturmak için aşağıdaki adımları gerçekleştirin ve IBM MQ Explorerkomutunu kullanarak abonelik gerçekleştirin.

   a) IBM MQ Explorer> **Yeni** > **Yerel kuyruk ...**' da `Queue Managers\`*`QmgrName`*`\Queues` klasörünü sağ tıklatın.

   b) Kuyruk adı olarak `MQTTExampleQueue` yazın > **Son**.

   c) IBM MQ Explorer> **Yeni** > **Abonelik ...**içindeki `Queue Managers\`*`QmgrName`*`\Subscriptions` klasörünü sağ tıklatın.

   d) Kuyruk adı olarak `MQTTExampleSubscription` yazın > **İleri**.

   e) **Seç ...** > `MQTTExampleTopic` > **Tamam**seçeneklerini tıklatın.

   You have already created the topic, `MQTTExampleTopic` in step “4” sayfa 119 of “IBM MQ Exploreristemcisinden MQTT istemci yardımcı programına bir ileti yayınlanıyor” sayfa 118.

f) Hedef adı olarak MQTTExampleQueue yazın > **Son**.

2. As an optional step, set the queue up for use by a different user, without mqm authority.

   If you are setting up the configuration for users with less authority than mqm, you must give put and get authority to MQTTExampleQueue. Konuya ve iletim kuyruğuna erişim “IBM MQ Exploreristemcisinden MQTT istemci yardımcı programına bir ileti yayınlanıyor” sayfa 118içinde yapılandırılmıştı.

   a) Authorize a user to put and get to the queue MQTTExampleQueue:

   ```
   setmqaut -m qMgrName -t queue -n MQTTExampleQueue -p User ID -all +put +get
   ```

3. Publish Hello IBM MQ! to the topic MQTT Example using the MQTT client utility.

   MQTT istemcisi yardımcı programını bağlı bırakmadıysanız, **PlainText** kanalı > **MQTT Client Utility programını çalıştır ...** > **Bağlan**öğesini sağ tıklatın.

   a) **Publication \ Topic** (Yayın) alanına MQTT Example yazın.

   b) **Yayın \ İleti** alanına Hello IBM MQ! yazın > **Yayınla**.

4. Queue Managers*QmgrName*\Queues klasörünü açın ve MQTTExampleQueuedosyasını bulun.

   Yürürlükteki kuyruk derinliği alanı 1olur.

5. MQTTExampleQueue > **İletilere göz at ...**seçeneğini sağ tıklatın. ve yayını inceleyin.

Linux Windows AIX MQTT yayınlama/abone olma uygulamaları

MQTT uygulamalarını yazmak için konu tabanlı yayınlama/abone olma.

MQTT istemcisi bağlandığında, yayın akışı istemci ile sunucu arasında her iki yönde bir akış sağlar. Belgeler, istemcide yayınlandığında istemciden gönderilir. İstemci tarafından oluşturulan bir abonelikle eşleşen bir konuya ileti yayınlandığında, bu yayınlar istemcide alınır.

IBM MQ yayınlama/abone olma aracısı, MQTT istemcileri tarafından yaratılan konuları ve abonelikleri yönetir. MQTT istemcileri tarafından yaratılan konular, IBM MQ uygulamaları tarafından yaratılan konu alanıyla aynı konuyu paylaşır.

Publications that match the topic string in an MQTT client subscription are placed on SYSTEM.MQTT.TRANSMIT.QUEUE with the remote queue manager name set to the ClientIdentifier of the client. Telemetri (MQXR) hizmeti, yayınları, aboneliği yaratan istemciye iletir. İstemciyi tanıtmak için uzak kuyruk yöneticisi adı olarak ayarlanan ClientIdentifier(ClientIdentifier) işlevini kullanır.

Tipik olarak, SYSTEM.MQTT.TRANSMIT.QUEUE varsayılan iletim kuyruğu olarak tanımlanmalıdır. MQTT ' u varsayılan iletim kuyruğunu kullanmayacak şekilde yapılandırmak mümkün, ancak zahmetli olabilir; bkz. Configure distributed queuing to send messages to MQTT clients.

Bir MQTT istemcisi, kalıcı bir oturum oluşturabilir; bkz. “MQTT durumsuz ve durumlu oturumlar” sayfa 129. Kalıcı bir oturumda oluşturulan abonelikler dayanıklıdır. Kalıcı oturuma sahip bir istemci için gelen yayınlar SYSTEM.MQTT.TRANSMIT.QUEUEiçinde saklanır ve yeniden bağlandığında istemciye iletilir.

Bir MQTT istemcisi de tutulan yayınları yayınlayabilir ve bunlara abone olabilir; bkz. Alıkonan yayınlar ve MQTT istemcileri. Alıkonan bir yayın konusuna abone olan bir abone, konuya en son yayını alır. Abone, abonelik yarattığında ya da daha önceki oturumuna yeniden bağlandığında tutulan yayını alır.

Linux Windows AIX Telemetri uygulamaları

IBM MQ ya da IBM Integration Bus ileti akışlarını kullanarak telemetri uygulamaları yazın.

Use JMS, MQI, or other IBM MQ programming interfaces to program telemetry applications in IBM MQ.

Telemetri (MQXR) hizmeti, MQTT v3 iletileri ile IBM MQ iletileri arasında dönüştürme yapar. MQTT istemcileri adına abonelikler ve yayınlar oluşturur ve yayınları MQTT istemcilerine iletir. Bir yayın, bir MQTT v3 iletisinin bilgi yüküdür. The payload comprises message headers and a byte array in jms-bytes

format. Telemetri sunucusu, bir MQTT v3 iletisi ve bir IBM MQ iletisi arasındaki üstbilgileri eşler; bkz. “Kuyruk yöneticileriyle MQ Telemetry bütünleştirmesi” sayfa 127.

Yayınları IBM Integration Bus ve MQTT istemcileri arasında göndermek ve almak için Yayın, MQInput ve JMSInput düğümlerini kullanın.

İleti akışlarının kullanılması, HTTP kullanan web siteleriyle ve IBM MQ ve WebSphere Adapters kullanan diğer uygulamalarla telemetrileri bütünleştirebilirsiniz.

Linux Windows AIX Kuyruk yöneticileriyle MQ Telemetry bütünleştirmesi

MQTT istemcisi, yayınlama/abone olma uygulaması olarak IBM MQ ile tümleştirilmiştir. IBM MQiçindeki konulara yayınlama ya da abone olma, yeni konular yaratma ya da varolan konuları kullanma. It receives publications from IBM MQ as a result of MQTT clients, including itself, or other IBM MQ applications publishing to the topics of its subscriptions. Bir yayının özniteliklerine karar vermek için kurallar uygulanır.

Konular, yayınlar, abonelikler ve IBM MQtarafından sağlanan iletiler ile ilişkili birçok öznitelik desteklenmez. “MQTT client to IBM MQ publish/subscribe broker” sayfa 127 ve “IBM MQ istemcisine MQTT istemcisi” sayfa 128 , yayınların özniteliklerinin nasıl ayarlanacağını açıklar. Bu ayarlar, yayının IBM MQ yayınlama/abone olma aracısından mı, yoksa aboneye mi gittiğine bağlı olarak değişir.

IBM MQ yayınlama/abone olma konuları, yönetimle ilgili konu nesneleriyle ilişkilendirilir. MQTT istemcileri tarafından oluşturulan konular farklı değildir. When an MQTT client creates a topic string for a publication the IBM MQ publish/subscribe broker associates it with an administrative topic object. Aracı, yayındaki konu dizesini en yakın denetim konusu nesne üst nesnesine eşler. Eşleme, IBM MQ uygulamaları için aynıdır. Kullanıcı tarafından oluşturulan bir konu yoksa, yayın konusu `SYSTEM.BASE.TOPIC`ile eşlenir. Yayına uygulanan öznitelikler, konu nesnesinden türetilir.

Bir IBM MQ uygulaması ya da bir yönetici abonelik yarattığında, abonelik adını taşır. Abonelikleri IBM MQ Explorerkomutunu kullanarak ya da **runmqsc** ya da PCF komutlarını kullanarak listeleyin. Tüm MQTT istemci abonelikleri adlandırılır. Onlara formun adı verilir: *`ClientIdentifier:Topic name`*

MQTT client to IBM MQ publish/subscribe broker

Bir MQTT istemcisi, IBM MQ' e bir yayın gönderdi. Telemetri (MQXR) hizmeti, yayını bir IBM MQ iletisine dönüştürür. IBM MQ iletisi üç kısım içerir:

1. MQMD
2. RFH2
3. İleti

MQMD özellikleri, Çizelge 9 sayfa 127' ta belirtilenler dışında, varsayılan değerlerine ayarlanır.

Çizelge 9. MQMD özelliklerine ilişkin ayarlar

MQMD alanı	**Tip**	**Değer**
Format	MQCHAR8	MQFMT_RF_HEADER_2

UserIdentifier	MQCHAR12	Aşağıdakilerden birine ayarla:
		`MqttClient.ClientIdentifier`
		`MqttConnectOptions.UserName`
		Telemetri kanalı için IBM MQ yöneticisi tarafından ayarlanan bir kullanıcı kimliği.

Priority	MQLONG	MQPRI_PRIORITY_AS_Q_DEF (Varsayılan değer olan JMS için varsayılan değer olan 4 değerine sahip IBM MQiçin varsayılan değer.)

Çizelge 9. MQMD özelliklerine ilişkin ayarlar (devamı var)		
MQMD alanı	**Tip**	**Değer**
Persistence	MQLONG	QoS=0→MQPER_NOT_PERSISTENT QoS=1→MQPER_PERSISTENT QoS=2→MQPER_PERSISTENT

RFH2 üstbilgisi, JMS iletisinin tipini tanımlamak için bir <msd> klasörü içermez. Telemetri (MQXR) hizmeti, IBM MQ iletisini varsayılan bir JMS iletisi olarak yaratır. Varsayılan JMS ileti-tipi bir `jms-bytes` iletidir. Bir uygulama, ek üstbilgi bilgilerine ileti özellikleri olarak erişebilir; bkz. İleti özellikleri.

RFH2 değerleri, Çizelge 10 sayfa 128içinde gösterildiği şekilde ayarlanır. Biçim özelliği RFH2 sabit üstbilgisinde ayarlanır ve diğer değerler RFH2 klasörlerinde ayarlanır.

Çizelge 10. RFH2 özelliklerine ilişkin ayarlar		
RFH2 özelliği	**Tip/Klasör**	**Üstbilgi**
Biçim	MQCHAR8	MQFMT_NONE
ClientIdentifier	mqtt/clientId	Copy MqttClient.ClientIdentifier with a length of 1...23 bytes.
QoS	mqtt/qos	Gelen MQTT iletisinden QoS kopyasını kopyalayın.
İleti Tanıtıcısı	mqtt/msgid	Copy İleti Tanıtıcısı from incoming MQTT message, if QoS is 1 or 2.
MQIsRetained	mqps/Ret	Özgün MQTT yayınının RETAIN özellik kümesiyle gönderilip gönderilmediğini ve iletinin alıkonan bir yayın olarak gönderilip gönderilmediğini belirleyin.
MQTopicString	mqps/Top	MQTT iletisinin yayınlandığı konu.

Bir MQTT yayınındaki bilgi yükü, bir IBM MQ iletisinin içeriğiyle eşlenir:

Çizelge 11. MQTT yayınındaki bilgi yükünün IBM MQ ileti içeriğiyle nasıl eşleneceği		
İleti içeriği	**Tip**	**IBM MQ iletisinin içeriği**
Arabellek	MQBYTE *n*	Gelen MQTT iletisinden bayt kopyası. Uzunluk sıfır olabilir.

IBM MQ istemcisine MQTT istemcisi

Bir istemci bir yayın konusuna abone oldu. An IBM MQ application has published to the topic, resulting in a publication being sent to the MQTT subscriber by the IBM MQ publish/subscribe broker. Alternatif olarak, bir IBM MQ uygulaması, istenmeyen bir iletiyi doğrudan bir MQTT istemcisine göndermiştir. Çizelge 12 sayfa 129 , sabit ileti üstbilgilerinin MQTT istemcisine gönderilen iletiyle nasıl ayarlandığını açıklar. IBM MQ ileti üstbilgisindeki ya da diğer üstbilgilerdeki diğer veriler atılır. The message data in the IBM MQ message is sent as the message payload in the MQTT message, with no alteration. MQTT iletisi, telemetri (MQXR) hizmetidir MQTT istemcisine gönderilir.

Çizelge 12. How fixed message headers are set in an IBM MQ message sent to the MQTT client		
MQTT alan	**Tip**	**Değer**
DUP	boole	QoS = 1 ya da 2ise, önceki bir iletimde bu istemciye ileti gönderildiyse ve ileti bir süre sonra onaylanmadıysa bu ileti ayarlanır.
QoS	int	IBM MQ içindeki yayınlama/abone olma aracısından giden bir yayındaki QoS değerinin ayarlandığı yol, gelen yayına bağlıdır. Bu, gelen yayının bir MQTT istemcisinden mi, yoksa bir IBM MQ uygulamasından mı gönderilmesine bağlıdır. **MQTT** Gelen yayındaki QoS alt değeri ve abone tarafından istenen QoS ' de alt değer. **IBM MQ** Gelen yayından türetilen QoS değerinin alt değeri: MQPER_NOT_PERSISTENT→QoS=0 MQPER_PERSISTENT→QoS=2 ve abone tarafından istenen QoS ' dir. İleti istemciye abonelik olmadan gönderilirse, QoS varsayılan değer olarak 2 değerine ayarlanır. A client can alter this value by subscribing to DEFAULT.QoS with a different QoS.
RETAIN	boole	Gelen yayının alıkonan özellik kümesine sahip olup olmadığını ayarlayın.

Çizelge 13 sayfa 129 , değişken ileti üstbilgilerinin MQTT istemcisine gönderilen MQTT iletisinde nasıl ayarının belirlendiğini açıklar.

Çizelge 13. How MQTT variable header properties are set in an MQTT message sent to the MQTT client		
MQTT alan	**Tip**	**Değer**
Topic name	Dizgi	İletinin yayınlandığı konu dizgisi.
Message ID	Dizgi	The last 2 bytes of the MQMD.MsgId property of the publication when it is placed in SYSTEM.MQTT.TRANSMIT.QUEUE.
Payload	bayt []	Gelen yayınlardan yayınlama/abone olma aracısına, byte 'ın doğrudan kopyası. Uzunluk sıfır olabilir.

Linux Windows AIX MQTT durumsuz ve durumlu oturumlar

MQTT istemcileri kuyruk yöneticisiyle durumlu bir oturum oluşturabilirler. Durumlu bir MQTT istemcisi bağlantısı kesildiğinde, kuyruk yöneticisi istemci tarafından yaratılan abonelikleri ve uçuş içi iletileri korur. İstemci yeniden bağlandığında, uçuş sırasında bir ileti çözer. Teslim almak üzere kuyruğa alınan iletileri gönderir ve bağlantısı kesilirken abonelikleri için yayınlanan iletileri alır.

Bir MQTT istemcisi bir telemetri kanalına bağlandığında ya yeni bir oturum başlatır ya da eski bir oturuma devam eder. Yeni bir oturumda, onaylanmamış, abonelik yok ve teslim bekleyen yayınlar yok. Bir istemci bağlandığında, temiz bir oturumla başlayıp var olan bir oturuma devam edip etmeyeceğini belirtir; bkz. Temizleme oturumları.

İstemci varolan bir oturumu devam ettirirse, bağlantı kesilmemiş gibi devam eder. Teslim edilmeyi bekleyen yayınlar istemciye gönderilir ve kesinleştirilmemiş olan ileti aktarımları tamamlanır. Kalıcı bir

oturumdaki bir istemci, telemetri (MQXR) hizmetiden bağlantıyı kestiğinde, istemcinin yarattığı abonelikler kalır. Aboneliklere ilişkin yayınlar yeniden bağlandığında istemciye gönderilir. Eski oturuma devam etmeksizin yeniden bağlantı kesilirse, yayınlar telemetri (MQXR) hizmeti tarafından atılır.

Oturum durumu bilgileri, `SYSTEM.MQTT.PERSISTENT.STATE` kuyruğunda kuyruk yöneticisi tarafından saklanır.

IBM MQ yöneticisi bir oturumu kesebilir ve kalıcı olarak temizleyebilir.

Linux Windows AIX Bir MQTT istemcisi bağlanmadığında

Bir istemci bağlanmadığında, kuyruk yöneticisi kendi adına yayınları almaya devam edebilir. Bağlantı yeniden bağlandığında istemciye iletilirler. İstemci, istemci beklenmedik bir şekilde kesilirse, kuyruk yöneticisinin istemci adına yayınladığı bir "Son irade ve ahit"(son işlem ve vasiyet) yaratabilir.

İstemci beklenmeyen bir şekilde bağlantı kesildiğinde size bildirim almak istiyorsanız, son bir vasiyet ve ahit yayınını kaydedebilirsiniz; bkz. Son irade ve ahit yayını. Müşteri, istemcinin istekte bulunmadan bozulmasını algıladığında, telemetri (MQXR) hizmeti tarafından gönderilir.

Bir istemci alıkonan bir yayını herhangi bir zamanda yayınlayabilir; bkz. Alıkonan yayınlar ve MQTT istemcileri. Bir konuya ilişkin yeni bir abonelik, konumla ilişkili alıkonan herhangi bir yayını göndermeyi isteyebilir. Alıkonan bir yayın olarak son vasiyet ve vasiyetleri oluşturursanız, istemcinin durumunu izlemek için bu belgeyi kullanabilirsiniz.

Örneğin, istemci alıkonacağı bir yayını yayınlarken, bu yayını bağlarken, bu belgenin kullanılabilirliğini de yayınlamaktadır. Aynı zamanda, kullanılırlığını açıklayacak son bir irade ve ahit yayınını da oluşturur. Buna ek olarak, planlanan bir bağlantı oluşturmadan hemen önce, alıkonmamış bir yayın olarak unavailability yayınlanıyor. İstemcinin kullanılabilir olup olmadığını öğrenmek için, alıkonan yayın konusuna abone olabilirsiniz. Her zaman üç yayından birini alırsınız.

İstemci bağlantısı kesildiğinde yayınlanan iletileri alacaksa, istemciyi önceki oturumuna yeniden bağlayın; bkz. “MQTT durumsuz ve durumlu oturumlar” sayfa 129. Abonelikleri silininceye kadar ya da istemci temiz bir oturum yaratıncaya kadar etkin olur.

Linux Windows AIX MQTT istemcileri ile IBM MQ uygulamaları arasında gevşek bağlaşım

The flow of publications between MQTT clients and IBM MQ applications is loosely coupled. Yayınlar, bir MQTT istemcisinden ya da bir IBM MQ uygulamasından kaynaklanabilir ve herhangi bir ayarın ayarlanmaması olabilir. Yayıncılar ve aboneler gevşek bir şekilde birleşiyor. Yayınlar ve abonelikler aracılığıyla birbirleri ile dolaylı olarak etkileşim kurarlar. Ayrıca, iletileri bir IBM MQ uygulamasından bir MQTT istemcisine de doğrudan gönderebilirsiniz.

MQTT istemcileri ve IBM MQ uygulamaları, gevşek olarak iki algıyla birleşir:

1. Yayıncılar ve aboneler, bir yayının ve bir konuyla ilgili aboneliğin ilişkilendirmesiyle gevşek bir şekilde birleşir. Yayıncılar ve aboneler, normalde bir yayın ya da aboneliğin diğer kaynağının adreslerinden ya da kimliğinden haberdar değildir.
2. MQTT istemcileri ayrı iş parçacıklarına ilişkin yayınlama, abone olma, yayın alma ve teslim onayları işlemlerini yayınlar.

Bir MQTT istemcisi uygulaması, bir yayının teslim edilinceye kadar beklemez. Uygulama, MQTT istemcisine bir ileti iletir ve uygulama kendi iş parçacığıda devam eder. Uygulamayı, bir yayının teslimi ile eşitlemek için bir teslim simgesi kullanılır; bkz. Teslim simgeleri.

After passing a message to the MQTT client, the application has the choice of waiting on the delivery-token. Müşteri beklemek yerine, yayın IBM MQ' a teslim edildiğinde çağrılan bir geri bildirme yöntemi sağlayabilir. Ayrıca, teslim simgesini yoksayabilir.

İletiyle ilişkilendirilen hizmet kalitesine bağlı olarak, teslim belirteci hemen geri bildirme yöntemine ya da büyük olasılıkla hatırı sayılır bir süre sonra döndürülür. İstemcinin bağlantısı kesildikten ve yeniden bağlanmasından sonra, teslim simgesi de döndürülebiliyor. Hizmet kalitesi *ateşle ve unutursa*,

teslim simgesi hemen döndürülür. Diğer iki durumda, teslim simgesi yalnızca istemci, yayının abonelere gönderildiğini kabul ettiğinde döndürülür.

Publications sent to an MQTT client as a result of a client subscription, are delivered to the `messageArrived` callback method. `messageArrived` , ana uygulama için farklı bir iş parçacığıda çalışır.

Doğrudan bir MQTT istemcisine ileti gönderme

Belirli bir MQTT istemcisine iki yoldan bir ileti gönderebilirsiniz.

1. Bir IBM MQ uygulaması, aboneliği olmayan bir MQTT istemcisine doğrudan ileti gönderebilir; bkz. İstemciye doğrudan ileti gönderme.
2. Diğer bir yaklaşım, `ClientIdentifier` adlandırma kuralınızı kullanmandır. Tüm MQTT abonelerinin, benzersiz `ClientIdentifier` (ClientIdentifierClientIdentifierClientIdentifier) ClientIdentifier.topic(*ClientIdentifier*olarak yayınlayın. Yayın, *ClientIdentifier*konusuna abone olan istemciye gönderilir. Bu tekniği kullanarak, belirli bir MQTT abonesine bir yayın gönderebilirsiniz.

Linux Windows AIX MQ Telemetry güvenliği

cihazların taşınabilir olması muhtemel olduğu ve dikkatli bir şekilde kontrol edilemeyen yerlerde kullanıldığı için telemetrik cihazların güvenliğini sağlamak önemli olabilir. You can use VPN to secure the connection from the MQTT device to the telemetry (MQXR) service. MQ Telemetry , diğer iki güvenlik düzeneği, TLS ve JAAS' ı sağlar.

TLS, aygıt ile telemetri kanalı arasındaki iletişimi şifrelemek ve aygıtın doğru sunucuya bağlanmasını doğrulamak için kullanılan birincil kullanım olarak kullanılır; bkz. TLS kullanan telemetri kanalı kimlik doğrulaması. İstemci aygıtının sunucuya bağlanmasına izin verildiğini denetlemek için TLS 'yi de kullanabilirsiniz; bkz. TLS' yi kullananMQTT istemci kimlik doğrulaması.

JAAS , aygıt kullanıcısının bir sunucu uygulamasını kullanma izni olup olmadığını denetlemek için kullanılan birincil kullanıcıdır; bkz. Bir parola kullanarakMQTT istemci kimlik doğrulaması. JAAS , tek oturum açma dizini kullanarak bir parolayı denetlemek için LDAP ile birlikte kullanılabilir.

TLS ve JAAS , iki katsayılı kimlik doğrulaması sağlamak için birlikte kullanılabilir. TLS tarafından kullanılan şifrelemeleri, FIPS standartlarını karşılayan şifrelemelere sınırlayabilirsiniz.

En az on binlerce kullanıcıyla, bireysel güvenlik profillerini sağlamak her zaman pratik değildir. Ayrıca, tek tek kullanıcılara IBM MQ nesnelerine erişim yetkisi vermek için profilleri kullanmak her zaman pratik bir uygulamadır. Bunun yerine, yayınlara yetki vermek ve konulara abone olmak ve istemcilere yayınlar göndermek için kullanıcıları sınıflara gruplamak gerekir.

İstemcileri ortak istemci kullanıcı kimlikleriyle eşlemek için her telemetri kanalını yapılandırın. Belirli bir kanala bağlanan her istemci için ortak bir kullanıcı kimliği kullanın; bkz. MQTT istemci kimliği ve yetkilendirmesi.

Kullanıcı gruplarına yetki vermek, her bir bireyin kimlik doğrulamasını tehlikeye atmaz. Her bir kullanıcının kimliği doğrulanabilir; istemci ya da sunucuda, `Kullanıcı adı` ve `Parola`ve daha sonra, ortak bir kullanıcı kimliği kullanılarak sunucuda yetkilendirilebilirler.

Linux Windows AIX MQ Telemetry küreselleşme

MQTT v3 iletişim kuralındaki ileti bilgi yükü byte dizisi olarak kodlanır. Generally, applications handling text create the message payload in UTF-8. Telemetri kanalı, ileti bilgi yükünü UTF-8olarak tanımlıyor, ancak kod sayfası dönüştürme işlemi yapmaz. Yayın konu dizgisinin UTF-8olması gerekir.

Uygulama, alfabetik verilerin doğru kod sayfasına ve sayısal veriye doğru sayı kodlamasına dönüştürülmesinden sorumludur.

MQTT Java istemcisinin uygun bir `MqttMessage.toString` yöntemi vardır. Yöntem, ileti bilgi yükünü, genel olarak UTF-8olan yerel altyapı varsayılan karakter kümesinde kodlandığı şekilde işler. Bu, bilgi yükünü bir Java Dizgisine dönüştürür. Java , bir dizeyi yerel altyapı varsayılan karakter kümesi kullanılarak kodlanmış bayt dizisine dönüştüren `getBytes` String yöntemine sahiptir. İleti bilgi yükünde metin

alışverişi yapılan iki MQTT Java programı, aynı varsayılan karakter kümesine sahip platformlar arasında, UTF-8içinde kolayca ve etkili bir şekilde yapılır.

Altyapılardan birinin varsayılan karakter kümesi UTF-8değilse, uygulamaların ileti alışverişi için bir kural oluşturmaları gerekir. Örneğin, yayınlayıcı, `getBytes("UTF8")` yöntemini kullanarak bir dizgiden UTF-8 ' e dönüştürme belirtiyor. Bir iletinin metnini almak için, abonede iletinin UTF-8 karakter kümesinde kodlanmış olduğu varsayılır.

The telemetry (MQXR) service describes the encoding of all incoming publications from MQTT clients messages as being UTF-8. Bu ayarlar `MQMD.CodedCharSetId` - `UTF-8`ve `RFH2.CodedCharSetId` - `MQCCSI_INHERIT` ; bkz. “Kuyruk yöneticileriyle MQ Telemetry bütünleştirmesi” sayfa 127. Yayının biçimi `MQFMT_NONE`olarak ayarlanır; bu nedenle, kanal ya da `MQGET`ile dönüştürme gerçekleştirilemez.

Linux Windows AIX Performance and scalability of MQ Telemetry

Consider the following factors when managing large numbers of clients and improving scalability of MQ Telemetry.

Kapasite planlama

MQ Telemetryiçin performans raporlarına ilişkin bilgi için bkz. MQ Performans belgeleri.

Bağlantılar

Bağlantılarla ilgili maliyetler şunlardır:

- Bir bağlantının, işlemci kullanımı ve saati bakımından kendisini ayarlamanının maliyeti.
- Ağ maliyetleri.
- Bir bağlantı açık tutulduğunda, ancak onu kullanmayan bellek kullanılır.

İstemciler bağlı kaldığında oluşan fazladan bir yük vardır. Bir bağlantı açık tutulursa, TCP/IP akışları ve MQTT iletileri, bağlantının hala orada olup olmadığını kontrol etmek için ağı kullanır. Ayrıca, açık tutulan her istemci bağlantısı için sunucuda bellek de kullanılır.

Her dakika için birden çok ileti gönderiyorsanız, yeni bir bağlantı başlatma maliyetinden kaçınmak için bağlantınızı açık tutun. Her 10-15 dakikada bir ileti gönderiyorsanız, açık tutma maliyetinden kaçınmak için bağlantınızı düşürmeyi düşünün. TLS bağlantısını açık, ancak boşta duran dönemler için daha pahalı olduğu için daha uzun süre tutmak isteyebilirsiniz.

Buna ek olarak, istemcinin yeteneğini de göz önünde bulundurun. İstemcide bir depo ve iletme olanağı varsa, iletileri toplu olarak işleyebilir ve toplu işleri göndermek arasındaki bağlantıyı bırakabilirsiniz. Ancak, istemcinin bağlantısı kesildiyse, istemcinin sunucudan bir ileti alması olanaklı değildir. Bu nedenle, başvurunuzun amacı, karar üzerinde bir dayanaklık sağlar.

Sisteminizde birden çok ileti gönderen bir istemci varsa, örneğin, dosya aktarımları için, ileti başına sunucu yanıtı beklemeyin. Bunun yerine, tüm iletileri gönderin ve bunların tümünün alındığı sona erme edin. Diğer bir seçenek olarak, Hizmet Kalitesi (QoS) olanağını kullanın.

You can vary the QoS by message, delivering unimportant messages using QoS 0 and important messages using a QoS of 2. The message throughput can be around twice as high with a QoS of 0 than with a QoS of 2.

Adlandırma kuralları

Uygulamanızı birçok istemci için tasarlıyorsanız, etkili bir adlandırma kuralı uygulayın. Her bir istemciyi doğru `ClientIdentifier`ile eşlemek için `ClientIdentifier` (İstemci Tanıtıcısı) öğesini anlamlı bir şekilde yapın. İyi bir adlandırma kuralı, yöneticinin hangi istemcilerin çalıştırdığı çalışmasını kolaylaştırması sağlar. Bir adlandırma kuralı, yöneticinin IBM MQ Gezgini 'nde uzun bir istemci listesini süzgeçten geçirmesine ve sorun belirlenmesine yardımcı olur; bkz. İstemci tanıtıcısı.

Verim

Konu adlarının uzunluğu, ağ üzerinde akan byte sayısını etkiler. Yayınlama ya da abone olma sırasında, bir iletteki byte sayısı önemli olabilir. Bu nedenle, bir konu addaki karakter sayısını sınırlayın. When an MQTT client subscribes for a topic IBM MQ gives it a name of the form:

```
ClientIdentifier: TopicName
```

To view all of the subscriptions for an MQTT client, you can use the IBM MQ MQSC **DISPLAY** command:

```
DISPLAY SUB(' ClientID1:*')
```

Defining resources in IBM MQ for use by MQTT clients

Bir MQTT istemcisi, IBM MQ uzak kuyruk yöneticisine bağlanır. There are two basic methods for an IBM MQ application to send messages to an MQTT client: set the default transmission queue to SYSTEM.MQTT.TRANSMIT.QUEUE or use queue manager aliases. Çok sayıda MQTT istemcisi varsa, kuyruk yöneticisi için varsayılan iletim kuyruğunu tanımlayın. Varsayılan iletim kuyruğu ayarının kullanılması, denetim çalışmasının basitleştirilmesini sağlar; bkz. MQTT istemcilerine ileti göndermek için dağıtımlı kuyruğa alma yapılandırılması.

Abonelikleri önleyerek ölçeklenebilirliği geliştirme.

When an MQTT V3 client subscribes to a topic, a subscription is created by the telemetry (MQXR) service in IBM MQ. Abonelik, istemciye ilişkin yayınları SYSTEM.MQTT.TRANSMIT.QUEUE' ta yönlendirir. Her yayının iletim üstbilgisindeki uzak kuyruk yöneticisi adı, aboneliği yapan MQTT istemcisinin ClientIdentifier değerine ayarlıdır. Her biri kendi aboneliklerini yapan birçok istemci varsa, bu, IBM MQ yayınlama/abone olma kümesi ya da sıradüzeni boyunca birçok yetkili sunucu aboneliğinin sağlanmakta olduğunu sağlar. Yayınlama/abone olma özelliğini kullanmamakla ilgili bilgi için, ancak bunun yerine nokta tabanlı bir çözüm kullanmak için Doğrudan istemciye ileti göndermebaşlıklı konuya bakın.

Çok sayıda istemciyi yönetme

To support many concurrently connected clients, increase the memory available for the telemetry (MQXR) service by setting the JVM parameters **-Xms** and **-Xmx**. Aşağıdaki adımları izleyin:

1. Telemetri hizmeti yapılandırma dizininde java.properties dosyasını bulun; bkz. Windowsüzerinde telemetri (MQXR) hizmeti yapılandırma dizini ya da Linuxüzerinde telemetri hizmeti yapılandırma dizini .
2. Dosyadaki yönergeleri izleyin; eşzamanlı olarak bağlanan 50.000 istemci için 1 GB ' lik bir yığın bellek yeterli olur.

   ```
   # Heap sizing options - uncomment the following lines to set the heap to 1G
   #-Xmx1024m
   #-Xms1024m
   ```

3. java.properties dosyasındaki telemetri (MQXR) hizmetini çalıştıran JVM ' ye geçmek için diğer komut satırı bağımsız değişkenlerini ekleyin; bkz. JVM parametrelerinin telemetri (MQXR) hizmetine eklenmesi).

Linux'ta açık dosya tanımlayıcıları sayısını artırmak için, aşağıdaki satırları /etc/security/limits.conf/' e ekleyin ve yeniden oturum açın.

```
@mqm soft nofile 65000
@mqm hard nofile 65000
```

Her yuva bir dosya tanımlayıcısı gerektirir. telemetri hizmeti bazı ek dosya tanımlayıcıları gerektirmektedir. bu nedenle bu sayı, gerekli açık yuva sayısından daha büyük olmalıdır.

Kuyruk yöneticisi, kalıcı olmayan her abonelik için bir nesne tanıtıcısı kullanır. Pek çok etkin özelliği desteklemek için, kalıcı olmayan abonelikler kuyruk yöneticisinde etkin çekme noktalarının üst sınır sayısını artırır; örneğin:

```
echo ALTER QMGR MAXHANDS(999999999) | runmqsc qMgrName
```

Şekil 49. Windowsüzerindeki tanıtıcı sayısı üst sınırını değiştir

```
echo "ALTER QMGR MAXHANDS(999999999)" | runmqsc qMgrName
```

Şekil 50. Linuxüzerindeki tanıtıcı sayısı üst sınırını değiştir

Diğer önemli noktalar

Sistem gereksinimlerinizi planlarken, sistemi yeniden başlatmak için gereken süreyi göz önünde bulundurun. Planlı kapalı kalma sürelerinin, kuyruğun işlenmesini bekleyen ileti sayısı üzerinde etkileri olabilir. Sistem konfigürasyonunun kabul edilebilir bir zamanda başarıyla işlenebilmesi için sistemi yapılandırın. Disk depolama, bellek ve işleme gücünü gözden geçirin. Bazı istemci uygulamalarıyla, istemci yeniden bağlandığında iletileri atmak da mümkün olabilir. İletileri atmak için, istemci bağlantısı parametrelerinde `CleanSession` (CleanSession) seçeneğini belirleyin; bkz. Clean seanss. Alternatif olarak, bir MQTT istemcisinde en iyi Hizmet Kalitesi (`0`) olanağını kullanarak yayınlayın ve abone olun; bkz. Hizmet Kalitesi. IBM MQ' tan ileti gönderirken `kalıcı olmayan` iletileri kullanın. Bu hizmet nitelikleri olan iletiler, sistem ya da bağlantı yeniden başlatıldığında kurtarılmaz.

Linux Windows AIX MQ Telemetrytarafından desteklenen aygıtlar

MQTT istemcileri, algılayıcılardan ve çalıştırıcılardan, tutulan aygıtları ve araç sistemlerini ele geçirmek için bir dizi aygıt üzerinde çalıştırılabilir.

MQTT istemcileri küçüktür ve küçük bellek ve düşük işleme gücü ile kısıtlanmış aygıtlarda çalıştırılır. MQTT protocol güvenilir ve düşük bant genişliğine, yüksek maliyete ve aralıklı kullanılabilirliğe göre kısıtlanan ağlara uygun küçük üstbilgiler içerir.

MQ Telemetry , telemetri aygıtlarıyla MQTT istemci uygulamaları aracılığıyla iletişim kurar. Bu uygulamalar, tüm MQTT v3 iletişim kuralını uygulayan aşağıdaki kaynakları kullanır:

- Aşağıdaki istemci kitaplıkları:
  - (örneğin) Android, OS X, Linux ya da Windows aygıtları için yerel uygulamalar oluşturmak için kullanılan *MQTT client for Java*. Bu istemci kitaplığını kullanan uygulamalar, CDC (Connected Device Configuration) /Foundation, J2SE (Java Platform, Standard Edition) ve J2EE (Java Platform, Enterprise Edition) aracılığıyla en küçük CLDC (Connected Limited Device Configuration) /MIDP (Mobile Information Device Profile) /MIDP (Connected Limited Device Configuration) /MIDP (Mobile Information Device Profile; Mobil Bilgi Aygıtı Profili) içinden tüm Java uyarlamalarında çalıştırılabilir. IBM jclRM uyarlanmış sınıf kitaplığı da desteklenir. Java ME platformu, genellikle çalıştırıcılar, algılayıcılar, cep telefonları ve diğer yerleşik aygıtlar gibi küçük aygıtlarda kullanılır. Java SE altyapısı, genellikle masaüstü bilgisayarlar ve sunucular gibi daha yüksek uç yerleştirilmiş aygıtlarda kurulur.
  - (örneğin,) iOS, OS X, Linux ya da Windows aygıtları için yerel uygulamalar oluşturmak için kullanılan *MQTT client for C*. Bu istemci kitaplığı, Windows ve Linux sistemleri için önceden oluşturulmuş yerel istemciyle birlikte bir C başvurusu somutlaması sağlar. C başvuru uygulaması, MQTT ' in çok çeşitli aygıt ve platformlara bildirilmesini sağlar. Some Windows systems on Intel, including Windows 7, RedHat, Ubuntu, and some Linux systems on ARM platforms such as Eurotech Viper, implement versions of Linux that run the C client, but IBM does not provide service support for the platforms. IBM destek merkezinizi aramayı amaçlıyorsanız, istemciyle desteklenen bir altyapıyla ilgili sorunları yeniden üretmeniz gerekir.
  - Tarayıcı tabanlı web uygulamaları oluşturmak için kullanılan *MQTT client for Java*.

MQTT istemci kitaplıkları, Eclipse Paho ve MQTT.org' da serbestçe kullanılabilir. Bkz. IBM MQ Telemetry Transport örnek programları.

IBM MQiçinde güvenlik

IBM MQ' ta güvenlik sağlamak için çeşitli yöntemler vardır: yetkilendirme hizmeti arabirimi; kullanıcı tarafından yazıldı ya da üçüncü kişi, kanal çıkışları; Transport Layer Security (TLS) kullanılarak kanal güvenliği, kanal kimlik doğrulama kayıtları ve ileti güvenliği.

Yetkilendirme hizmeti arabirimi

MQI çağrılarını, komutlarını ve nesnelere erişimi kullanma yetkisi, varsayılan olarak geçerli kılındığı **nesne yetkisi yöneticisi** (OAM) tarafından sağlanır. IBM MQ ' a erişim, IBM MQ kullanıcı grupları ve OAM aracılığıyla denetlenir. Yöneticiler, yetkileri gerektiği şekilde vermek ya da iptal etmek için bir komut satırı arabirimi kullanabilir.

Yetkilendirme hizmeti bileşenleri oluşturma hakkında daha fazla bilgi için bkz. AIX, Linux, and Windows sistemlerindeki güvenliğin ayarlanması.

Kullanıcı tarafından yazılan ya da üçüncü kişi kanal çıkışları

Kanallar, kullanıcı tarafından yazılan ya da üçüncü kişi kanal çıkışlarını kullanabilir. Ek bilgi için İleti alışverişi kanallarına ilişkin kanal çıkışı programlarıbaşlıklı konuya bakın.

TLS ' yi kullanarak kanal güvenliği

TLS (Transport Layer Security; İletim Katmanı Güvenliği) protokolü, gizlice dinleme, kurcalama ve kimliğine bürünme karşı koruma sağlayan, endüstri standardlı kanal güvenliği sağlar.

TLS, ileti gizliliği ve bütünlüğü ve karşılıklı kimlik doğrulaması sağlamak için ortak anahtar ve simetrik teknikler kullanır.

TLS ' ye ilişkin ayrıntılı bilgiler de dahil olmak üzere, IBM MQ içinde güvenlik hakkında kapsamlı bir inceleme için bkz. Securing. Bu kısımda açıklanan komutlara ilişkin işaretçiler de içinde olmak üzere, TLS ' ye ilişkin genel bilgiler için Cryptographic security protocols: TLSbaşlıklı konuya bakın.

Kanal doğrulama kayıtları

Kanal düzeyinde bağlantı kurmak için verilen erişim üzerinde kesin denetime sahip olmak için kanal kimlik doğrulama kayıtlarını kullanın. Ek bilgi için Kanal doğrulama kayıtlarıbaşlıklı konuya bakın.

İleti güvenliği

Use Advanced Message Security, which is a separately installed and licensed component of IBM MQ, to provide cryptographic protection to messages sent and receive using IBM MQ. Bkz. Advanced Message Security.

İlgili görevler

güvenlik

Güvenlik gereksinimlerinizin planlanması

IBM MQ.NET yönetilen istemci TLS desteği

The IBM MQ.NET fully managed client provides Transport Layer Security (TLS) support that is based on the Microsoft.NET SSLStreams kit. Bu, GSKit 'i temel alan diğer IBM MQ istemcilerinden farklıdır.

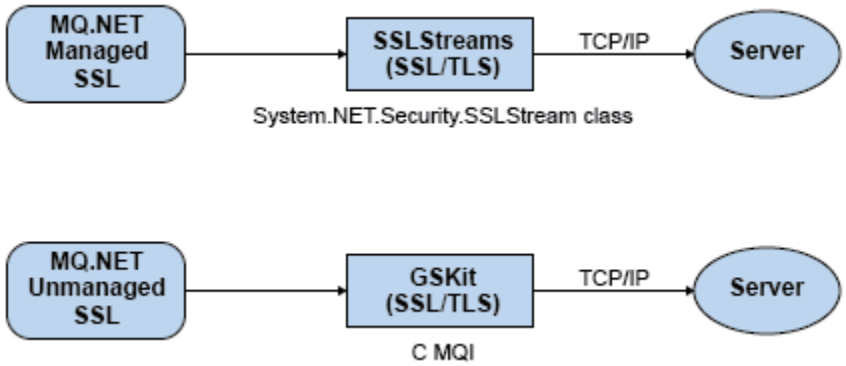
Yönetilen kipte ya da yönetilmeyen kipte çalışmak için IBM MQ.NET uygulamalarını geliştirebilirsiniz.

- Yönetilen kipte, .NET uygulamaları, C MQI ' ı çağırma gibi herhangi bir çapraz platform çağrısı olmadan .NET CLR (Common Language Runtime) içinde çalışır.

- Yönetilmeyen kipte, temeldeki MQI işlemleri için C MQI çağrılır. Temel olarak, yönetilmeyen kip arabirimi, C MQI ' nin üstündeki .NET sarıcı sınıflarından oluşur.

Yönetilen IBM MQ.NET istemcisi, TLS güvenli yuva iletişim kurallarını uygulamak için Microsoft.NET Framework kitaplıklarını kullanır. The System.NET.Security.SSLStream class from Microsoft is used for implementing Security (TLS) in IBM MQ.NET.

Yönetilmeyen IBM MQ.NET istemci kipi, C MQI ' ye (ve GSKit) dayalı olan TLS özelliğini zaten destekler. Yani, TLS işlemleri C MQI tarafından işlenmektedir. Bu durumda, GSKit TLS güvenli yuva protokollerini uygular.



Şekil 51. IBM MQ.NET yönetilen ve yönetilmeyen TLS karşılaştırması

Aşağıdaki çizelge, yönetilen ve yönetilmeyen somutlamalar arasındaki farkları özetlemektedir:

Çizelge 14. Yönetilen ve yönetilmeyen somutlamalar arasındaki farklar

Kip	İletişim kuralları	Uygulama	Açıklamalar
IBM MQ.NET yönetilen SSL	TLS	Sistem.NET.Security.SSL Stream sınıfı SSLStream sınıfı, bağlı bir TCP yuvasının üzerinde bir akış olarak çalışır	TLS 1.0 TLS 1.2 (yalnızca Microsoft.NET Framework v4.5 ile)
IBM MQ.NET unmanaged SSL	TLS	GSKIT ve C-MQI	TLS güvenli yuva iletişim kuralları

İlgili kavramlar

.NETiçin Secure Sockets Layer (SSL) ve Transport Layer Security (TLS) desteği

İstemciler ve sunucular

IBM MQ ' in uygulamaları için istemci-sunucu yapılandırmalarını nasıl desteklediği hakkında bir giriş.

IBM MQ MQI *istemcisi* , bir sistemde çalışan bir uygulamanın MQI yayınlanmasını sağlayan bir bileşendir. Bu, başka bir sistemde çalışan bir kuyruk yöneticisine çağrı sağlar. Çağrıdan gelen çıkış istemciye geri gönderilir ve bu işlem uygulamaya geri gönderilir.

Bir IBM MQ *sunucusu* , bir ya da daha çok istemciye kuyruğa alma hizmetleri sağlayan bir kuyruk yöneticidir. Tüm IBM MQ nesneleri (örneğin, kuyruklar), istemcide değil, yalnızca kuyruk yöneticisi makinesinde bulunur (IBM MQ sunucu makinesi). Bir IBM MQ sunucusu, yerel IBM MQ uygulamalarını da destekleyebilir.

Bir IBM MQ sunucusu ile sıradan bir kuyruk yöneticisi arasındaki fark, bir sunucunun her bir istemciyle özel olarak ayrılmış bir iletişim bağlantısının olması. İstemciler ve sunucular için kanal oluşturma hakkında daha fazla bilgi için bkz. Dağıtılmış kuyruğa alma yapılandırması.

Genel olarak istemcilerle ilgili bilgi için bkz. “IBM MQ MQI clients' a genel bakış” sayfa 137.

Bir istemci-sunucu ortamındakiIBM MQ uygulamaları

İstemci IBM MQ uygulamaları bir sunucuya bağlandığında, birçok MQI çağrısını yerel uygulamalarla aynı şekilde yayınlayabilir. İstemci uygulaması, belirtilen bir kuyruk yöneticisine bağlanmak için bir MQCONN çağrısı yayınlar. Bağlanma isteğinden döndürülen bağlantı tanıtıcısını belirten ek MQI çağrıları, bu kuyruk yöneticisi tarafından işlenir.

Uygulamalarınızı uygun istemci kitaplıklarına bağlamanız gerekir. Bkz. IBM MQ MQI clientsiçin uygulamalar oluşturma.

İlgili kavramlar

“İşlem yönetimi ve desteği” sayfa 143
İşlem yönetimine ilişkin bir giriş ve IBM MQ ' in hareketleri nasıl desteklediği.

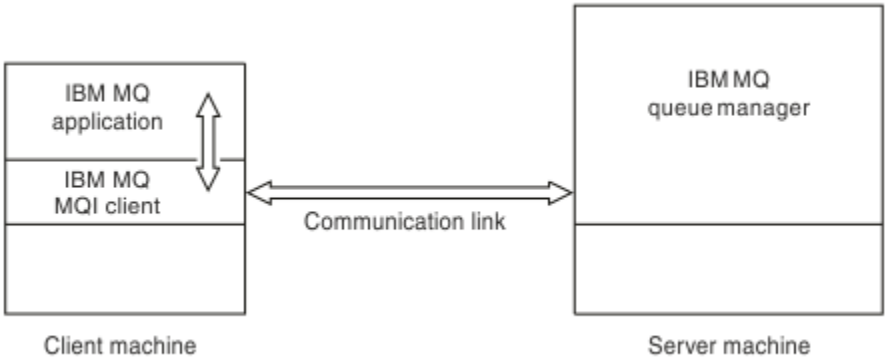
“Kuyruk yöneticisi olanaklarının genişletmesi” sayfa 145
Kuyruk yöneticisi olanaklarını kullanıcı çıkışlarını, API çıkışlarını ya da kurulabilir hizmetleri kullanarak genişletebilirsiniz.

IBM MQ MQI clients' a genel bakış

An IBM MQ MQI client is a component of the IBM MQ product that can be installed on a system on which no queue manager runs.

Bir IBM MQ MQI clientkullanılarak, istemciyle aynı sistemde çalışan bir uygulama, başka bir sistemde çalışan bir kuyruk yöneticisine bağlanabiliyor. Uygulama, o kuyruk yöneticisine yönelik MQI çağrılarını yayınlayabilir. Bu tür bir uygulamaya IBM MQ MQI client uygulaması adı verilir ve kuyruk yöneticisine *sunucu kuyruk yöneticisi*adı verilir.

Bir IBM MQ MQI client uygulaması ve sunucu kuyruğu yöneticisi, bir *MQI kanalı*kullanarak birbirleriyle iletişim kurar. Bir MQI kanalı, istemci uygulaması kuyruk yöneticisine bağlanmak için bir **MQCONN** ya da **MQCONNX** çağrısı yayınlarken başlar ve istemci uygulaması, kuyruk yöneticisinden bağlantıyı kesmek için bir **MQDISC** çağrısı yayınlarken sona erer. Bir MQI kanalının giriş değiştirgelerinin giriş değiştirgeleri, bir MQI kanalının ve çıkış değiştirgelerinin ters yöndeki akışıdır.



Şekil 52. İstemci ile sunucu arasındaki bağlantı

Aşağıdaki altyapılar kullanılabilir. Birleşimler, kullanmakta olduğunuz IBM MQ ürününe ve “IBM MQ istemcileri için platform desteği” sayfa 140' ta anlatılanlara bağlıdır.

IBM MQ MQI client	IBM MQ sunucu
AIX and Linux Windows IBM i	AIX and Linux Windows IBM i z/OS

MQI, istemci altyapısında çalışan uygulamalar için kullanılabilir; kuyruklar ve diğer IBM MQ nesneleri, bir sunucuya kurduğunuz bir kuyruk yöneticisinden yapılır.

Önce IBM MQ MQI client ortamında çalıştırmak istediğiniz bir uygulamanın ilgili istemci kitaplıkla bağlantısı olması gerekir. When the application issues an MQI call, the IBM MQ MQI client directs the request to a queue manager, where it is processed and from where a reply is sent back to the IBM MQ MQI client.

Uygulama ile IBM MQ MQI client arasındaki bağlantı yürütme sırasında dinamik olarak kurulur.

You can also develop client applications using the IBM MQ classes for .NET, IBM MQ classes for Java or IBM MQ classes for Java Message Service (JMS). You can use Java and JMS clients on the following platforms:

- IBM i
- AIX
- Linux
- Windows

Java ve JMS kullanımı burada açıklanmamaktadır. For full details on how to install, configure, and use IBM MQ classes for Java and IBM MQ classes for JMS see IBM MQ classes for Javakomutunu kullanma and IBM MQ classes for JMSkomutunu kullanma.

İlgili kavramlar

“Neden IBM MQ istemcileri kullanılsın?” sayfa 138
IBM MQ istemcilerinin kullanılması, IBM MQ ileti sisteminin ve kuyruğa alma yöntemlerinin verimli bir şekilde gerçekleştirilmesini sağlar.

“IBM MQ MQI clientnasıl ayarlanacak” sayfa 140
Bir istemci kurmak için bu yönergeleri izleyin.

“Genişletilmiş işlemsel istemci nedir?” sayfa 141
An IBM MQ extended transactional client can update resources managed by another resource manager, under the control of an external transaction manager.

“İstemcinin sunucuya nasıl bağlandığı” sayfa 142
Bir istemci, MQCONN ya da MQCONNX komutunu kullanarak bir sunucuya bağlanır ve bir kanalla iletişim kurar.

Neden IBM MQ istemcileri kullanılsın?

IBM MQ istemcilerinin kullanılması, IBM MQ ileti sisteminin ve kuyruğa alma yöntemlerinin verimli bir şekilde gerçekleştirilmesini sağlar.

Bir makinede çalışan bir MQI ' yi ve farklı bir makinede çalışan kuyruk yöneticisini (fiziksel ya da sanal) kullanan bir uygulama olabilir. Bunu yapmanın yararları şunlardır:

- İstemci makinesinde tam bir IBM MQ uygulamasına gerek yoktur.
- İstemci sistemindeki donanım gereksinimleri azaltılır.

- Sistem denetimi gereksinimleri azaltılır.
- Bir istemcide çalışan bir IBM MQ uygulaması, farklı sistemlerdeki birden çok kuyruk yöneticisine bağlanabilir.
- Farklı iletim protokollerini kullanan alternatif kanallar kullanılabilir.

İlgili başvurular

"What applications run on an IBM MQ MQI client?" sayfa 139
İstemci ortamında tam MQI desteklenir.

"IBM MQ istemcileri için platform desteği" sayfa 140
Desteklenen tüm sunucu platformlarındaIBM MQ , bir dizi altyapıda IBM MQ MQI clients ' den istemci bağlantılarını kabul eder.

What applications run on an IBM MQ MQI client?

İstemci ortamında tam MQI desteklenir.

This enables almost any IBM MQ application to be configured to run on an IBM MQ MQI client system by linking the application on the IBM MQ MQI client to the MQIC library, rather than to the MQI library. Kural dışı durumlar şunlardır:

- MQGET with signal
- Diğer kaynak yöneticileriyle tutarlılık noktası eşgüdümü gerektiren bir uygulama, genişletilmiş bir işlemsel istemci kullanmalıdır

Önceden okuma etkinleştirilirse, kalıcı olmayan ileti sistemi performansını geliştirmek için tüm MQGET seçenekleri kullanılabilir değildir. Çizelge, izin verilen seçenekleri ve bunların MQGET çağrıları arasında değiştirilip değiştirilmediklerini gösterir.

Çizelge 15. İleriyi okurken izin verilen MQGET seçenekleri geçerli kılındığında

Değerler	**Önceden okuma etkinleştirildiğinde ve MQGET çağrıları arasında değiştirilebilir olduğunda izin verilir**	**Okuma önden okuma etkinleştirildiğinde izin verilir, ancak MQGET çağrıları arasında değiştirilemez [1]**	**Önceden okuma etkinleştirildiğinde izin verilmeyen MQGET seçenekleri etkinleştirilir [2]**
MQGET MD değerleri	MsgId [3] CorrelId [3]	Kodlama CodedCharSetId	
MQGET MQGMO seçenekleri	MQGMO_BEKLE MQGMO_NO_BEKLEME MQGMO_FAIL_IF_QUIESCING MQGMO_BROWSE_FIRST [4] MQGMO_BROWSE_NEXT [4] MQGMO_BROWSE_MESSAGE _UNDER_CURSOR [4]	MQGMO_SYNCPOINT_IF_PERSISTEN T MQGMO_NO_SYNCPOINT MQGMO_ACCEPT_TRUNCATED_MSG MQGMO_CONVERT MQGMO_LOGICAL_ORDER MQGMO_COMPLE_MSG MQGMO_ALL_MSGS_AVALABILIR MQGMO_ALL_SEGMENTS_AVALABIL IR MQGMO_MARK_BROWSE_HANDLE MQGMO_MARK_BROWSE_CO_OP MQGMO_UNMARK_BROWSE_CO_OP MQGMO_UNMARK_BROWSE_HANDL E MQGMO_UNMARKET_BROWSE_MSG MQGMO_PROPERTIES_FORCE_MQR FH2 MQGMO_NO_ÖZELLİKLERİ MQGMO_PROPERTIES_IN_HANDLE MQGMO_PROPERTIES_COMPATIBILI TY	MQGMO_SET_SIGNAL MQGMO_SYNCPOINT MQGMO_MARK_SKIP _BACKUT MQGMO_MSG_ALT _CURSOR [4] MQGMO_LOCK MQGMO_UNLOCK
MQGMO değerleri		MsgHandle	

1. MQGET çağrıları arasında bu seçenekler değiştirilirse, bir MQRC_OPTIONS_CHANGED neden kodu döndürülür.

2. Bu seçenekler ilk MQGET çağrısında belirtilirse, önden okuma geçersiz kılınır. Sonraki bir MQGET çağrısında bu seçenekler belirtilirse, MQRC_OPTIONS_ERROR neden kodu döndürülür.
3. İstemci uygulamalarının, MsgId ve CorrelId değerleri önceki değerlerle MQGET çağrıları iletileri arasında değiştirilirse, istemciye önceden gönderilmiş olabileceğini ve tüketilinceye (ya da otomatik olarak temizleninceye) kadar istemcide önden okuma arabelleğinde kalacağını bilmeleri gerekir.
4. İlk MQGET çağrısı, önden okuma etkinleştirildiğinde iletilere göz atılacağını ya da kuyruktan ileti alınacağını belirler. Uygulama göz atma ve alma birleşimi kullanmayı denerse, MQRC_OPTIONS_CHANGED neden kodu döndürülür.
5. MQGMO_MSG_UNDER_CURSOR, önden okuma ile olanaklı değil. Önden okuma etkinleştirildiğinde iletilere göz atılabilir ya da iletiler alınabilir, ancak her ikisinin birleşimi olamaz.

Bir IBM MQ MQI client üzerinde çalışan bir uygulama, koşutzamanlı olarak birden çok kuyruk yöneticisine bağlanabilir ya da bir MQCONN ya da MQCONNX çağrısında yıldız (*) ile bir kuyruk yöneticisi adı kullanabilir (Connecting IBM MQ MQI client applications to queue managers içindeki örneklere bakın).

IBM MQ istemcileri için platform desteği

Desteklenen tüm sunucu platformlarındaIBM MQ , bir dizi altyapıda IBM MQ MQI clients ' den istemci bağlantılarını kabul eder.

Desteklenen tüm sunucu platformlarında *Temel ürün ve Sunucu* olarak kurulanIBM MQ , aşağıdaki altyapılarda IBM MQ MQI clients ' den bağlantıları kabul edebilir:

- IBM i
- AIX
- Linux
- Windows

İstemci bağlantıları, kodlanmış karakter takımı tanıtıcısı (CCSID) ve iletişim protokolündeki farklılıklara tabi olur.

IBM MQ MQI clientnasıl ayarlanacak

Bir istemci kurmak için bu yönergeleri izleyin.

Bir IBM MQ MQI client ayarlamak için, kurulu bir IBM MQ sunucusunun kurulu olması ve istemcinin bağlanacağı bir çalışmanız olmalıdır. İstemci ayarlarında yer alan adımlar şunlardır:

1. IBM MQ MQI client için uygun bir platformun olup olmadığını ve donanımın ve yazılımın gereksinimleri karşıladığını doğrulayın. Platform desteği “IBM MQ istemcileri için platform desteği” sayfa 140içinde açıklanmıştır.
2. IBM MQ ' u istemci iş istasyonunuza nasıl kuracağınıza karar verin ve daha sonra, istemci ve sunucu platformlarının özel birleşiminize ilişkin yönergeleri izleyin. Kuruluş aşağıdaki konularda açıklanır:
   - AIX AIXüzerinde bir IBM MQ istemcisinin kurulması
   - Linux Linuxüzerinde bir IBM MQ istemcisi kurulması
   - Windows Windowsüzerinde bir IBM MQ istemcisi kurulması
   - IBM i IBM iüzerinde bir IBM MQ istemcisi kurulması
3. İletişim bağlantılarınızın yapılandırıldığından ve bağlandığından emin olun. İletişim bağlantılarının yapılandırılması, Sunucu ile istemci arasındaki bağlantıların yapılandırılmasıbaşlıklı konu altında açıklanmıştır.
4. Kuruluşunuzun doğru biçimde çalıştığından emin olun. Kurumsal kullanımınızın platformuna ya da platformlarına ilişkin kuruluş yordamınızın doğrulama bölümüne bakın.

5. Doğrulanmış IBM MQ MQI client kurulumuna sahip olduğunda, istemcinizin güvenliğini sağlayıp sağlamadığınızı göz önünde bulundurun. Müşteri güvenliği, IBM MQ MQI client güvenliğini ayarlamakonusunda açıklanmaktadır.
6. Set up the channels between the IBM MQ MQI client and server that are required by the IBM MQ applications you want to run on the client. Kanalları ayarlama MQI kanallarını tanımlamabaşlıklı konu altında açıklanmıştır. TLS kullanıyorsanız dikkat edilmesi gereken bazı noktalar vardır. Bu dikkat edilmesi gereken noktalar, Bir MQI kanalının TLS kullandığını belirtmebaşlıklı konuda açıklanmaktadır. Kanalları ayarlamak için bir IBM MQ MQI client yapılandırma dosyası ya da IBM MQ ortam değişkeni kullanmanız gerekebilir. IBM MQ ortam değişkenleri IBM MQ ortam değişkenlerinin kullanılmasıiçinde açıklanmıştır.
7. IBM MQ uygulamaları Uygulamaların geliştirilmesiiçinde tam olarak açıklanmıştır.
8. There are some differences from a queue manager environment to consider when designing, building, and running applications in the IBM MQ MQI client environment. Bu farklılıklara ilişkin bilgi için aşağıdaki başlıklara bakın:
   - Bir istemci uygulamasında ileti kuyruğu arabiriminin (MQI) kullanılması
   - IBM MQ MQI clientsiçin uygulama oluşturma
   - IBM MQ MQI client uygulamalarının kuyruk yöneticilerine bağlanması
   - IBM MQ MQI clientsile ilgili sorunların çözülmesi

Genişletilmiş işlemsel istemci nedir?

An IBM MQ extended transactional client can update resources managed by another resource manager, under the control of an external transaction manager.

Hareket yönetimi kavramlarına aşina değilseniz, “İşlem yönetimi ve desteği” sayfa 143başlıklı konuya bakın.

XA işlemsel istemcisinin artık IBM MQ' nin bir parçası olarak sağlandığı unutulmamaktadır.

Bir istemci uygulaması, bağlı olduğu bir kuyruk yöneticisi tarafından yönetilen bir iş birimine katılabilir. İş birimi içinde, istemci uygulaması ileti alabilir ve kuyruk yöneticisinin sahibi olduğu kuyruklara ileti alabilir. Daha sonra istemci uygulaması, iş birimini onaylamak için **MQCMIT** çağrısını ya da iş birimini yedeklemek için **MQBACK** çağrısını kullanabilir. Ancak aynı iş birimi içinde, istemci uygulaması başka bir kaynak yöneticisinin kaynaklarını (örneğin, bir Db2 veritabanı çizelgeleri) güncelleyemez. IBM MQ genişletilmiş bir işlemsel istemci kullanılması bu kısıtlamayı kaldırır.

An IBM MQ extended transactional client is an IBM MQ MQI client with some additional function. Bu işlev, aynı iş birimi içinde bir istemci uygulaması kullanılarak aşağıdaki görevleri gerçekleştirebilirler:

- İleti içeren ve bağlı olduğu kuyruk yöneticisinin sahip olduğu kuyruklar için iletileri girin ve bu kuyruklardan ileti alın
- IBM MQ kuyruk yöneticisi dışındaki bir kaynak yöneticisinin kaynaklarını güncelleyin.

Bu iş birimi, istemci uygulamasıyla aynı sistemde çalışan bir dış hareket yöneticisi tarafından yönetilmelidir. İş birimi, istemci uygulamasının bağlı olduğu kuyruk yöneticisinden yönetilemiyor. Bu, kuyruk yöneticisinin bir hareket yöneticisi olarak değil, yalnızca kaynak yöneticisi olarak işlev görebileceği anlamına gelir. Ayrıca, istemci uygulamasının iş birimini, yalnızca dış hareket yöneticisi tarafından sağlanan uygulama programlama arabirimini (API) kullanarak işleyebileceği ya da geri alabileceği anlamına gelir. İstemci uygulaması bu nedenle, MQI çağrılarını, **MQBEGIN**, **MQCMIT**ve **MQBACK**adlı MQI çağrılarını kullanamaz.

Dış hareket yöneticisi, kuyruk yöneticisine bağlı istemci uygulaması tarafından kullanılan MQI kanalını kullanarak, kuyruk yöneticisiyle kaynak yöneticisi olarak iletişim kurar. Ancak, bir hata sonrasında kurtarma durumunda, hiçbir uygulama çalışmadığında, hareket yöneticisi, hata sırasında kuyruk yöneticisinin katıldığı tamamlanmamış iş birimlerini kurtarmak için özel olarak ayrılmış bir MQI kanalı kullanabilir.

Bu bölümde, genişletilmiş işlemsel işleve sahip olmayan bir IBM MQ MQI client , IBM MQ temel istemcisi olarak anılır. You can consider, therefore, an IBM MQ extended transactional client to consist of an IBM MQ base client with the addition of the extended transactional function.

Not: IBM i üzerindeIBM MQ MQI client , IBM MQ genişletilmiş işlemsel işlevini desteklemez.

İlgili başvurular

“Genişletilmiş işlemsel istemciler için platform desteği” sayfa 142
Genişletilmiş işlemsel istemciler, temel istemciyi destekleyen tüm çoklu platformlar için kullanılabilir. İstemciler z/OSiçin kullanılamaz.

Multi *Genişletilmiş işlemsel istemciler için platform desteği*

Genişletilmiş işlemsel istemciler, temel istemciyi destekleyen tüm çoklu platformlar için kullanılabilir. İstemciler z/OSiçin kullanılamaz.

Genişletilmiş bir işlemsel istemci kullanan bir istemci uygulaması, yalnızca aşağıdaki IBM MQ 9.0 ürünlerinin bir kuyruk yöneticisine bağlanabiliyor:

- AIX IBM MQ for AIX
- IBM i IBM MQ for IBM i
- Linux IBM MQ - Linux
- Windows IBM MQ for Windows

z/OS z/OSüzerinde çalışan genişletilmiş işlem istemcileri olmamasına rağmen, genişletilmiş bir işlemsel istemci kullanan bir istemci uygulaması, z/OSüzerinde çalışan bir kuyruk yöneticisine bağlanabilir.

Her platform için, genişletilmiş işlemsel istemciye ilişkin donanım ve yazılım gereksinimleri, IBM MQ temel istemcisi için bu gereksinimlerle aynıdır. Bir programlama dili, IBM MQ temel istemcisi ve kullanmakta olduğunuz hareket yöneticisi tarafından destekleniyorsa, genişletilmiş bir işlemsel istemci tarafından desteklenir.

For information about the external transaction managers for all platforms, see IBM MQ.

İstemcinin sunucuya nasıl bağlandığı

Bir istemci, MQCONN ya da MQCONNX komutunu kullanarak bir sunucuya bağlanır ve bir kanalla iletişim kurar.

IBM MQ istemcisi ortamında çalışan bir uygulama, istemci ile sunucu makineleri arasında etkin bir bağlantı sağlamalıdır.

Bağlantı, bir MQCONN ya da MQCONNX çağrısını yayınlayan bir uygulama tarafından yapılır. İstemciler ve sunucular, *MQI kanalları*aracılığıyla iletişim kurar ya da paylaşım konuşmalarını kullanırken, her biri bir MQI kanalı yönetim ortamını paylaşan sohbetler. Çağrı başarılı olduğunda, uygulama bir MQDISC çağrısı yayınlanıncaya kadar, MQI kanalı yönetim ortamı ya da etkileşimi bağlı kalır. Bu, bir uygulamanın bağlanması gereken her kuyruk yöneticisine ilişkin bir vakaya neden olur.

İlgili kavramlar

“Aynı makineden istemci ve kuyruk yöneticisi” sayfa 143
Ayrıca, makineniz de kuyruk yöneticisi kurulu olduğunda, bir uygulamayı IBM MQ MQI client ortamında çalıştırabilirsiniz.

“Farklı platformlardaki istemciler” sayfa 143
Burada, IBM MQ MQI client ve sunucu sistemine ilişkin başka bir örnek de vardır. Bu örnekte, sunucu makinesi farklı platformlarda üç IBM MQ MQI clients ile iletişim kurar.

“İstemci ve sunucu yazılımının farklı sürümlerini kullanma” sayfa 143
IBM MQ ürünlerinin önceki sürümlerini kullanıyorsanız, istemcinizin CCSID 'sinden kod dönüşümünün sunucu tarafından desteklendiğinden emin olun.

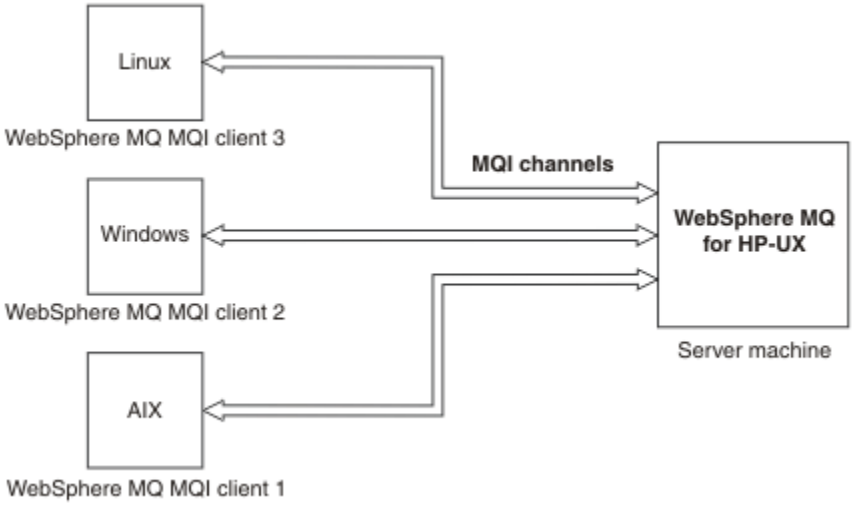
Aynı makineden istemci ve kuyruk yöneticisi

Ayrıca, makineniz de kuyruk yöneticisi kurulu olduğunda, bir uygulamayı IBM MQ MQI client ortamında çalıştırabilirsiniz.

Bu durumda, kuyruk yöneticisi kitaplıklarına ya da istemci kitaplıklarına bağlanma seçeneğiniz vardır; ancak, istemci kitaplıklarına bağlantı kurarsanız, kanal bağlantılarını tanımlamanız gerektiğini unutmayın. Bu, bir uygulamanın geliştirme aşaması sırasında yararlı olabilir. Kendi makinenizdeki programınızı test edebilir, diğer kullanıcılara bağımlılık yapabilir ve bu programı bağımsız bir IBM MQ MQI client ortamına taşıdığınızda işe yarayacağından emin olabilirsiniz.

Farklı platformlardaki istemciler

Burada, IBM MQ MQI client ve sunucu sistemine ilişkin başka bir örnek de vardır. Bu örnekte, sunucu makinesi farklı platformlarda üç IBM MQ MQI clients ile iletişim kurar.



Şekil 53. Farklı platformlardaki istemcilere bağlıIBM MQ sunucusu

Diğer karmaşık ortamlar da mümkündür. Örneğin, bir IBM MQ istemcisi birden çok kuyruk yöneticisine ya da kuyruk paylaşım grubunun bir parçası olarak bağlı kuyruk yöneticilerine ya da birden çok kuyruk yöneticisine bağlanabilir.

İstemci ve sunucu yazılımının farklı sürümlerini kullanma

IBM MQ ürünlerinin önceki sürümlerini kullanıyorsanız, istemcinizin CCSID 'sinden kod dönüşümünün sunucu tarafından desteklendiğinden emin olun.

IBM MQ istemcisi, kuyruk yöneticisinin desteklenen tüm sürümlerine bağlanabilir. Daha önceki bir sürüm kuyruk yöneticisine bağlanıyorsanız, istemcideki IBM MQ uygulamanızda ürünün daha sonraki bir sürümündeki özellikleri ve yapıları kullanamazsınız.

Bir IBM MQ kuyruk yöneticisi, karşılıklı olarak desteklenen en yüksek protokol düzeyine kadar anlaşarak farklı sürümlerdeki istemcilerle iletişim kurabilir. Bu, daha eski istemcilerin daha sonraki kuyruk yöneticisi düzeyleriyle kullanılabileceği anlamına gelir. Hem istemci hem de sunucunun, sorun tanılamayı kolaylaştırmak ve IBMtarafından desteği etkinleştirmek için şu anda desteklenmekte olan IBM MQ sürümlerinde olması önerilir.

Daha fazla bilgi için Uygulamaları geliştirmealanında desteklenen programlama dillerine bakın.

İşlem yönetimi ve desteği

İşlem yönetimine ilişkin bir giriş ve IBM MQ ' in hareketleri nasıl desteklediği.

Kaynak yöneticisi , uygulamalar tarafından erişilebilen ve güncellenebilen kaynaklara sahip olan ve kaynakları yöneten bir bilgisayar altsistemidir. Kaynak yöneticilerine ilişkin örnekler:

- Kuyrukları olan kaynaklarla birlikte bir IBM MQ kuyruk yöneticisi

- Tablolarını içeren kaynakları olan bir Db2 veritabanı

Bir uygulama bir ya da daha çok kaynak yöneticisinin kaynaklarını güncellediğinde, belirli güncellemelerin bir grup olarak başarıyla tamamlandığından ya da bunların hiçbiri tamamlandığından emin olmak için bir iş gereksinimi olabilir. Bu tür bir gereksinimin nedeni, bu güncellemelerin bazıları başarıyla tamamlanırsa, iş verilerinin tutarsız durumda bırakılmasına neden olur, ancak diğerleri bu tür bir gereksinimin tutarsız durumda bırakılmasına neden olur.

Bu şekilde yönetilen kaynaklara ilişkin güncellemeler, *iş birimi*ya da bir *işlem*içinde gerçekleşeceği söyleniyor. Bir uygulama programı, bir dizi güncelleme kümesini bir iş biriminde gruplayabilir.

Bir iş birimi sırasında, bir uygulama, kaynak yöneticilerine kaynaklarını güncelleştirmelerini ister. Uygulama, tüm güncellemeleri kesinleştirmek için bir istek gönderdiğinde iş birimi sona erer. Güncellemeler kesinleştirilinceye kadar, bunların hiçbiri aynı kaynaklara erişen diğer uygulamalar tarafından görülemez. Diğer bir seçenek olarak, uygulama herhangi bir nedenle iş birimini tamamlayamayacağına karar verirse, bu noktaya kadar talep ettiği tüm güncellemeleri geri almak için bir istek yayınlayabilir. Bu durumda, güncellemelerin hiçbiri diğer uygulamalar tarafından görülemez. Bu güncellemeler genellikle mantıksal olarak ilişkilidir ve veri bütünlüğünün korunması için başarılı olması gerekir. Bir güncelleme başarılı olursa, veri bütünlüğü kaybedilir.

Bir iş birimi başarıyla tamamlandığında, *commit*(kesinleştirme) değeri gelir. Kesinleştirdikten sonra, bu çalışma birimi içinde yapılan tüm güncellemeler kalıcı ve geri çevrilemez hale getirilmektedir. Ancak, iş birimi başarısız olursa, tüm güncellemeler yerine *yedeklenir*olur. İş birimlerinin bütünlük ile kesinleştirildiği ya da yedekleneceği bu işlem, *tutarlılık noktası eşgüdümü*olarak bilinir.

The point in time when all the updates within a unit of work are either committed or backed out is called a *eşitleme noktası*. An update within a unit of work is said to occur *eşitleme noktası denetimi içinde*. Bir uygulama, *eşitleme noktası denetimi dışında*bir güncelleme isteği isterse, devam etmekte olan bir iş birimi olsa bile, kaynak yöneticisi güncellemeyi hemen kesinleştirir ve güncelleme daha sonra yedeklenemez.

İş birimlerini yöneten bilgisayar altsistemi *transaction manager*ya da bir *nokta koordinatörü*olarak adlandırılır.

yerel iş birimi, güncellenen tek kaynakların IBM MQ kuyruk yöneticilerinden biri olduğu bir iş birimidir. Burada eşitleme noktası eşgüdümü, tek aşamalı kesinleştirme işlemi kullanılarak kuyruk yöneticisinin kendisi tarafından sağlanır.

genel iş birimi, XA uyumlu veritabanları gibi diğer kaynak yöneticilerine ait kaynakların da güncellendiği bir iş birimidir. Burada, iki aşamalı kesinleştirme yordamı kullanılmalıdır ve iş birimi, kuyruk yöneticisinin kendisi tarafından ya da IBM TXSeriesya da BEA Tuxedo gibi başka bir XA uyumlu hareket yöneticisi tarafından dışsal olarak eşgüdümlü olarak kullanılabilir.

Bir işlem yöneticisi, bir iş birimindeki kaynaklarla ilgili tüm güncellemelerin başarıyla tamamlanmasını ya da bunların hiçbirinin tamamlamasını sağlamaktan sorumludur. Bir uygulama, bir uygulamanın kesinleştirme ya da iş birimini geri alma isteğini içeren bir hareket yöneticisindir. Examples of transaction managers are CICS and WebSphere Application Server, although both of these possess other function as well.

Bazı kaynak yöneticileri kendi işlem yönetimi işlevini sağlar. For example, an IBM MQ queue manager can manage units of work involving updates to its own resources and updates to Db2 tables. Kuyruk yöneticisinin bu işlevi gerçekleştirmek için ayrı bir hareket yöneticisine gerek yoktur; ancak, bir kullanıcı gereksinmesi söz ettiyse, bu işlev kullanılabilir. Ayrı bir hareket yöneticisi kullanılırsa, *dış hareket yöneticisi*olarak anılır.

Dış hareket yöneticisinin bir iş birimini yönetmesi için, hareket yöneticisi ile iş birimine katılan her kaynak yöneticisi arasında standart bir arabirim olması gerekir. Bu arabirim, hareket yöneticisinin ve kaynak yöneticisinin birbiriyle iletişim kurmasını sağlar. Bu arabirimlerden biri, bir dizi hareket yöneticisi ve kaynak yöneticisi tarafından desteklenen standart bir arabirim olan *XA Arabirimi*' dir. The XA Interface is published by The Open Group in *Dağıtılmış İşlem İşlemesi: XA Belirtimi*.

Bir iş birimine birden çok kaynak yöneticisi katıldığında, bir hareket yöneticisinin, bir sistem hatası olsa bile, iş birimi içindeki tüm güncellemelerin başarıyla tamamlandığından ya da bunların hiçbiri tamamlanamadığından emin olmak için *iki aşamalı kesinleştirme* iletişim kuralı kullanması gerekir. Bir

uygulama, bir iş birimini kesinleştirmek için bir hareket yöneticisine istekte bulunduğunda, işlem yöneticisi şunları yapar:

Aşama 1 (Kesinleştirmek üzere hazırla)
Hareket yöneticisi, iş birimine katılan her kaynak yöneticisinde, amaçlanan güncellemelerle ilgili tüm bilgilerin kurtarılabilir bir durumda olduğundan emin olmasını ister. Bir kaynak yöneticisi olağan durumda, bilgileri bir günlüğe yazarak ve bilgilerin sabit diske yazıldığından emin olarak bunu yapar. 1. Aşama, hareket yöneticisi her kaynak yöneticisinden, kaynaklarına ilişkin olarak kaynaklarına ilişkin bilgilerin kurtarılabilir bir durumda olduğunu bildirdiğinde tamamlanır.

Aşama 2 (Kesinleştir)
1. Aşama tamamlandığında, hareket yöneticisi, iş birimini kesinleştirmek için geri alınamaz bir karar verir. Her kaynak yöneticisinde, çalışma birimine katılan her kaynak yöneticisinde, bu güncellemelerin kaynaklarına ilişkin güncellemeleri kesinleştirmelerini ister. Bir kaynak yöneticisi bu isteği aldığında, güncellemeleri kesinleştirmelidir. Onları bu aşamada geri getirmenin bir yolu yok. 2. Aşama, hareket yöneticisinin kaynaklarındaki güncellemeleri kesinleştirdiği her kaynak yöneticisinden bildirim aldığında tamamlanır.

XA Arabirimi, iki aşamalı kesinleştirme protokolünü kullanır.

Daha fazla bilgi için bakınız: Transactional support scenarios.

IBM MQ , Microsoft Transaction Server (COM +) için de destek sağlar. Microsoft Transaction Server olanağının kullanılması (COM +) , COM + desteğinden yararlanmak için IBM MQ ' in nasıl ayarlanacak hakkında bilgi sağlar.

Kuyruk yöneticisi olanaklarının genişletmesi

Kuyruk yöneticisi olanaklarını kullanıcı çıkışlarını, API çıkışlarını ya da kurulabilir hizmetleri kullanarak genişletebilirsiniz.

Kullanıcı çıkışları

Kullanıcı çıkışları, kendi kodunuzu kuyruk yöneticisi işlevine yerleştirmeniz için bir mekanizma sağlar. Desteklenen kullanıcı çıkışları şunlardır:

Kanal çıkışları
Bu çıkışlar, kanalların işleyiş şeklini değiştirir. Kanal çıkışları İleti alışverişi kanallarına ilişkin kanal çıkışı programlarıiçinde anlatılır.

Veri dönüştürme çıkışları
Bu çıkışlar, verileri bir biçimden diğerine dönüştürmek için uygulama programlarına yerleştirilebilir kaynak kod parçaları oluşturur. Veri dönüştürme çıkışları, Yazma verileri-dönüştürme çıkışlarıiçinde açıklanır.

Küme iş yükü çıkışı
Bu çıkış tarafından gerçekleştirilen işlev, çıkışa ilişkin sağlayıcıya göre tanımlanır. Arama tanımı bilgileri MQ_CLUSTER_WORKLOAD_EXIT-Call descriptioniçinde verilir.

API çıkışları

API çıkışları, IBM MQ API çağrılarının davranışını değiştiren (MQPUT ve MQGET gibi) kod yazmanızı sağlar ve bu çağrıların hemen ardından ya da hemen sonra bu kodu eklemenize olanak sağlar. Ekleme otomatidir; kuyruk yöneticisi çıkış kodunu kayıtlı noktalarda yönlendirir. API çıkışlarıyla ilgili daha fazla bilgi için API çıkışlarının kullanılması ve yazılmasıbaşlıklı konuya bakın.

Kurulabilir hizmetler

Kurulabilir hizmetler, birden çok giriş noktası içeren biçimlendirilmiş arabirimlere (API) sahiptir.

Kurulabilir bir hizmete ilişkin bir somutlama *hizmet bileşeni*olarak adlandırılır. IBM MQile verilen bileşenleri kullanabilir ya da gereksinim duyduğunuz işlevleri gerçekleştirmek için kendi bileşeninizi yazabilirsiniz.

Şu anda aşağıdaki kurulabilir hizmetler verilmiştir:

Yetkilendirme hizmeti
Yetki hizmeti, kendi güvenlik tesisinizi oluşturmanıza olanak sağlar.

Hizmeti gerçekleştiren varsayılan hizmet bileşeni, nesne yetkilisi yöneticidir (OAM). Varsayılan değer olarak, OAM etkindir ve bunu yapılandırmak için herhangi bir şey yapmanız gerekmez. OAM ' yi değiştirmek ya da genişletmek üzere başka bileşenler yaratmak için yetkilendirme hizmeti arabirimini kullanabilirsiniz. OAM hakkında daha fazla bilgi için bkz. AIX, Linux, and Windows systems üzerinde güvenliğin ayarlanması.

Ad hizmeti
Ad hizmeti, uzak kuyrukları yerel kuyruklar gibi tanımlayarak, uygulamaların kuyrukları paylaşmasını sağlar.

Kendi ad hizmeti bileşeninizi yazabilirsiniz. Örneğin, ad hizmetini IBM MQile birlikte kullanmayı amaçlıyorsanız, bunu yapmak isteyebilirsiniz. Ad hizmetini kullanmak için, kullanıcı tarafından yazılan ya da farklı bir yazılım satıcısı tarafından sağlanan bir bileşeniniz olmalıdır. Varsayılan olarak, ad hizmeti etkin değildir.

İlgili kavramlar
Kullanıcı çıkışları, API çıkışları ve IBM MQ kurulabilir hizmetleri

IBM MQ Java dil arabirimleri

IBM MQ , Java uygulamalarında kullanım için iki alternatif uygulama programlama arabirimi (API) sağlar: IBM MQ classes for Java Message Service ve IBM MQ classes for Java.

IBM , açık standartların etkin bir katılımcısı, açık standartları ve API standardındaki ileti alanı içinde Java Message Service (JMS) olduğunu destekler. Ürün, IBM MQ 8.0'den, paylaşılan abonelikler gibi özelliklerle birlikte yeni bir basitleştirilmiş API' yi tanıtan JMS 2.0 standardını uygular. Buna ek olarak, WebSphere Liberty , hem varsayılan ileti sağlayıcısı hem de IBM MQile JMS 2.0 için destek içerir.

IBM MQ içinde, Java uygulamalarında kullanılmak üzere iki alternatif API vardır:

IBM MQ classes for JMS
IBM MQ classes for Java Message Service (JMS), IBM MQile birlikte verilen JMS sağlayıcısıdır. Java Platform, Enterprise Edition Connector Architecture (JCA), Java EE ortamında çalışan uygulamaların IBM MQ ya da Db2gibi bir Enterprise Information System (EIS) ortamına bağlanması için standart bir yol sağlar.

IBM MQ classes for Java
IBM MQ classes for Java enable you to use IBM MQ in a Java environment. IBM MQ classes for Java allow a Java application to connect to IBM MQ as an IBM MQ client, or connect directly to an IBM MQ queue manager.

Not: IBM MQ classes for Java , IBM MQ 8.0' ta teslim edilen düzeyde işlevsel olarak sabitlenmektedir. IBM MQ classes for Java ' u kullanan var olan uygulamalar tam olarak desteklenmeye devam eder, ancak bu API dengelenir, bu nedenle yeni özellikler eklenmez ve geliştirmeler reddedilmeye yönelik istekler için istekte bulunmaz. Tam olarak desteklenen bir şekilde, hataların IBM MQ Sistem Gereksinmeleri üzerindeki değişikliklerle birlikte belirlendiği değişiklikler ile birlikte düzeltileceği anlamına gelir.

IBM MQ 8.0, IBM MQ classes for Java ve IBM MQ classes for JMS , Java 7ile oluşturulmuştur. Java 7 yürütme ortamı, daha önceki sınıf dosyası sürümlerinin çalıştırılmasını destekler.

İlgili kavramlar
Neden IBM MQ classes for JMSkullanmalıyım?
Neden IBM MQ classes for Javakullanmalıyım?
JMS modeli
İlgili görevler
JMS 2.0 işlevselliğini kullanma

IBM MQ classes for JMS

IBM MQ classes for JMS , IBM MQile birlikte verilen JMS sağlayıcısıdır. IBM MQ classes for JMS , javax.jms paketinde tanımlanan arabirimleri uygular ve ayrıca JMS API ' ye iki uzantı kümesi sağlar. Hem Java Platform, Standard Edition (Java SE), hem de Java Platform, Enterprise Edition (Java EE) uygulamaları IBM MQ classes for JMS' i kullanabilir.

JMS belirtimi, uygulamaların ileti alışverişi işlemlerini gerçekleştirmek için kullanabilecekleri bir arabirim kümesini tanımlar. From IBM MQ 8.0, the product supports the JMS 2.0 version of the JMS standard. Bu uygulama, klasik API ' nin tüm özelliklerini sunar, ancak daha az arabirim gerektirir ve kullanımı daha kolay olur. Daha fazla bilgi için, bkz. “JMS model” sayfa 147 ve Java.net' daki JMS 2.0 belirtimine bakın.

javax.jms paketi, JMS arabirimlerinin ayrıntılarını belirtir ve bir JMS sağlayıcısı, belirli bir ileti alışverişi ürünü için bu arabirimleri uygular. IBM MQ classes for JMS , IBM MQ için JMS arabirimlerini uygulayan bir JMS sağlayıcısıdır ve ayrıca JMS API ' ya aşağıdaki iki uzantı kümesini de sağlar:

- IBM MQ JMS uzantıları
- IBM JMS uzantıları

javax.jms arabirimi ya da JMS uzantıları kümesi kullanılarak oluşturulan bir bağlantı üreticisi, kuyruğu ya da konu nesnesi bu APIs'lerden herhangi biri kullanılarak adreslenilebilir; bu, arabirimlerin herhangi birine dönüştürülebilmektedir. Uygulama taşınabilirliğini en üst düzeyde tutmak için gereksinimlerinize uygun en genel API ' yı kullanın.

IBM MQ JMS uzantıları

IBM MQ classes for JMS , JMS API ' ye uzantılar da sağlar. IBM MQ classes for JMS ' ın önceki yayın düzeyleri, MQConnectionFactory, MQQueue ve MQTopic nesnelerinde uygulanan uzantıları içerir. Bu nesneler, IBM MQ' e özgü özellikleri ve yöntemleri içerir. Nesneler yönetilebilir ya da bir uygulama, çalıştırma zamanında nesneleri dinamik olarak yaratabilir. IBM MQ classes for JMS , bu uzantıları korur ve bu uzantıları kullanan uygulamalar, değişiklik olmadan, kullanmaya devam edebilirsiniz. Bu uzantılar, IBM MQ JMS uzantıları olarak bilinir. Bu belge kümesinde, yürütme sırasında bir uygulama tarafından devingen olarak yaratılan nesnelerin, denetlenmekte olan nesneler olarak değerlendirilmediğini unutmayın.

IBM JMS uzantıları

In addition to the IBM MQ JMS extensions, IBM MQ classes for JMS provides a more generic set of extensions to the JMS API. Bu uzantılar, IBM JMS uzantıları olarak bilinir ve aşağıdaki geniş hedeflere sahiptir:

- IBM JMS sağlayıcıları arasında daha yüksek bir tutarlılık düzeyi sağlamak için
- İki IBM ileti sistemi arasında bir köprü uygulaması yazmanın daha kolay olmasını sağlamak
- Bir uygulamayı bir IBM JMS sağlayıcısından başka biranothersağlayıcıya bir uygulamayı daha kolay bir şekilde kaplamak için

Bu uzantıların ana odağı, yürütme sırasında bağlantı üreticilerinin ve hedeflerin dinamik olarak oluşturulması ve yapılandırılması ile ilgilidir; ancak, uzantılar sorunun saptanması için işlev gibi ileti sistemiyle doğrudan ilgili olmayan bir işlev de sağlar.

JMS model

The JMS model defines a set of interfaces that Java applications can use to perform messaging operations. IBM MQ classes for JMS, JMS sağlayıcısı olarak, JMS nesnelerinin IBM MQ kavramlarına nasıl ilişkin olduğunu tanımlar. JMS belirtimi, belirli JMS nesnelerinin yönetilecek nesneler olmasını bekler.

The JMS specification and the javax.jms package define a set of interfaces that Java applications can use to perform messaging operations.

From IBM MQ 8.0, the product supports the JMS 2.0 version of the JMS standard, which introduces a simplified API, while also retaining the classic API, from JMS 1.1.

Basitleştirilmiş API

JMS 2.0 introduces the simplified API, while also retaining the domain specific and domain independent interfaces from JMS 1.1. Basitleştirilmiş API, ileti göndermek ve almak için gerekli olan nesnelerin sayısını azaltır ve aşağıdaki arabirimlerden oluşur:

ConnectionFactory
ConnectionFactory , bir JMS istemcisi tarafından bir Bağlantı oluşturmak için kullanılan, yönetilen bir nesnedir. Bu arabirim, klasik API ' da da kullanılır.

JMSBağlam
Bu nesne, klasik API ' nin Connection ve Session nesnelerini birleştirir. JMSBağlam nesneleri, diğer JMSBağlam nesnelerinden yaratılabilir ve temeldeki bağlantı yinelenir.

JMSÜretici
Bir JMSüreticisi bir JMSBağlamı tarafından yaratılır ve bir kuyruğa ya da konuya ileti göndermek için kullanılır. JMSüretici nesnesi, iletiyi göndermek için gereken nesnelerin yaratılmasına neden olur.

JMSTüketici
JMSTüketici bir JMSBağlamı tarafından oluşturulur ve bir konu ya da kuyruktan ileti almak için kullanılır.

Basitleştirilmiş API ' nin bir dizi etkisi vardır:

- JMSBağlam nesnesi her zaman temel bağlantıyı otomatik olarak başlatır.
- JMSProducers and JMSConsumers can now work directly with message bodies, without having to get the whole message object, by using the Message's getBody method.
- Bir 'body' göndermeden önce, bir 'body' göndermeden önce, JMSüretici nesnesi üzerinde ileti özellikleri ayarlanabilir. JMSüreticisi, iletiyi göndermek için gerekli olan tüm nesnelerin yaratılmasını işleyecek. JMS 2.0komutunu kullanarak, özellikler ayarlanabilir ve aşağıdaki gibi gönderilen bir ileti kullanılabilir:

```
context.createProducer().
setProperty("foo", "bar").
setTimeToLive(10000).
setDeliveryMode(NON_PERSISTENT).
setDisableMessageTimestamp(true).
send(dataQueue, body);
```

JMS 2.0 , iletilerin birden çok tüketici arasında paylaşılabildiği paylaşımlı abonelikler de sunar. Tüm JMS 1.1 abonelikleri paylaşılmayan abonelikler olarak ele alınır.

Klasik API

Aşağıdaki liste, klasik API ' nin ana JMS arabirimlerini özetlemektedir:

Hedef
Bir hedef, bir uygulamanın iletileri gönderdiği ya da bir uygulamanın iletileri aldığı bir kaynaktır ya da her ikisi de olabilir.

ConnectionFactory
Bir ConnectionFactory nesnesi, bir bağlantı için bir yapılandırma özellikleri kümesini sarsalıyor. Bir uygulama, bağlantı oluşturmak için bir bağlantı üreticisi kullanır.

Bağlantı
Bir bağlantı nesnesi, bir uygulamanın ileti sistemi ile etkin bağlantısını sarsalıyor. Uygulama, oturum yaratmak için bir bağlantı kullanır.

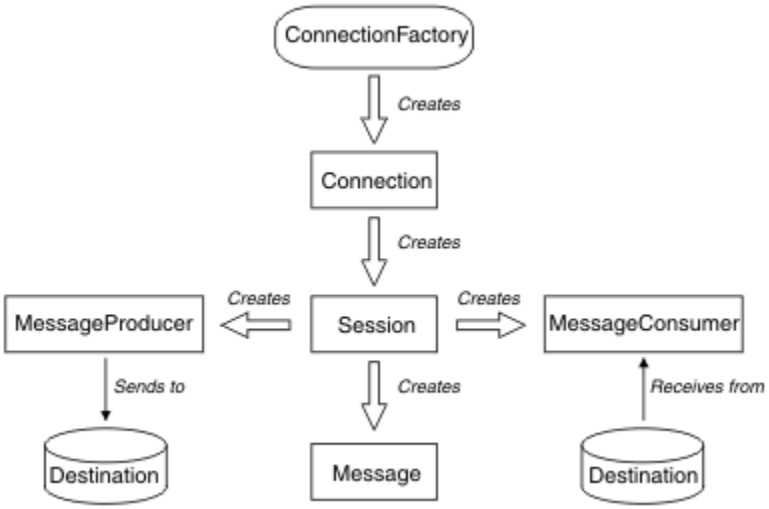
Oturum
Oturum, ileti göndermek ve almak için tek bir iş parçacıklı bağlamdır. Bir uygulama, ileti, ileti üreticileri ve ileti tüketicileri oluşturmak için bir oturumu kullanır. Bir oturum hareket edilir ya da hareket edilmez.

İleti
Bir ileti nesnesi, bir uygulamanın gönderdiği ya da gönderdiği bir iletiyi sarsalıyor.

MessageProducer
Bir uygulama, iletileri bir hedefe göndermek için bir ileti üreticiyi kullanır.

MessageConsumer
Bir uygulama, bir hedefe gönderilen iletileri almak için bir ileti tüketicisi kullanır.

Şekil 54 sayfa 149 , bu nesneleri ve bunların ilişkilerini gösterir.



Şekil 54. JMS nesneleri ve ilişkileri

Çizgede ana arabirimler şunlardır: ConnectionFactory, Connection, Session, MessageProducer, MessageConsumer, Message ve Destination. Uygulama, bağlantı yaratmak için bir bağlantı üreticisi kullanır ve oturum yaratmak için bir bağlantı kullanır. Daha sonra, uygulama, ileti, ileti üreticileri ve ileti tüketicileri oluşturmak için bir oturumu kullanabilir. Uygulama, bir hedefe ileti göndermek için bir ileti üreticisi kullanır ve bir hedefe gönderilen iletileri almak için bir ileti tüketicisi kullanır.

Bir Hedef, ConnectionFactoryya da Connection nesnesi, çok iş parçacıklı bir uygulamanın farklı iş parçacıkları tarafından eşzamanlı olarak kullanılabilir, ancak bir Oturum, MessageProducerya da MessageConsumer nesnesi farklı iş parçacıkları tarafından koşutzamanlı olarak kullanılamaz. Bir Oturum, MessageProducerya da MessageConsumer nesnesinin koşutzamanlı olarak kullanılmamasını sağlamanın en basit yolu, her iş parçacığı için ayrı bir Oturum nesnesi yaratmamaktadır.

JMS , iki ileti sistemini destekler:

- Noktadan Noktaya İleti Sistemi
- Yayınlama/abone olma ileti alışverişi

Bu ileti alışverişi biçemleri de *ileti sistemi etki alanları*olarak da adlandırılır ve her iki ileti sistemi stilini de bir uygulamada birleştirebilirsiniz. Noktadan noktaya iletişim alanında, hedef bir kuyruktur ve yayınlama/abone olma etki alanında, hedef bir konudur.

JMS 1.1tarihinden önce JMS sürümleriyle, noktadan noktaya iletişim etki alanı için programlama bir arabirim ve yöntem kümesini kullanır ve yayınlama/abone olma etki alanı için programlama başka bir küme kullanır. İki set birbirine benzer, ama ayrı. JMS 1.1' tan, hem ileti sistemi etki alanlarını destekleyen, ortak bir arabirim ve yöntem kümesi kullanabilirsiniz. Ortak arabirimler, her ileti sistemi etki alanı için bir etki alanı bağımsız görünümü sağlar. Çizelge 16 sayfa 149 , JMS etki alanı bağımsız arabirimlerini ve bunlara karşılık gelen etki alanlarını özel arabirimlerini listeler.

Çizelge 16. JMS etki alanı bağımsız ve bunlara karşılık gelen etki alanına özgü arabirimler

Etki alanı bağımsız arabirimleri	**Noktadan noktaya etki alanına ilişkin etki alanına özgü arabirimler**	**Yayınlama/abone olma etki alanı için etki alanına özgü arabirimler**
ConnectionFactory	QueueConnectionÜreticisi	TopicConnectionÜreticisi

Çizelge 16. JMS etki alanı bağımsız ve bunlara karşılık gelen etki alanına özgü arabirimler (devamı var)		
Etki alanı bağımsız arabirimleri	**Noktadan noktaya etki alanına ilişkin etki alanına özgü arabirimler**	**Yayınlama/abone olma etki alanı için etki alanına özgü arabirimler**
Bağlantı	QueueConnection	TopicConnection
Hedef	Kuyruk	Konu
Oturum	QueueSession	TopicSession
MessageProducer	QueueSender	TopicPublisher
MessageConsumer	QueueReceiver QueueBrowser	TopicSubscriber

JMS 2.0 , etki alanına özgü tüm arabirimleri korur ve bu nedenle var olan uygulamalar bu arabirimleri kullanmaya devam edebilir. For new applications, however, consider using the domain independent interfaces of JMS 1.1 or the simplified API of JMS 2.0.

IBM MQ classes for JMSiçinde, JMS nesneleri IBM MQ kavramlarına aşağıdaki şekillerde ilgilidir:

- Bağlantı nesnesinin, bağlantıyı yaratmak için kullanılan bağlantı üreticisinin özelliklerinden türetilmiş özellikleri vardır. Bu özellikler, bir uygulamanın kuyruk yöneticisine nasıl bağlanacağını denetler. Bu özelliklere örnek olarak, kuyruk yöneticisinin adı ve istemci kipinde kuyruk yöneticisine bağlanan bir uygulama için, kuyruk yöneticisinin çalıştığı sistemin anasistem adı ya da IP adresi.
- Bir oturum nesnesi, oturumun işlem kapsamını tanımlayan bir IBM MQ bağlantı tanıtıcısını sarmadır.
- Bir MessageProducer nesnesi ve her biri bir IBM MQ nesne tanıtıcısını sarmalayan bir MessageConsumer nesnesinden.

When using IBM MQ classes for JMS, all the normal rules of IBM MQ apply. Özellikle, bir uygulamanın uzak bir kuyruğa ileti gönderebileceği, ancak yalnızca, uygulamanın bağlı olduğu kuyruk yöneticisinin sahip olduğu bir kuyruktan bir ileti alabileceği unutulmamalı.

JMS belirtimi, ConnectionFactory ve Hedef nesnelerin denetlenmesini bekler. Bir denetimci, denetlenen nesneleri merkezi bir havuzda yaratır ve bakımını yapar; JMS uygulaması bu nesneleri Java Naming and Directory Interface (JNDI) kullanarak alır.

IBM MQ classes for JMS' ta, Hedef arabirimin somutlaması, Kuyruk ve Konu 'nın soyut bir üst sınıfıdır; dolayısıyla, Hedef eşgörünümü bir Kuyruk nesnesi ya da Konu nesnesidir. Etki alanı bağımsız arabirimleri, bir kuyruğu ya da konuyu hedef olarak kabul eder. Bir MessageProducer ya da MessageConsumer nesnesine ilişkin ileti alışverişi etki alanı, hedefin bir kuyruk mu, yoksa bir konu mı tarafından belirlenir.

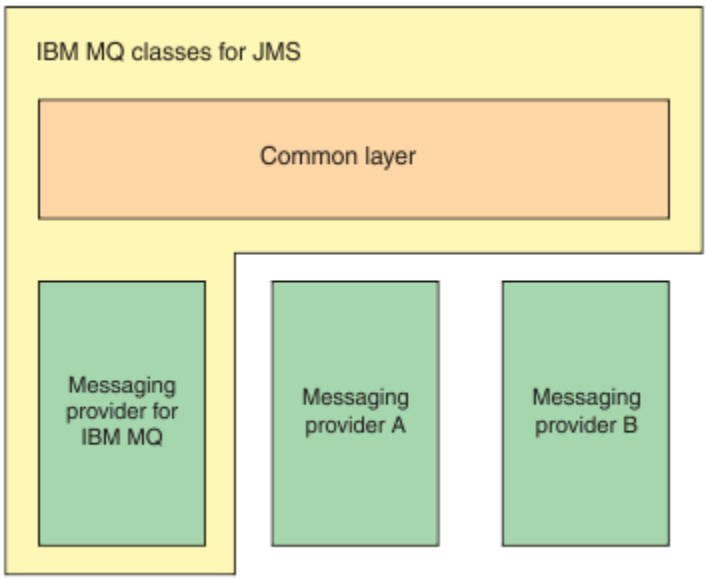
Bu nedenle, IBM MQ classes for JMS içinde aşağıdaki tiplerin nesneleri denetleyebilirler:

- ConnectionFactory
- QueueConnectionÜreticisi
- TopicConnectionÜreticisi
- Kuyruk
- Konu
- XAConnectionFactory
- XAQueueConnectionÜreticisi
- XATopicConnectionÜreticisi

IBM MQ classes for JMS Mimari

IBM MQ classes for JMS , katmanlı bir mimariye sahiptir. Kodun en üst katmanı, herhangi bir IBM JMS sağlayıcısının kullanabileceği ortak bir katmanıdır.

IBM MQ classes for JMS has a layered architecture as shown in the diagram Şekil 55 sayfa 151. Kodun en üst katmanı, herhangi bir IBM JMS sağlayıcısı tarafından kullanılabilen ortak bir katmanıdır. Bir uygulama bir JMS yöntemini çağırdığında, bir ileti sistemi sistemine özgü olmayan bir çağrı işlemi, çağrıya tutarlı bir yanıt da sağlayan ortak katman tarafından gerçekleştirilir. Bir ileti sistemi sistemine özgü çağrıların işlenmesi daha düşük bir katmana devredilir. In the following diagram, the IBM MQ messaging provider is shown in the lower layer, together with two further messaging providers (Messaging provider A and Messaging provider B.)



Şekil 55. The layered architecture for IBM JMS providers

Katmanlı bir mimari, aşağıdaki hedefleri yerine getirir:

- Çeşitli IBM JMS sağlayıcılarının davranışlarının tutarlılığını artırmak için
- İki IBM ileti sistemi arasında bir köprü uygulaması yazmanın daha kolay olmasını sağlamak
- Bir uygulamayı bir IBM JMS sağlayıcısından başka biranothersağlayıcıya bir uygulamayı daha kolay bir şekilde kaplamak için

Denetlenen nesneler için destek

IBM MQ classes for JMS denetimli nesne kullanımını destekler.

Bir JMS uygulaması içindeki mantık akışı, ConnectionFactory ve Destination nesneleriyle başlar. Uygulama, uygulamadan bir ileti alışverişi sunucusuna etkin bağlantıyı gösteren bir Connection nesnesi yaratmak için bir ConnectionFactory nesnesi kullanır. Uygulama, iletiler üretmek ve tüketmek için tek bir iş parçacıklı bağlam olan bir Oturum nesnesi yaratmak için Connection nesnesini kullanır. Uygulama daha sonra, uygulamanın belirtilen hedefe ileti göndermek için kullandığı bir MessageProducer nesnesi yaratmak için Oturum nesnesini ve hedef nesneyi kullanabilir. Hedef, ileti sistemi sistemindeki bir kuyrukta ya da bir konudur ve Hedef nesne tarafından kapsüllenmiş olur. Uygulama ayrıca, uygulamanın belirtilen hedefe gönderilen iletileri almak için kullandığı bir MessageConsumer nesnesi yaratmak için Oturum nesnesini ve hedef nesneyi kullanabilir.

JMS belirtimi, ConnectionFactory ve Hedef nesnelerin denetlenmesini bekler. Bir yönetici, merkezi bir havuzda denetlenen nesneleri yaratır ve bakımını yapar ve JMS uygulaması bu nesneleri Java Naming Directory Interface (JNDI) kullanarak alır. Denetlenen nesnelerin havuzu, basit bir dosyadan bir LDAP (Lightweight Directory Access Protocol; Temel Dizin Erişimi Protokolü) dizinine kadar aralığa neden olabilir.

IBM MQ classes for JMS denetimli nesne kullanımını destekler. An application can use all the features of IBM MQ classes for JMS that are exposed through IBM MQ without having any IBM MQ -specific information hard coded into the application itself. Bu düzenleme, temeldeki IBM MQ yapılandırmasından

bir derece bağımsızlığa sahip bir uygulama sağlar. Bu bağımsızlığı elde etmek için uygulama, yönetilen nesneler olarak depolanan bağlantı fabrikalarını ve hedeflerini almak için JNDI kullanabilir ve ileti alışverişi işlemlerini gerçekleştirmek için yalnızca javax.jms paketinde tanımlanan arabirimleri kullanır. An administrator can use the IBM MQ JMS administration tool or IBM MQ Explorer to create and maintain administered objects in a central repository. Ancak, bir uygulama sunucusu, nesneleri oluşturmak ve korumak için yönetilen nesneler ve kendi araçları için kendi havuzunu sağlar. Bu nedenle bir Java EE uygulaması, yönetilen nesneleri uygulama sunucusu havuzundan ya da merkezi bir havuzdan almak için JNDI ' i kullanabilir.

İlgili bilgiler

JMS kaynaklarının yapılandırılması

Java EE altyapılarında desteklenen iletişim tipleri

Java EE platformunda, IBM MQ classes for JMS , bir uygulama bileşeni ile IBM MQ kuyruk yöneticisi arasında iki iletişim tipini destekler.

Bir uygulama bileşeni ile IBM MQ kuyruk yöneticisi arasındaki iki iletişim tipi desteklenir:

- Giden iletişim
- Gelen iletişim

Giden iletişim

Using the JMS API directly, an application component creates a connection to a queue manager, and then sends and receives messages.

Örneğin, uygulama bileşeni bir uygulama istemcisi, bir sunucu uygulaması, Java Server Page (JSP), kurumsal Java Bean (EJB) ya da ileti odaklı bir Bean (MDB) olabilir. Bu iletişim tipinde, uygulama sunucusu taşıyıcısı, bağlantı havuzlama ve iş parçacığı yönetimi gibi ileti alışverişi işlemlerini desteklemek için yalnızca düşük düzeyli işlevler sağlar.

Gelen iletişim

Gelen iletişim durumunda, bir hedefe varılan bir ileti bir MDB ' ye teslim edilir ve sonra iletiyi işler.

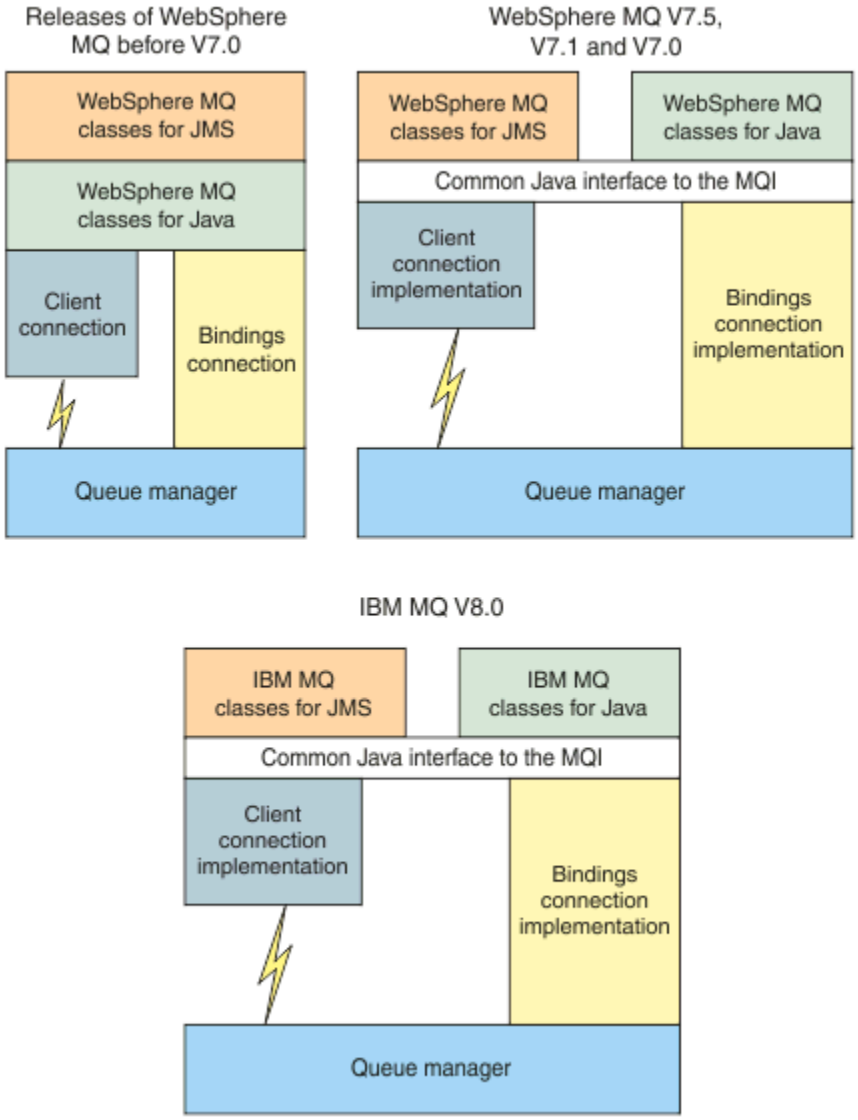
Java EE uygulamaları, iletileri zamanuyumsuz olarak işlemek için MDBs ' yi kullanır. An MDB acts as a JMS message listener and is implemented by an onMessage() method, which defines how a message is processed. Bir MDB, bir uygulama sunucusunun EJB taşıyıcısında konuşlandırılır. Bir MDB 'nin yapılandırıldığı kesin yol, kullandığınız uygulama sunucusuna bağlıdır; ancak, yapılanış bilgilerinin hangi kuyruk yöneticisinin bağlanacağını, kuyruk yöneticisine nasıl bağlanacağını, hangi hedefe ilişkin iletileri izleyeceğini ve MDB' nin hareket davranışını belirtmesi gerekir. Bu bilgi daha sonra EJB taşıyıcısı tarafından kullanılır. Belirtilen hedefte MDB 'nin seçim ölçütlerini karşılayan bir ileti geldiğinde, EJB taşıyıcısı iletiyi kuyruk yöneticisinden almak için IBM MQ classes for JMS öğesini kullanır ve daha sonra, onMessage() yöntemini çağırarak iletiyi MDB' ye aktarır.

IBM MQ classes for Javaile ilişki

IBM WebSphere MQ 7.0, IBM WebSphere MQ classes for JMS ve IBM WebSphere MQ classes for Java , eş olarak uygulanmaktadır. This implementation is different from earlier releases where the implementation of IBM WebSphere MQ classes for JMS depended on IBM WebSphere MQ classes for Java.

IBM WebSphere MQ 7.0'dan önceki sürümler için, IBM WebSphere MQ classes for JMS neredeyse tamamen IBM WebSphere MQ classes for Java' un üst kısmında bir kod katmanı olarak uygulanmış. Bu düzenleme, uygulama geliştiricileri arasında bazı karışıklığa neden olmuştur; çünkü, MQEnvironment sınıfındaki alanlar ya da çağrı yöntemleri, IBM MQ classes for JMSkullanılarak yazılmış kodun yürütme zamanı davranışlarında istenmeyen ve beklenmeyen etkilere neden olabilir. Buna ek olarak, IBM MQ classes for JMS uygulaması, JMS API 'nın IBM MQ classes for Java' ın üzerine doğal bir uyum göstermediği alanlarda bazı kısıtlamalar getirmişti ve bu kısıtlamalar, çalıştırma zamanı performansıyla ilgili bazı sorunlara yol açmıştı.

From IBM WebSphere MQ 7.0, the implementation of IBM MQ classes for JMS is no longer dependent on IBM MQ classes for Java. IBM MQ classes for Java ve IBM MQ classes for JMS , MQI ' ye ortak bir Java arabirimi kullanan eşdüzeylerdir. Bu düzenleme, performansın en iyi duruma getirilmesi için daha fazla kapsama olanak sağlar ve MQEnvironment sınıfındaki alanların ya da çağrı yöntemlerinin, IBM MQ classes for JMSkullanılarak yazılan kod yürütme davranışı üzerinde hiçbir etkisi olmadığını gösterir. Şekil 56 sayfa 153 shows the relationship between IBM MQ classes for JMS and IBM MQ classes for Java in previous releases of IBM WebSphere MQ classes for JMS and in releases before IBM WebSphere MQ 7.0 and how this relationship has changed for later releases.



Şekil 56. IBM MQ classes for JMS ile IBM MQ classes for Javaarasındaki ilişki

To maintain compatibility with releases before IBM WebSphere MQ 7.0, channel exit classes that are written in Java can still use the IBM MQ classes for Java interfaces, even if the channel exit classes are called from IBM MQ classes for JMS. Ancak, IBM MQ classes for Java arabirimlerinin kullanılması, uygulamalarınızın hala IBM MQ classes for Java JAR dosyasına (`com.ibm.mq.jar`) bağımlı olduğu anlamına gelir. Sınıf yolunuzda `com.ibm.mq.jar` istemiyorsanız, bunun yerine `com.ibm.mq.exits` paketindeki arabirimler kümesini kullanabilirsiniz.

From IBM WebSphere MQ 7.0, you can create and configure JMS administered objects with the IBM MQ Explorer.

IBM MQ ileti alışverişi sağlayıcısı

IBM MQ ileti alışverişi sağlayıcısında üç işlem kipi vardır: Olağan kip, kısıtlamalar içeren normal kip ve geçiş kipi.

IBM MQ ileti alışverişi sağlayıcısında üç işlem kipi vardır:

- IBM MQ ileti alışverişi sağlayıcısı olağan kipi
- Kısıtlamaları olanIBM MQ ileti alışverişi sağlayıcısı normal kipi
- IBM MQ ileti alışverişi sağlayıcısı geçiş kipi

The IBM MQ messaging provider normal mode uses all the features of an IBM MQ queue manager to implement JMS. Bu kip, JMS 2.0 API ve işlevlerini kullanmak için eniyilenir.

The IBM MQ messaging provider normal mode with restrictions uses the JMS 2.0 API, but not the new IBM MQ 8.0 features such as shared subscriptions, delayed delivery, or asynchronous send.

IBM MQ ileti alışverişi sağlayıcısı geçiş kipi IBM WebSphere MQ 6.0 işlevini temel alır ve yalnızca IBM WebSphere MQ 6.0 kuyruk yöneticisinde bulunan ve JMS' yi uygulamak için kullanılan özellikleri kullanır. You can connect to IBM WebSphere MQ 7.0, or later, queue managers that use the migration mode but you cannot use any of the IBM WebSphere MQ 7.0 optimizations. Bu kip, aşağıdaki kuyruk yöneticisi sürümlerinden herhangi birine bağlantı sağlar:

1. IBM WebSphere MQ 7.0 ya da sonraki bir sürümü, bağ tanımlarında ya da istemci kipinde kuyruk yöneticisi, ancak bu kip yalnızca IBM WebSphere MQ 6.0 kuyruk yöneticisi tarafından kullanılabilecek özellikleri kullanır.
2. İstemci kipindeIBM WebSphere MQ 6.0 ya da daha önceki bir kuyruk yöneticisi.

If you want to connect to IBM Integration Bus by using IBM MQ Enterprise Transport, use the migration mode. IBM MQ Real-Time Transport özelliğini kullanırsanız, bağlantı üreticisi nesnesinde belirtik olarak seçtiğiniz özellikler olduğundan geçiş kipi otomatik olarak seçilir. IBM MQ Enterprise Transport kullanılarak IBM Integration Bus bağlantısı, JMS **PROVIDERVERSION** özelliğinin yapılandırılmasıbaşlıklı konuda açıklanan kip seçimine ilişkin genel kuralları izler.

İlgili görevler

JMS kaynaklarının yapılandırılması

z/OS IBM MQ for z/OS kavramlar

IBM MQ for z/OS ile kullanılan bazı kavramlar, z/OS platformunda benzersizdir. Örneğin, günlüğe kaydetme mekanizması, depolama yönetimi teknikleri, kurtarma düzeni birimi ve kuyruk paylaşım grupları yalnızca IBM MQ for z/OSile birlikte sağlanır. Bu kavramlarla ilgili daha fazla bilgi için bu konuyu kullanın.

İlgili kavramlar

“z/OSüzerindeki kuyruk yöneticisi” sayfa 155
Uygulama programlarının z/OS sisteminde IBM MQ kullanmasına izin vermeden önce, IBM MQ for z/OS ürününü kurmalı ve bir kuyruk yöneticisi başlatmalısınız. Kuyruk yöneticisi, IBM MQtarafından kullanılan kaynak kümesini sahiplenir ve yönetir.

“z/OSüzerindeki kanal başlatıcısı” sayfa 156
Kanal başlatıcı, IBM MQ dağıtılmış kuyruğa alma işlevini etkinleştiren kaynakları sağlar ve yönetir. IBM MQ , bir kuyruk yöneticisinden başka bir kuyruk yöneticisinden ileti göndermek için *Message Channel Agents* (MCA ' lar) olanağını kullanır.

“IBM MQ for z/OSyönetimine ilişkin terimler ve görevler” sayfa 158
Bu konuyu terminolojiye giriş olarak kullanın ve IBM MQ for z/OS' e özgü görevler.

“Paylaşılan kuyruklar ve kuyruk paylaşım grupları” sayfa 160
You can use shared queues and queue sharing groups, to implement high availability of IBM MQ resources. Paylaşılan kuyruklar ve kuyruk paylaşım grupları, z/OS altyapısındaki IBM MQ for z/OS için benzersiz işlevlerdir.

“Grup içi kuyruğa alma” sayfa 206

Bu bölümde grup içi kuyruğa alma, z/OS platformuna özgü bir IBM MQ for z/OS işlevi ele alınmıştır. Bu işlev yalnızca, bir kuyruk paylaşım grubuna tanımlı kuyruk yöneticilerine kullanılabilir.

“z/OSüzerinde depolama yönetimi” sayfa 218
IBM MQ for z/OS , kalıcı ve geçici veri yapıları gerektirir ve bu verileri saklamak için sayfa kümelerini ve bellek arabelleklerini kullanır. Bu konular, IBM MQ ' in bu sayfa kümelerini ve arabellekleri nasıl kullandığı hakkında daha fazla ayrıntı verir.

“oturum açmaIBM MQ for z/OS” sayfa 222
IBM MQ maintains *günlükler* of data changes and significant events as they occur. Bu günlükler, gerekiyorsa önceki bir duruma veri kurtarmak için kullanılabilir.

“Recovery and restart on z/OS” sayfa 244
Yeniden başlatma ve kurtarma işlemi için IBM MQ for z/OS özelliklerini öğrenmek üzere bu konudaki bağlantıları kullanın.

“IBM MQ for z/OSiçindeki güvenlik kavramları” sayfa 260
IBM MQgüvenliğinin önemini anlamak için bu konuyu ve sisteminizde yeterli güvenlik ayarına sahip olmamanın etkilerini anlamak için bu konuyu kullanın.

“z/OSüzerinde kullanılabilirlik” sayfa 266
IBM MQ for z/OS , yüksek düzeyde kullanılabilirlik için birçok özelliğe sahiptir. Bu konuda, kullanılabilirlik açısından dikkate alınması gereken bazı noktalar açıklanmaktadır.

“z/OSüzerinde kurtarma yok etme birimi” sayfa 271
Bazı işlemsel uygulamalar QMGR yerine, kuyruk yöneticisi adı yerine QSG adını belirterek bir kuyruk paylaşım grubundaki (QSG) bir kuyruk yöneticisine bağlandığında kurtarma düzeni birimi olarak bir GROUP (grup) kullanabilir. Bu, QSG ' de aynı kuyruk yöneticisine yeniden bağlanma gereksinimini kaldırarak hareket kurtarmanın daha esnek ve güçlü olmasını sağlar.

İlgili başvurular

“Defining your system on z/OS” sayfa 233
IBM MQ for z/OS , birçok varsayılan nesne tanımlaması kullanır ve bu varsayılan nesneleri yaratmak için örnek JCL sağlar. Bu varsayılan nesneleri ve örnek JCL ' yi anlamak için bu konuyu kullanın.

“IBM MQ for z/OSile ilgili izleme ve istatistikler” sayfa 270
IBM MQ for z/OS , kuyruk yöneticisini izlemek ve istatistikleri toplamak için bir dizi tesis içerir.

z/OS z/OSüzerindeki kuyruk yöneticisi

Uygulama programlarının z/OS sisteminde IBM MQ kullanmasına izin vermeden önce, IBM MQ for z/OS ürününü kurmalı ve bir kuyruk yöneticisi başlatmalısınız. Kuyruk yöneticisi, IBM MQtarafından kullanılan kaynak kümesini sahiplenir ve yönetir.

Kuyruk yöneticisi

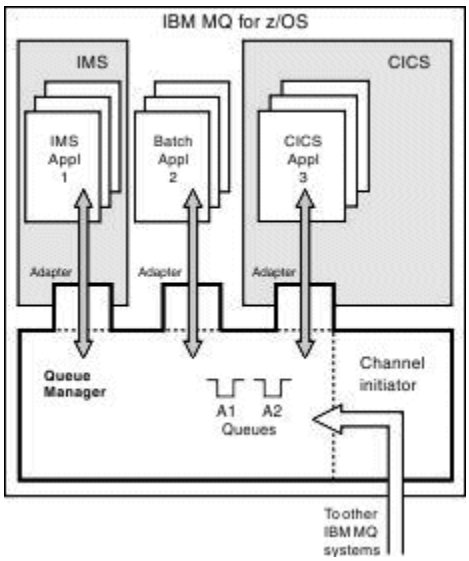
Kuyruk yöneticisi , uygulamalara ileti alışverişi hizmetleri sağlayan bir programdır. İleti Kuyruğu Arabirimi 'ni (MQI) kullanan uygulamalar kuyruklara ileti alabilir ve kuyruklardan ileti alabilir. Kuyruk yöneticisi iletilerin doğru kuyruğa gönderilmesini sağlar ya da başka bir kuyruk yöneticisine yöneltilir. Kuyruk yöneticisi, ona verilen MQI çağrılarını ve ona gönderilen komutları (hangi kaynaktan gelen) işler. Kuyruk yöneticisi, her çağrı ya da komut için uygun tamamlanma kodlarını oluşturur.

Kuyruk yöneticisi tarafından yönetilen kaynaklar şunlardır:

- IBM MQ nesne tanımlarını ve ileti verilerini tutan sayfa kümeleri
- Kuyruk yöneticisi hatası olayında iletileri ve nesneleri kurtarmak için kullanılan günlükler
- İşlemci depolama alanı
- Farklı uygulama ortamlarının (CICS, IMSve Toplu İş) IBM MQ API ' ye erişebileceği bağlantılar
- z/OS sistemi ve diğer sistemleriniz üzerinde IBM MQ arasında iletişim sağlayan IBM MQ kanalı başlatıcısı

Kuyruk yöneticisinin bir adı var ve uygulamalar bu adı kullanarak bu ada bağlanabiliyor.

Şekil 57 sayfa 156 , farklı uygulama ortamlarıyla ve kanal başlatıcısıyla bağlantıları gösteren bir kuyruk yöneticisini gösterir.



Şekil 57. IBM MQ for z/OS' a genel bakış

z/OSüzerindeki kuyruk yöneticisi altsistemi

z/OSüzerinde, IBM MQ IPL zamanında başlatılan bir z/OS altsistemi olarak çalışır. Altsistemde, kuyruk yöneticisi, günlüklerle ilgili bilgileri içeren ve nesne tanımlamalarını ve ileti verilerini (sayfa kümeleri) bulunduran z/OS veri kümelerini belirleyen bir JCL yordamı yürüterek başlatılır. Altsistem ve kuyruk yöneticisi, en çok dört karakterden oluşan aynı ada sahiptir. Ağınızdaki tüm kuyruk yöneticilerinin farklı sistemler, sysplexes ya da platformlar üzerinde olsalar da benzersiz adlara sahip olması gerekir.

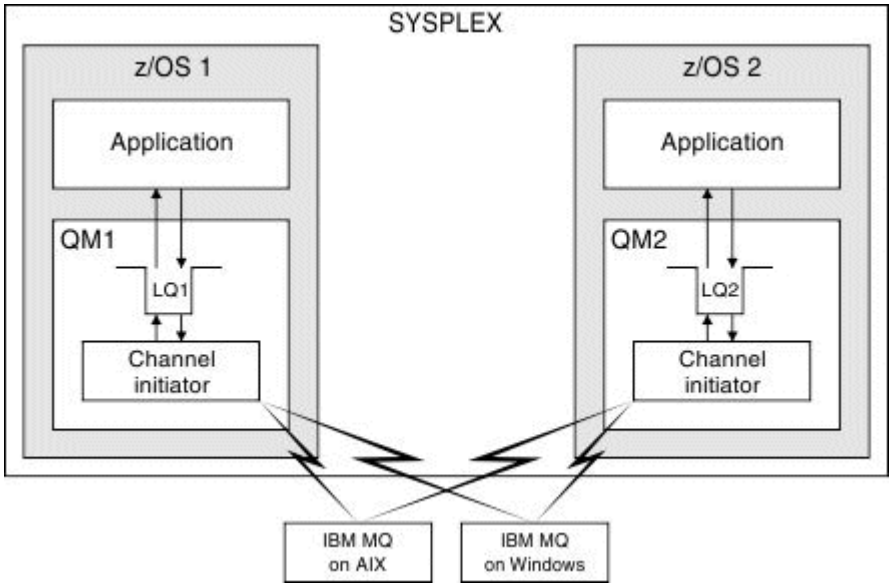
z/OS z/OSüzerindeki kanal başlatıcısı

Kanal başlatıcı, IBM MQ dağıtılmış kuyruğa alma işlevini etkinleştiren kaynakları sağlar ve yönetir. IBM MQ , bir kuyruk yöneticisinden başka bir kuyruk yöneticisinden ileti göndermek için *Message Channel Agents* (MCA ' lar) olanağını kullanır.

Kuyruk yöneticisi A 'dan kuyruk yöneticisi B' ye ileti göndermek için, kuyruk yöneticisi A 'daki *gönderme* MCA' nın kuyruk yöneticisi B ' ye bir iletişim bağlantısı ayarlaması gerekir. İletişim bağlantısından ileti almak için kuyruk yöneticisi B ' de bir *alıcı* MCA başlatılmalıdır. Gönderme MCA, iletişim bağlantısı ve alma MCA ' dan oluşan bu tek yönlü yol, *kanal*olarak bilinir. MCA 'nın gönderilmesi, iletileri iletim kuyruğundan alır ve bunları alma MCA' sına gönderir. Alma MCA, iletileri alır ve hedef kuyruklara yerleştirir.

IBM MQ for z/OS'ta, kanal başlatıcının içinde çalışan tüm MCA' ları gönderme ve alma (kanal başlatıcı, *mover* olarak da bilinir). Kanal başlatıcısı, kuyruk yöneticisinin denetimin altında bir z/OS adres alanı olarak çalışır. Kuyruk yöneticisine bağlı yalnızca tek bir kanal başlatıcısı olabilir ve kuyruk yöneticiyle aynı z/OS görüntüsünün içinde çalıştırılır. Kanal başlatıcısı içinde koşutzamanlı olarak çalışan binlerce MCA işlemi olabilir.

Şekil 58 sayfa 157 , bir sistem birleşimi (sysplex) içinde iki kuyruk yöneticisini gösterir. Her kuyruk yöneticisinin bir kanal başlatıcısı ve yerel kuyruğu vardır. AIX ve Windows ' da kuyruk yöneticileri tarafından gönderilen iletiler, bir uygulama tarafından alındıkları yerden yerel kuyruğa yerleştirilir. Yanıt iletileri benzer bir rota tarafından döndürülür.



Şekil 58. Kuyruk yöneticileri arasındaki iletişim

Kanal başlatıcı, kanalların yönetimiyle ilgili diğer süreçleri de içerir. Bu işlemler şunlardır:

Dinleyiciler
Bu işlemler, TCP gibi bir iletişim altsisteminde gelen kanal isteklerini dinler ve gelen istek alındığında bir MCA adını başlatır.

Denetmen
Bu, kanal başlatıcı adres alanını yönetir; örneğin, bir hatadan sonra kanalların yeniden başlatılmasından sorumlu olur.

Ad sunucusu
Bu, TCP adlarını adreslere çözmek için kullanılır.

TLS görevleri
Bunlar, şifreleme ve şifre çözme işlemlerini gerçekleştirmek ve sertifika iptal listelerini denetlemek için kullanılır.

z/OS Kanal başlatıcısı için SMF kayıtları

Kanal başlatıcı (CHINIT), SMF istatistik kayıtlarını ve muhasebe kayıtlarını görev ve kanallara ilişkin bilgilerle birlikte üretebilir.

CHINIT, SMF istatistik kayıtlarını ve muhasebe kayıtlarını aşağıdaki bilgi tipleriyle üretebilir:

- Görevler: dağıtıcı, bağdaştırıcı, Etki Alanı Ad Sunucusu (DNS) ve SSL. Bu görevler, CHINIT istatistikleri adı verilen görevleri oluşturur.
- Kanallar: DIS CHSTATUS komutu ile birlikte kullanılabilecek muhasebe bilgilerini sağlar. Buna kanal muhasebesi deniyor.

IBM MQ for Multiplatforms provides similar information by writing PCF messages to the SYSTEM.ADMIN.STATISTICS.QUEUE. See Kanal istatistikleri ileti verileri for further information on how statistics information is recorded on IBM MQ for Multiplatforms.

İstatistik Verileri

Aşağıdaki bilgileri öğrenmek için bu bilgileri kullanabilirsiniz:

- SSL TCB 'lerinin sayısı ve bu görevler tarafından kullanılan CPU' nun sayısı gibi daha fazla CHINIT görevi gerekip gerekmediği.

- Bu görevlerdeki istekler için ortalama süre.
- Aralık içindeki en uzun süre isteği ve bu, DNS ve SSL görevleri için ortaya çıkan günün saati. Kanal ile deneyimleyebileceğiniz sorunlarla ilgili günün bu saatini ilişkilendirebilirsiniz.

Hesap verileri

Kanal kullanımını izlemek ve aşağıdaki bilgileri öğrenmek için bu bilgileri kullanabilirsiniz:

- En yüksek verimi içeren kanallar.
- İletilerin gönderileceği ücret ve MB/saniye cinsinden veri gönderme hızı.
- Elde edilen toplu iş boyutu. Elde edilen toplu iş büyüklüğü, kanal için belirlenen toplu iş büyüklüğüne yakınsa, kanal, ileti gönderme sınırına yakın olabilir.

Hesap izleme ve istatistik izleme bilgilerini denetlemek için START TRACE ve STOP TRACE komutlarını kullanıyorsunuz. Kanalların SMF verileri üretip üretmediğini denetlemek için kanal ve kuyruk yöneticisindeki STATCHL ve STATACLS seçeneklerini kullanabilirsiniz.

z/OS IBM MQ for z/OSyönetimine ilişkin terimler ve görevler

Bu konuyu terminolojiye giriş olarak kullanın ve IBM MQ for z/OS' e özgü görevler.

IBM MQ for z/OS yönetimi için gereken bazı kayıt ve görevler, z/OS platformuna özgütür. Aşağıdaki listede bu terimlerin ve görevlerin bazıları yer almaktadır.

- Paylaşılan kuyruklar
- Sayfa kümeleri ve arabellek havuzları
- Günlük Kaydı
- Kuyruk yöneticisi ortamının uyarlması
- Yeniden başlatma ve kurtarma
- Güvenlik
- Kullanılabilirlik
- Nesnelerin kullanılması
- İzleme ve İstatistik
- Uygulama ortamları

Paylaşılan kuyruklar

Queues can be *paylaşılmayan*, owned by and accessible to only one queue manager, or *paylaşılıyor*, owned by a *kuyruk paylaşım grubu*. Kuyruk paylaşım grubu, aynı IBM MQ nesne tanımlarına ve ileti verilerine koşutzamanlı olarak erişebilen tek bir z/OS sistem birleşimi (sysplex) içinde çalışan bir dizi kuyruk yöneticisinden oluşur. Bir kuyruk paylaşım grubu içinde, paylaşılabilir nesne tanımlamaları paylaşılan bir Db2 veritabanında saklanır. Paylaşılan kuyruk iletileri bir ya da daha fazla bağlaşım olanağı yapısı (CF yapıları) içinde tutulur. İleti verileri, doğrudan yapıda saklanacak kadar büyükse (63 KB ' den fazla KB) ya da ileti, kuruluş tarafından tanımlanan kuralların yüklenmek üzere seçilmesine yetecek kadar büyükse, ileti denetim bilgileri bağlaşım olanağı girdisinde saklanabilir, ancak ileti verileri paylaşılan bir ileti veri kümesine (SMDS) ya da paylaşılan bir Db2 veritabanına yüklenmektedir. Paylaşılan ileti veri kümeleri, paylaşılan Db2 veritabanı ve bağlaşım tesisi yapıları, gruptaki tüm kuyruk yöneticileri tarafından ortaklaşa yönetilen kaynaklardır.

Sayfa kümeleri ve arabellek havuzları

Bir ileti paylaşılmayan bir kuyruğa konduğunda, kuyruk yöneticisi verileri, sonraki bir işlem aynı kuyruktan ileti aldığında alınabileceği bir şekilde bir sayfa kümesine saklar. İleti kuyruktan kaldırılırsa, verileri tutan

sayfa kümesindeki alan daha sonra yeniden kullanılmak üzere serbest bırakılır. Bir kuyrukta tutulan ileti sayısı arttıkça, sayfa kümesinde kullanılan alan miktarı artar ve kuyrukta ileti sayısı azaldıkça, sayfa kümesinde kullanılan boşluk azalır.

Sayfa kümelerinden veri yazma ve veri okuma performansını azaltmak için, kuyruk yöneticisi güncellemeleri işlemci depolamaya arabelleğe alma olanağına neden olur. Sayfa kümesi erişimini arabelleğe almak için kullanılan depolama alanı miktarı, *arabellek havuzları*adı verilen IBM MQ nesnelerinden denetlenir.

Sayfa kümeleri ve arabellek havuzları hakkında daha fazla bilgi için bkz. Depolama yönetimi.

Günlük Kaydı

Sayfa kümeleri üzerinde tutulan nesnelerle ve kalıcı iletilerde yapılan işlemler, günlük kayıtları olarak kaydedilir. Bu günlük kayıtları, *etkin günlük*adı verilen bir günlük veri kümesine yazılır. Etkin günlük veri kümesinin adı ve boyutu, *önyükleme verileri kümesi* (BSDS) adı verilen bir veri kümesinde tutulur.

Etkin günlük verileri doldurulduğunda, kuyruk yöneticisi günlüğe kaydetme işlemine devam edebilmesi için başka bir günlük veri kümesine geçer ve tam etkin günlük verilerinin içeriğini bir *arşiv günlüğü* veri kümesine kopyalar. Önyükleme verileri kümesinde, arşiv günlüğü veri kümesinin adı da içinde olmak üzere, bu işlemlerle ilgili bilgiler tutulur. Kavramsal olarak, etkin günlük veri kümelerinin bir halkası vardır; kuyruk yöneticisi çevrimiçidir; etkin bir günlük dolduğunda, günlük verileri bir arşiv günlüğüne boşaltılır ve etkin günlük veri kümesi yeniden kullanım için kullanılabilir.

Günlük ve önyükleme veri kümeleriyle ilgili daha fazla bilgi için bkz. “oturum açmaIBM MQ for z/OS” sayfa 222.

Kuyruk yöneticisi ortamının uyarması

Kuyruk yöneticisi başlatıldığında, kuyruk yöneticisinin nasıl çalışacağını denetleyen bir kullanıma hazırlama değiştirgeleri kümesi. Ayrıca, IBM MQ komutlarını içeren veri kümeleri okunur ve içerdikleri komutlar yürütülmektedir. Genellikle bu veri kümeleri, IBM MQ ' un çalışması için gerekli olan sistem nesnelerinin tanımlarını içerir ve bunları, işletim ortamınız için gerekli olan IBM MQ nesnelerini tanımlamak ya da başlatmak üzere uyarlayabilirsiniz. Bu veri kümeleri okunduğunda, bunlar tarafından tanımlanan nesneler bir sayfa kümesinde ya da Db2içinde depolanır.

Kullanıma hazırlama değiştirgeleri ve sistem nesnelerine ilişkin ek bilgi için bkz. “Defining your system on z/OS” sayfa 233.

Kurtarma ve yeniden başlatma

IBM MQişletim sistemi işlemi sırasında herhangi bir zamanda, işlemci depolamada henüz sayfa kümesine yazılmamış olan değişiklikler olabilir. Bu değişiklikler, kuyruk yöneticisi içinde bir arka plan görevi tarafından en az kullanılan sayfa kümesine yazılır.

Kuyruk yöneticisi olağan dışı bir şekilde sonlanırsa, kalıcı ileti verileri günlük kayıtlarında tutulduğu için, kuyruk yöneticisi yeniden başlatma işlemi kurtarma aşaması, sayfa kümesinin kaybedilen sayfa kümesini kurtarma işlemini kurtarabilir. Bu, IBM MQ ' in kalıcı ileti verilerini ve nesne değişikliklerini hata noktasına kadar kurtarabileceği anlamına gelir.

Bir kuyruk paylaşım grubunun üyesi olan bir kuyruk yöneticisi bir bağlaşım olanağı hatasıyla karşılaşırsa, bu kuyruktaki kalıcı iletiler yalnızca bağlaşım olanağı yapılarınızı yedeklediyseniz kurtarılabilir.

Kurtarma ve yeniden başlatma hakkında daha fazla bilgi için bkz. “Recovery and restart on z/OS” sayfa 244.

Güvenlik

Security Server (önceden RACF olarak bilinen) gibi bir dış güvenlik yöneticisi kullanabilirsiniz. IBM MQ ' in sahip olduğu kaynakları korumak ve yetkisiz kullanıcılar tarafından erişimden yönetebilmek için. Kanal güvenliği için TLS (Transport Layer Security; İletim Katmanı Güvenliği) olanağını da kullanabilirsiniz. TLS, IBM MQ ürününün bir parçası olarak dahil edilir.

IBM MQ güvenliği hakkında daha fazla bilgi için bkz. “IBM MQ for z/OSiçindeki güvenlik kavramları” sayfa 260.

Kullanılabilirlik

Kuyruk yöneticisi ya da iletişim altsistemi hatası durumunda sistem kullanılabilirliğini artırmak üzere tasarlanmış IBM MQ ' un çeşitli özellikleri vardır. Bu özelliklerle ilgili daha fazla bilgi için bkz. “z/OSüzerinde kullanılabilirlik” sayfa 266.

Nesneleri işleme

Kuyruk yöneticisi çalışırken, IBM MQ nesnelerini bir z/OS konsol arabirimi aracılığıyla ya da TSO altında ISPF hizmetleri kullanan bir yönetim yardımcı programı aracılığıyla yönlendirebilirsiniz. Her iki mekanizma da IBM MQ nesnelerini tanımlamanızı, değiştirmenizi ya da silmenizi sağlar. Ayrıca, çeşitli IBM MQ ve kuyruk yöneticisi işlevlerinin durumunu da denetleyebilir ve görüntüleyebilirsiniz.

You can also manipulate IBM MQ objects using the IBM MQ Explorer, a graphical user interface that provides a visual way of working with queues, queue managers, and other objects.

Bu olanaklarla ilgili daha fazla bilgi için bkz. Verme komutları.

İzleme ve İstatistik

Kuyruk yöneticilerinizi ve kanal kullanıma hazırlayıcılarınızı izlemek için birkaç olanak vardır. Ayrıca, performans değerlendirmesi ve muhasebe amacıyla istatistikler toplayabilirsiniz.

Bu olanaklarla ilgili daha fazla bilgi için bkz. “IBM MQ for z/OSile ilgili izleme ve istatistikler” sayfa 270.

Uygulama ortamları

Kuyruk yöneticisi başlatıldığında, uygulamalar buna bağlanabilir ve IBM MQ API ' yı kullanmaya başlayabilirler. Bunlar CICS, IMS, Batch ya da WebSphere Application Server uygulamaları olabilir. IBM MQ applications can also access applications on CICS and IMS systems that are not aware of IBM MQ, using the CICS and IMS bridges.

Bu olanaklarla ilgili daha fazla bilgi için bkz. “IBM MQ ve diğer z/OS ürünleri” sayfa 274.

IBM MQ uygulamalarını yazma hakkında bilgi için aşağıdaki belgelere bakın:

- Uygulamaları geliştirme
- C++ kullanılması
- IBM MQ classes for Java Olanağının Kullanılması

z/OS Paylaşılan kuyruklar ve kuyruk paylaşım grupları

You can use shared queues and queue sharing groups, to implement high availability of IBM MQ resources. Paylaşılan kuyruklar ve kuyruk paylaşım grupları, z/OS altyapısındaki IBM MQ for z/OS için benzersiz işlevlerdir.

Bu bölümde, öznitelikler ve avantajlar anlatılır ve kaç kuyruk yöneticisinin aynı kuyrukları ve bu kuyruklardaki iletileri paylaşabileceği hakkında bilgiler sunar.

İlgili kavramlar

“Paylaşılan kuyruk nedir?” sayfa 161

Paylaşılan kuyruk, yerel bir kuyruğun tipidir. Bir ya da daha çok kuyruk yöneticisi tarafından bir sistem birleşimi (sysplex) içinde olan iletiler bu kuyruğa erişebilir.

“Kuyruk paylaşım grubu nedir?” sayfa 162
Aynı paylaşılan kuyruklara erişebilen kuyruk yöneticilerinden oluşan bir grup, kuyruk paylaşım grubu adı verilir. Kuyruk paylaşım grubunun her üyesinin, aynı paylaşılan kuyruklara erişimi vardır.

“Paylaşılan kuyruk iletileri nerede tutuluyor?” sayfa 164
Paylaşılan kuyruktaki her ileti, z/OS bağlaşım olanağı listesi yapısındaki bir girişle gösterilir. İleti verileri aynı girdiye sığmayacak kadar büyükse, paylaşılan bir ileti veri kümesine (SMDS) ya da Db2' e boşaltılır.

“Paylaşılan kuyruklar kullanmanın avantajları” sayfa 180
Paylaşılan kuyruk, IBM MQ uygulamalarının ölçeklenebilir, yüksek düzeyde kullanılabilir olmasını ve iş yükü dengelemenin gerçekleştirilmesini sağlar.

“Dağıtılmış kuyruğa alma ve kuyruk paylaşım grupları” sayfa 199
Dağıtılmış kuyruklama ve kuyruk paylaşım grupları, uygulama sistemlerinizin kullanılabilirliğini artırmak için kullanabileceğiniz iki yöntemdir. Bu tekniklerle ilgili daha fazla bilgi bulmak için bu konuyu kullanın.

“Paylaşılan kuyruklarla iş yükü dağılımını etkileme” sayfa 203
Bir kuyruk paylaşım grubundaki paylaşılan kuyruklarla iş yükü dağılımını etkileyen faktörleri anlamak için bu konuyu kullanın.

İlgili başvurular

“Paylaşılan kuyruklar ve kuyruk paylaşım grupları hakkında daha fazla bilgi bulabileceğiniz yerler” sayfa 205
IBM MQ for z/OS ' in paylaşılan kuyrukları ve kuyruk paylaşım gruplarını nasıl kullandığı hakkında daha fazla bilgi bulmak için bu konudaki tabloyu kullanın.

z/OS Paylaşılan kuyruk nedir?

Paylaşılan kuyruk, yerel bir kuyruğun tipidir. Bir ya da daha çok kuyruk yöneticisi tarafından bir sistem birleşimi (sysplex) içinde olan iletiler bu kuyruğa erişebilir.

Kuyruk paylaşım grubu

Aynı paylaşılan kuyruklar kümesine erişebilen kuyruk yöneticileri, *kuyruk paylaşım grubu*adı verilen bir grubu oluşturur.

Herhangi bir kuyruk yöneticisi iletilere erişebilir

Kuyruk paylaşımı grubundaki herhangi bir kuyruk yöneticisi paylaşılan bir kuyruğa erişebilir. Başka bir deyişle, bir kuyruk yöneticisinden paylaşılan bir kuyruğa ileti yerleştirebilir ve kuyruktan aynı iletiyi farklı bir kuyruk yöneticisinden alabilirsiniz. Bu, kuyruk yöneticileri arasında kanal gerektirmeyen bir kuyruk paylaşım grubu içinde iletişim için hızlı bir mekanizma sağlar.

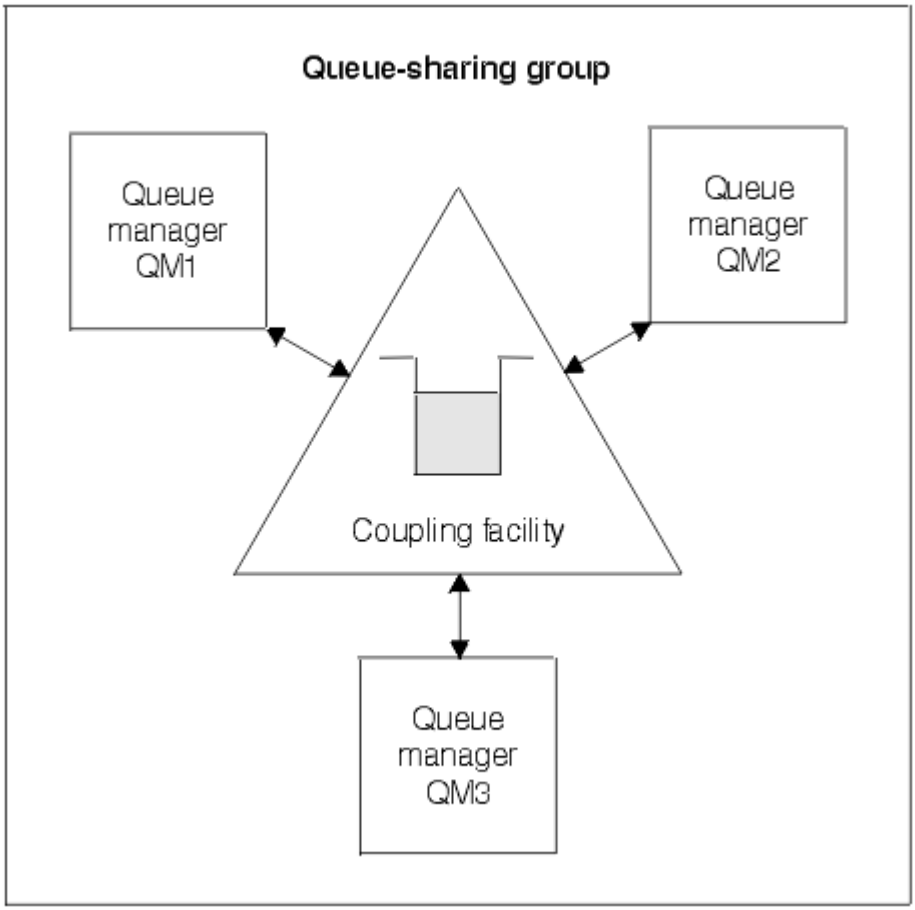
IBM WebSphere MQ 7.1 ve daha sonra, iletilerin Db2 ' e ya da paylaşılan bir ileti veri kümesine (SMDS) boşaltma işlemi desteklenir. Herhangi bir boyuttan ileti boşaltma işlemi yapılandırılabilir.

IBM MQ' in önceki sürümlerinde, büyük iletiler (> 63 KB), bağlaşım tesisinde (4 K) saklanmış bir yertutucu ve Db2içinde saklanan ileti verilerini içerir.

Şekil 59 sayfa 162 , bir kuyruk paylaşım grubu oluşturan üç kuyruk yöneticisini ve bir bağlaşım olanağını gösterir. Tüm üç kuyruk yöneticisi, bağlaşım tesisinde paylaşılan kuyruğa erişebilir.

Bir uygulama, kuyruk paylaşım grubu içindeki kuyruk yöneticilerine bağlanabilir. Kuyruk paylaşım grubundaki tüm kuyruk yöneticileri tüm paylaşılan kuyruklara erişebileceğinden, uygulama belirli bir kuyruk yöneticisinin kullanılabilirliğine bağlı değildir; kuyruk paylaşım grubundaki herhangi bir kuyruk yöneticisi kuyruğa hizmet verebilir.

Kuyruk paylaşım grubundaki diğer tüm kuyruk yöneticileri, kuyruk yöneticilerinden birinin bir sorunu varsa, kuyruğun işlenmesine devam edebileceğinden, bu işlem daha yüksek kullanılabilirlik sağlar.



Şekil 59. Kuyruk paylaşım grubu

Kuyruk tanımlaması tüm kuyruk yöneticileri tarafından paylaşılıyor

Paylaşılan kuyruk tanımlamaları, OBJ_B_QUEUE adlı Db2 veritabanı çizelgesinde saklanır. Bu yüzden, kuyruğu yalnızca bir kez tanımlamanız ve sonra da kuyruk paylaşım grubundaki tüm kuyruk yöneticilerine erişilmesi gerekir. Bunun anlamı, daha az tanımın olması anlamına gelir.

Buna karşılık, paylaşılmayan bir kuyruğun tanımı, kuyruğun sahibi olan kuyruk yöneticisinin sıfır olan sayfa kümesinde saklanır (Sayfa kümeleri ' nde açıklandığı gibi).

Tanımlayan kuyruk yöneticisinin sayfa kümelerinde o adı taşıyan bir kuyruk önceden tanımlandıysa, paylaşılan bir kuyruk tanımlayamazsınız. Benzer şekilde, aynı adı taşıyan bir paylaşılan kuyruk varsa, kuyruk yöneticisi sayfası kümelerinde bir kuyruğun yerel sürümünü tanımlayamazsınız.

z/OS Kuyruk paylaşım grubu nedir?

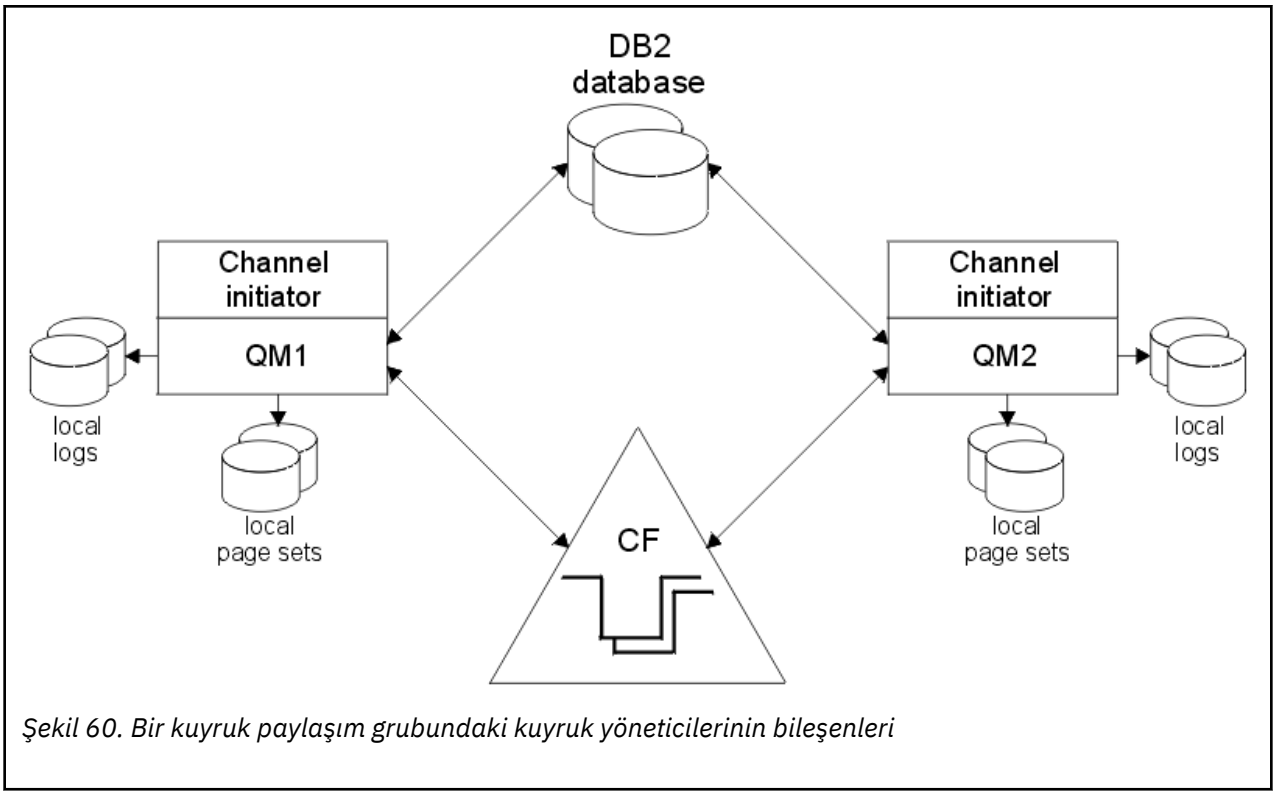
Aynı paylaşılan kuyruklara erişebilen kuyruk yöneticilerinden oluşan bir grup, kuyruk paylaşım grubu adı verilir. Kuyruk paylaşım grubunun her üyesinin, aynı paylaşılan kuyruklara erişimi vardır.

Kuyruk paylaşım grupları en çok dört karakterden oluşan bir ada sahiptir. Adın ağınızda benzersiz olması ve kuyruk yöneticisi adlarından farklı olması gerekir.

Şekil 60 sayfa 163 , iki kuyruk yöneticisi içeren bir kuyruk paylaşım grubunu gösterir. Her kuyruk yöneticisinin bir kanal başlatıcısı ve kendi yerel sayfa kümeleri ve günlük veri kümeleri vardır.

Kuyruk paylaşım grubunun her bir üyesi de bir Db2 sistemine bağlanmalıdır. Kuyruk yöneticilerinin, paylaşılan nesne tanımlamalarını tutmak için kullanılan Db2 paylaşılan havuzuna erişebilmeleri için, Db2 sistemlerinin tümü aynı Db2 veri paylaşımı grubunda olmalıdır. Bunlar, yalnızca bir kez tanımlanan herhangi bir IBM MQ nesnesi tipine (örneğin, kuyruklara ve kanallara) ilişkin tanımlardır ve gruptaki herhangi bir kuyruk yöneticisi bunları kullanabilir. Bunlar *genel* tanımlamaları olarak adlandırılır ve Özel ve genel tanımlamalariçinde açıklanır.

Belirli bir veri paylaşma grubuna birden çok kuyruk paylaşım grubu gönderme yapabilir. You specify the name of the Db2 subsystem and which data-sharing group a queue manager uses in the IBM MQ system parameters at startup.



Şekil 60. Bir kuyruk paylaşım grubundaki kuyruk yöneticilerinin bileşenleri

Bir kuyruk yöneticisi bir kuyruk paylaşım grubuna katıldığında, o grup için tanımlanmış paylaşılan nesnelere erişimi vardır ve bu kuyruk yöneticisini grup içinde yeni paylaşılan nesneler tanımlamak için kullanabilirsiniz. Grup içinde paylaşılan kuyruklar tanımlandıysa, bu kuyruk yöneticisini kullanarak, bu paylaşılan kuyruklara ileti koymak ve bu paylaşılan kuyruklardan ileti almak için kullanabilirsiniz. Gruptaki kuyruk yöneticisi, paylaşılan bir kuyrukta tutulan iletileri alabilir.

Bir MQSC komutu bir kez girebilir ve kuyruk paylaşım grubundaki tüm kuyruk yöneticilerine tek tek girilmiş gibi, kuyruk paylaşım grubu içindeki tüm kuyruk yöneticilerini de yürütmesini sağlar. Bu öğe için *komut kapsamı* özniteliği kullanılır. Bu öznitelik, Farklı kuyruk yöneticilerine komut yöneltmesibaşlıklı konuda açıklanmaktadır.

Kuyruk yöneticisi bir kuyruk paylaşım grubunun bir üyesi olarak çalıştığında, kuyruk paylaşım grubundaki tüm kuyruk yöneticileri tarafından kullanılabilecek, genel olarak tanımlı olan kuyruk yöneticisine ve genel olarak tanımlı IBM MQ nesnelerine özel olarak tanımlanan IBM MQ nesneleri arasında ayrım yapmak mümkün olmalıdır. Bu öğe için *kuyruk paylaşımı grubu yok etme* özniteliği kullanılır. Bu öznitelik Özel ve genel tanımlamalariçinde açıklanmaktadır.

Grup içindeki herhangi bir yerde IBM MQ nesnelerine erişimi denetleyen tek bir güvenlik profili kümesi tanımlayabilirsiniz. Bu, tanımlamak zorunda olduğunuz profillerin sayısının büyük ölçüde azaltıldığı anlamına gelir.

Kuyruk yöneticisi yalnızca bir kuyruk paylaşım grubuna ait olabilir ve gruptaki tüm kuyruk yöneticileri aynı sistem birleşimi içinde olmalıdır. Başlatma sırasında sistem parametrelerinde kuyruk yöneticisinin ait olduğu kuyruk paylaşım grubunu belirtiyorsunuz.

z/OS Paylaşılan kuyruk iletileri nerede tutuluyor?

Paylaşılan kuyruktaki her ileti, z/OS bağlaşım olanağı listesi yapısındaki bir girişle gösterilir. İleti verileri aynı girdiye sığmayacak kadar büyükse, paylaşılan bir ileti veri kümesine (SMDS) ya da Db2' e boşaltılır.

CF yapısı Sistem Sınıfı Bellek (SCM) kullanacak şekilde yapılandırıldıysa, IBM MQ ek yapılandırma olmadan bunu kullanabilir. Bkz. IBM MQ V8 Özellikler ve Geliştirmeler, Bölüm 8.

Önemli: IBM z16 , Coupling Facility görüntüleri için Virtual Flash Bellek (Depolama Sınıfı Bellek ya da SCM olarak da bilinir) kullanımını desteklemek üzere son nesil IBM Z ® olması planlanmıştır. Daha fazla bilgi için bkz. IBM Z ve IBM LinuxONE 4Q 2023 Yönelimin Bildirimleri.

Alternatif olarak, daha büyük yapılar kullanmalı ya da iletileri SMDS ' ye boşaltmalısınız.

Paylaşılan kuyruk iletisi saklama alanı

Paylaşılan kuyruklara konan iletiler sayfa kümelerinde saklanmaz ve arabellek havuzlarını kullanmaz.

Paylaşılan kuyruklardaki iletilerin, z/OS bağlaşım olanağındaki (CF) liste yapılarında girişleri var. Aynı sistem şebekesinde bulunan birçok kuyruk yöneticisi, CF liste yapısını kullanarak bu iletilere erişebilir.

Küçük paylaşılan kuyruk iletilerine ilişkin ileti verileri genellikle bağlaşım olanağı girişinde bulunur. Daha büyük iletiler için, ileti verileri bir paylaşılan ileti veri kümesinde (SMDS) ya da bir Db2 veri paylaşım grubu tarafından paylaşılan bir Db2 çizelgesinde bir ya da daha fazla ikili büyük nesne (BLOB) olarak saklanabilir. 63 KB 'yi aşan ileti verileri her zaman SMDS ya da Db2' e yüklenir. Daha küçük iletiler, bağlaşım olanağı yapısında yer tasarrufu sağlamak için isteğe bağlı olarak aynı şekilde boşaltılabilir. Daha fazla ayrıntı için bkz. “Paylaşılan iletiler için boşaltma seçeneklerini belirtme” sayfa 166 .

Paylaşılan kuyruğa konan iletilere, bir MQGET tarafından alınıncaya kadar bağlaşım olanağı yapısında gönderme yapılır. Bağlaşım olanağı işlemleri aşağıdaki işlemler için kullanılır:

- Sonraki alınabilir iletiyi ara
- Paylaşılan kuyruklardaki kesinleştirilmemiş iletileri kilitle
- İlgili kuyruk yöneticilerine kesinleştirilen iletilerin gelişini bildir

Kalıcı iletiler üzerinde MQPUT ve MQGET işlemleri, o işlemi gerçekleştiren kuyruk yöneticisinin günlüğüne kaydedilir. Bu, bir bağlaşım olanağı arızası durumunda veri kaybı riskini en aza indirir.

Bağlaşım olanağı

Paylaşılan kuyruklarda tutulan iletilere bir bağlaşım olanağı içinde gönderme yapılıyor. Bağlaşım olanağı, sistem şebekesinde bulunan z/OS görüntülerinin herhangi birinin dışında bulunur ve genellikle farklı bir güç kaynağında çalışacak şekilde yapılandırılır. Bu nedenle, bağlaşım olanağı yazılım hatalarına dayanıklıdır ve bunu, donanım arızalarına ya da güç kesintilerine dayanıklı olacak şekilde yapılandırabilirsiniz. Bu, bağlaşım olanağında saklanan iletilerin yüksek kullanılabilirlikli olduğu anlamına gelir.

IBM MQ tarafından kullanılan her bağlaşım olanağı listesi yapısı belirli bir kuyruk paylaşım grubuna ayrılır, ancak bir bağlaşım olanağı birden çok kuyruk paylaşım grubuna ilişkin yapıları tutabilir. Farklı kuyruk paylaşım gruplarındaki kuyruk yöneticileri verileri paylaşamaz. Bir kuyruk paylaşım grubundaki 32 'ye kadar kuyruk yöneticisi aynı anda bir bağlaşım olanağı listesi yapısına bağlanabilir.

Tek bir bağlaşım olanağı listesi yapısı en çok 512 paylaşılan kuyruk içerebilir. Yapıda depolanan ileti verilerinin toplam miktarı yapı kapasitesiyle sınırlıdır. Ancak **CFLEVEL(5)** ile, 63 KB ' den küçük iletilere

ilişkin verileri boşaltmak için boşaltma değiştirgelerini kullanabilirsiniz; böylece, her ileti en az bir bağlaşım olanağı girişi ve en az 512 bayt veri gerektirir.

Liste yapısının boyutu aşağıdaki etmenlerle sınırlıdır:

- Tek bir bağlantı tesisinde olmalı.
- Kullanılabilir bağlaşım olanağı depolamasını IBM MQ ve diğer ürünlere ilişkin diğer yapılarla paylaşabilir.

Bağlaşım olanağı listesi yapılarıyla ilişkilendirilmiş depolama sınıfı belleği olabilir. Bazı durumlarda bu depolama sınıfı belleği, paylaşılan kuyruklarla kullanıldığında yararlı olabilir. Ek bilgi için bkz. “Paylaşılan kuyruklar ile depolama sınıfı belleğinin kullanılması” sayfa 181 .

Not: IBM MQ kullanırken bir bağlaşım olanağı yapısını şifreleyebilirsiniz. Daha fazla bilgi için bkz. Eşleme tesisi yapısı verilerinin şifrelenmesi .

CF yapısı boyutunun planlanması

CF yapılarınızın boyutlandırılmasına ilişkin kılavuza gereksinim duyarsanız, MP16: IBM MQ for z/OS Capacity planning and tuning supportpac komutunu kullanabilirsiniz. CF boyutlarına yardımcı olmak için IBM tarafından sağlanan web tabanlı CFSizeraracını da kullanabilirsiniz.

CF yapısı nesnesi

Kuyruk yöneticisinin bağlaşım olanağı yapısı kullanımı bir CF yapısı (CFSTRUCT) IBM MQ nesnesinde belirtilir.

Bu yapı nesneleri Db2içinde saklanır.

Bir bağlaşım olanağı yapısıyla ilgili z/OS komutlarını ya da tanımlamalarını kullanırken, kuyruk paylaşım grubu adının ilk dört karakteri gereklidir. Ancak, bir IBM MQ CFSTRUCT nesnesi her zaman tek bir kuyruk paylaşım grubunda bulunur ve bu nedenle adı, kuyruk paylaşım grubunun adının ilk dört karakterini içermez. Örneğin, SQ03 ile başlayan kuyruk paylaşım grubunda tanımlanan CFSTRUCT (MYDATA), SQ03MYDATAbağlaşım olanağı liste yapısını kullanır.

CF yapılarının işlevsel yeteneklerini belirleyen bir CFLEVEL özniteliği vardır:

- 1, 2-63 KB ' den küçük kalıcı olmayan iletiler için kullanılabilir
- 3-63 KB ' den küçük kalıcı ve kalıcı olmayan iletiler için kullanılabilir
- 4-100 MB ' ye kadar kalıcı ve kalıcı olmayan iletiler için kullanılabilir
- 5-100 MB 'ye kadar kalıcı ve kalıcı olmayan iletiler için kullanılabilir ve seçmeli olarak paylaşılan ileti veri kümelerine (SMDS) veya Db2' e boşaltılabilir.

Bağlaşım olanağının yedeklenmesi ve kurtarılması

IBM MQ BACKUP CFSTRUCT komutunu kullanarak bağlaşım olanağı listesi yapılarını yedekleyebilirsiniz. Bu işlem, CF yapısındaki kalıcı iletilerin bir kopyasını yedeklemeyi yapan kuyruk yöneticisinin etkin günlük verileri kümesine koyar ve yedeklemenin bir kaydını Db2' e yazar.

Bağlaşım olanağı arızalanırsa, IBM MQ RECOVER CFSTRUCT komutunu kullanabilirsiniz. Bu, CF yapısının yedeğindeki kalıcı iletileri bulmak ve geri yüklemek için Db2 yedek kaydını kullanır. Son yedeklemenin kuyruk paylaşım grubundaki tüm kuyruk yöneticilerinin günlükleri kullanılarak yeniden yürütülmesinden bu yana gerçekleştirilen herhangi bir etkinlik ve CF yapısı hatadan önceki noktaya kadar geri yüklenir.

Ek bilgi için BACKUP CFSTRUCT ve RECOVER CFSTRUCT komutlarına bakın.

İlgili kavramlar

“Paylaşılan iletiler için boşaltma seçeneklerini belirtme” sayfa 166
Bir Db2 tablosunda ya da paylaşılan bir ileti veri kümesinde (SMDS) paylaşılan bir kuyruk iletisi için ileti verilerinin nerede saklanabileceğini seçebilirsiniz. İletinin boyutuna ve bağlaşım olanağı yapısının (CF) geçerli kullanımına bağlı olarak hangi iletilerin yüklendiğini de seçebilirsiniz.

“Paylaşılan ileti verileri kümesi (SMDS) ortamınızın yönetilmesi” sayfa 168
Büyük iletileri boşaltmak için paylaşılan ileti veri kümelerini seçerseniz, IBM MQ ' un bu veri kümelerini yönetmek için kullandığı bilgileri ve bu bilgilerle çalışmak için kullanılan komutları da bilmeniz gerekir. Paylaşılan ileti veri kümelerinin nasıl yönetileceğini anlamak için bu konuyu kullanın.

z/OS *Paylaşılan iletiler için boşaltma seçeneklerini belirtme*

Bir Db2 tablosunda ya da paylaşılan bir ileti veri kümesinde (SMDS) paylaşılan bir kuyruk iletisi için ileti verilerinin nerede saklanabileceğini seçebilirsiniz. İletinin boyutuna ve bağlaşım olanağı yapısının (CF) geçerli kullanımına bağlı olarak hangi iletilerin yüklendiğini de seçebilirsiniz.

Paylaşılan kuyruklara ilişkin ileti verileri, bağlaşım tesisinden yüklenebilir ve bir Db2 çizelgesinde ya da *paylaşılan ileti veri kümesi* (SMDS) adı verilen bir IBM MQ yönetilen veri kümesinde saklanabilir.

63 KB 'lik bağlaşım olanağı giriş büyüklüğünden daha büyük iletiler için, ileti verilerinin bir SMDS' ye boşaltılması, Db2' a yapılan boşaltma ile karşılaştırıldığında önemli bir başarım artışı olabilir.

Her paylaşılan kuyruk iletisi, bir bağlaşım olanağı yapısında liste girişi kullanılarak yönetilmeye devam eder, ancak ileti verileri SMDS ' ye boşaltıldığında, bağlaşım olanağı girişi yalnızca bazı denetim bilgileri ve iletinin saklandığı ilgili disk bloklarına yapılan başvuruların bir listesini içerir. Bu mekanizmanın kullanılması, her ileti için gereken bağlaşım olanağı öğesi saklama alanı miktarı, iletinin gerçek büyüklüğünün yalnızca bir kısmından yalnızca bir kesir.

Paylaşılan kuyruk iletilerinin saklandığı yeri seçme

SMDS ya da Db2 paylaşılan ileti depolaması seçimi, **CFSTRUCT** tanımlamasındaki **OFFLOAD(SMDS|DB2)** parametresiyle denetlenir. Varsayılan değer, '**OFFLOAD(SMDS)** ' değeridir.

Bu parametre, **CFSTRUCT** ' in **CFLEVEL(5)** ya da daha büyük bir kullanım için kullanılmasını gerektirir. Yalnızca IBM WebSphere MQ 7.1 ya da daha sonraki düzeylerde bulunan kuyruk yöneticileri bir CF yapısına bu düzeyde bağlanabilir.

Yalnızca, kuyruk paylaşım grubundaki tüm kuyruk yöneticileri IBM WebSphere MQ 7.1 ya da daha yüksekse, bir yapıyı **CFLEVEL(5)** değerine kadar değiştirmek mümkündür.

OFFLOAD parametresi yalnızca **CFLEVEL(5)** parametresinden geçerlidir. Daha ayrıntılı bilgi için DEFINE CFSTRUCT konusuna bakın.

OFFLOAD(DB2) , öncelikle geçiş amacıyla desteklenir.

Hangi paylaşılan kuyruk iletilerinin yüklendiğini seçme

İleti verileri, ileti verilerinin büyüklüğüne ve bağlaşım olanağı yapısının yürürlükteki kullanımına dayalı olarak SMDS ya da Db2 ' e boşaltılır. Üç kural vardır ve her kural eşleşen bir parametre çiftini belirtir. Bu parametreler, karşılık gelen bir bağlaşım tesisi yapısı kullanım eşiği yüzdesidir (**OFFLDnTH**) ve ileti boyutu sınırı (**OFFLDnSZ**).

Üç kuralın yürürlükteki somutlaması, aşağıdaki anahtar sözcük çiftleri kullanılarak belirtilir:

- OFFLD1TH ve OFFLD1SZ
- OFFLD2TH ve OFFLD2SZ
- OFFLD3TH ve OFFLD3SZ

Kural çifti	Varsayılan değer	Tanım
Kural çifti 1	OFFLD1TH(70) ve OFFLD1SZ(32K)	Bağlaşım olanağı yapısı, 32 KB ' yi aşan iletiler için %70 'ten fazlaysa tam olarak yük dışı
Kural çifti 2	OFFLD2TH(80) ve OFFLD2SZ(4K)	Bağlaşım olanağı yapısı, 4 KB ' yi aşan iletiler için %80 'den fazlaysa tam olarak yük dışı veri

Kural çifti	Varsayılan değer	Tanım
Kural çifti 3	OFFLD3TH(90) ve OFFLD3SZ(0K)	Bağlaşım olanağı yapısı, 0 KB ' yi aşan iletiler için %90 'dan fazlaysa (tüm iletiler)

Bir göreli yükleme kuralı OFFLD *x* SZ değeri 64K ise, bu kuralın yürürlükte olmadığını gösterir. Bu durumda, başka bir boşaltma kuralı geçerli olursa ya da ileti 63.75 KB ' den büyükse ve bu durumda depolanacak çok büyük bir ileti varsa, bu durumda iletiler boşaltılır.

Each message which is offloaded still requires 0.75 KB of storage in the coupling facility.

Her bir yapı için belirlenebilecek üç adet boşaltma kuralı aşağıda belirtildiği şekilde kullanılmak üzere tasarlanmıştır.

- Başarım
  - Uygulama yapısında bir sürü alan olduğunda, ileti verileri yalnızca yapıda saklanacak kadar büyükse ya da alt ileti boyutu eşiğini aşması durumunda, bunu yapıda saklamanın performans değerinin gereksinim duyacak yapı alanı miktarına değmemesi durumunda boşaltılmalıdır.
  - Belirli bir ileti boyutu eşiği gerekliyse, bu ileti ilk göreli yükleme kuralı kullanılarak toplanır.
- Kapasite
  - Uygulama yapısında çok az yer olduğunda, geri kalan alanın en iyi şekilde kullanılmasını sağlamak için ileti verilerinin üst sınır değeri boşaltılmalıdır.
  - Üçüncü göreli konum kuralı, yapı neredeyse tam olduğunda, çoğu iletinin boşaltılması gerektiğini belirtmek için toplanır. Böylece, uygulama yapısındaki girdiler genellikle minimum boyuttan (yaklaşık 0.75K bayt gerektirir) yer alacak.
  - Kullanım eşiği parametresi, uygulama yapısı boyutuna ve beklenen birikim üst sınırına dayalı olarak seçilmelidir. Örneğin, beklenen birikim üst sınırı 1M iletilerse, bu ileti sayısı için gereken yapı deposu miktarı 0.75G bayttır. Bu, örneğin, yapı 10G baytsa, tüm iletilerin boşaltılmasına ilişkin kullanım eşiğinin %92 ya da daha düşük bir değere ayarlanması gerektiği anlamına gelir.
  - Yapı alanı, öğelere ve girdilere ayrılır ve genel olarak yeterli alan olsa da, bunlardan biri diğerinden önce çıkabilir. Sistem, gerektiğinde oranı ayarlamak için AUTOALTER yetenekleri sağlar, ancak bu çok hassas değildir; bu nedenle, kullanılabilir alan miktarı biraz daha az olabilir. Bu nedenle, maksimum yapı alanının %90 'ından fazlasını kullanmamayı hedeflemek daha iyi olabilir. Bu nedenle, önceki örnekte, tüm iletilerin boşaltılmasına ilişkin kullanım eşiği %80 civarında daha iyi bir şekilde ayarlanabilirdi.
- Kesilen geçiş:
  - bağlaşım tesisi yapısında kalan alan miktarı azalırken, performans özelliklerinde büyük bir ani değişme olması istenmeyen bir durum olacaktır. Ayrıca, bağlaşım tesisi yönetiminin, kullanılan öğelerin tipik olarak belirli oranlarda bir ani eşik değişikliğine sahip olması da istenmeyen bir durum değildir.
  - İkinci yük kuralı, performans ve kapasite taraflı offload kuralları arasında bazı ara minder sağlamak için kullanılır. Bağlaşım olanağı yapısında kullanılan alan bir ara eşiği aştığında, yük boşaltma etkinleşmesinde önemli bir artışa neden olacak şekilde ayarlanabilir. Diğer bir deyişle, kalan alan daha yavaş bir şekilde kullanılır ve bağlaşım tesisi otomatik değiştirme işlemini daha yüksek kullanım düzeylerine uyarlamak için daha fazla zaman sağlar.

Bağlaşım olanağı yapısı genişletilemezse ve en az bazı önceden belirlenmiş sayıda ileti saklamaya gerek varsa, üçüncü kural, verilerin önceden belirlenmiş sayıda ileti için ayrılmış olmasını sağlamak üzere, tüm iletiler için verilerin boşaltılmasını uygun bir eşikten başladığından emin olmak için gerekli olduğu şekilde değiştirilebilir.

For example, if the coupling facility structure size is 4 GB, and the predetermined number of messages is 1 million, then 1,000,000 * 0.75 KB are needed, which is 768 MB, 18.75% of 4 GB. Bu durumda, tüm iletilerin boşaltılmasına ilişkin eşik değerinin %90 yerine %80 olarak ayarlanması gerekir. Bu,

OFFLD3TH(80) ve OFFLD3SZ(0K) parametrelerinin yer değiştirmesine neden olur. Diğer boşaltma parametrelerinin de ayarlanması gerekir.

eğer çok küçük mesajların yüklendiği çok küçük mesajların önemli bir performans etkisi olduğu saptansa da, göreceli etki daha büyük iletiler için daha azdır, daha sonra diğer kurallar için kullanım eşikleri daha erken daha büyük mesajlara karşı daha büyük iletiler boşaltmak için azaltılabilir, küçük iletiler için daha fazla yer bırakması gerekmeden önce daha fazla boşluk bırakılabilir.

Örneğin, 32KB ' yi aşan iletiler sık sık oluşursa, ancak bunları boşaltmak için geçen zaman başarımı (RMF istatistiklerinden ya da uygulama performansından saptanırsa), bunları bağlaşım tesisinde tutmak için çok benzerdir; sonra, ilk kural eşiği, tüm bu tür iletileri boşaltmak için 0% olarak ayarlanabilir. Bu, OFFLD1TH(0) ve OFFLD1SZ(32K) parametrelerini verir. Diğer boşaltma parametrelerinin de ayarlanması gerekir.

16 KB ve 6 KB gibi belirli ara büyüklüklerde çok sayıda ileti varsa, ikinci kural için ileti boyutu seçeneğini değiştirmek yararlı olabilir, böylece daha büyük olanlar oldukça düşük bir kullanım eşiğine yüklenerek önemli miktarda alan tasarrufu sağlar, ancak daha küçük olanlar yalnızca bağlaşım tesisinde saklanabilir.

z/OS *Paylaşılan ileti verileri kümesi (SMDS) ortamınızın yönetilmesi*

Büyük iletileri boşaltmak için paylaşılan ileti veri kümelerini seçerseniz, IBM MQ ' un bu veri kümelerini yönetmek için kullandığı bilgileri ve bu bilgilerle çalışmak için kullanılan komutları da bilmeniz gerekir. Paylaşılan ileti veri kümelerinin nasıl yönetileceğini anlamak için bu konuyu kullanın.

SMDS nesneleri

Her paylaşılan ileti veri kümesinin özellikleri ve durumu, kuyruk paylaşım grubundaki herhangi bir kuyruk yöneticisi aracılığıyla güncellenebilen paylaşılan bir SMDS nesnesinde izlenir.

Her bir bağlaşım olanağı uygulama yapısına erişebilen her kuyruk yöneticisi için bir paylaşılan ileti veri kümesi vardır. Paylaşılan ileti veri kümesi, SMDS anahtar sözcüğü kullanılarak belirlenen sahip kuyruk yöneticisi adıyla ve CFSTRUCT anahtar sözcüğü kullanılarak belirlenen uygulama yapısı adıyla tanıtılır.

Not: Bir yapı için SMDS veri kümelerini tanımlarken, her kuyruk yöneticisi için bir tane olmalıdır.

SMDS nesnesi, Db2içinde saklanan ilgili CFSTRUCT nesnesinin uzantısını oluşturan bir dizide (gruptaki kuyruk yöneticisi başına bir giriş) saklanır.

SMDS nesnesi CFSTRUCT nesnesinin bir parçası olarak yaratıldığı ya da silindiği için, SMDS nesnesini DEFINE ya da DELETE komutuna ilişkin bir komut yoktur; ancak, sahip olan tek bir kuyruk yöneticisine ilişkin ayarları değiştirmek için bu nesneyi ALTER (Değiştirme) komutuna ilişkin bir komut vardır.

SMDS komutlarıyla ilgili daha fazla bilgi için bkz. “SMDS ile ilgili komutlar” sayfa 179

SMDSCONN bilgileri

Paylaşılan ileti verileri normal durumda olabilir, ancak bir ya da daha çok kuyruk yöneticisi, örneğin bir güvenlik tanımlamasıyla ya da doğrudan erişim aygıtı bağlantısıyla ilgili bir sorun nedeniyle bu iletiye bağlanamıyor olabilir. Bu nedenle, her kuyruk yöneticisinin bağlantı durumunu ve her paylaşılan ileti veri kümesine ilişkin kullanılabilirlik bilgilerini takip etmesi gerekir; bu bilgiler, örneğin, bağlantı kurup kuramayacağını ve neden bağlanamayacağını gösterir.

SMDSCONN bilgileri, paylaşılan bir ileti veri kümesine yönelik bir kuyruk yöneticisi bağlantısını gösterir. Paylaşılan ileti veri kümesi, CFSTRUCT adıyla birlikte, paylaşılan ileti veri kümesinin iyesi olan kuyruk yöneticisi tarafından tanımlanır (paylaşılan nesnenin kendisi için SMDS anahtar sözcüğünde belirtildiği gibi).

Belirli bir kuyruk yöneticisine gönderilen komutlar yalnızca aynı kuyruk yöneticisine ilişkin SMDSCONN bilgilerine başvurabildiğinden, bağlantı kuyruk yöneticisini tanıtacak bir parametre yoktur.

SMDSCONN bilgi girişleri, sahip olan kuyruk yöneticisindeki ana saklama alanında tutulur ve kuyruk yöneticisi yeniden başlatıldığında yeniden yaratılır. Ancak, tek bir kuyruk yöneticisinden gelen bir bağlantı

belirtik olarak durdurulduysa, bu bilgiler ilgili CFSTRUCT ya da SMDS nesnesindeki bir bağlantı dizisinde işaret olarak saklanır; böylece, kuyruk yöneticisi yeniden başlatma işlemi boyunca devam eder.

Durum ve kullanılabilirlik bilgileri

Durum bilgileri, bir kaynağın ya da bağlantının durumunu gösterir (örneğin, henüz kullanılmadığını, normal kullanımda olduğunu ya da kurtarma ihtiyacı olduğunu). Genellikle STATUS anahtar sözcüğü kullanılarak açıklanır. Olası değerler nesnenin tipine bağlıdır.

Durum bilgileri genellikle otomatik olarak güncellenir; örneğin, kaynak ya da bağlantı kullanılırken bir hata algılandığında. Ancak, bazı durumlarda, bir kuyruk yöneticisinin doğru durumu otomatik olarak saptayamadığı durumlarda, durumu güncellemek için bir komut da kullanılabilir.

Kullanılabilirlik bilgileri, kaynağın ya da bağlantının kullanılıp kullanılamayacağını belirtir ve genellikle öncelikle durum bilgileri tarafından belirlenir. Paylaşılan ileti veri kümesi desteğinde kullanılan kaynak ya da bağlantı tipleri için üç kullanılabilirlik düzeyi uygulanır:

Kullanılabilir
Bu, kaynağın normal olarak kullanılabilmesi anlamına gelir. Bu, şu anda kullanımda olduğu anlamına gelmez (STATUS değerinden belirlenebilir). Bir veri kümesi için, yeniden başlatma işlemi gerektiriyorsa, bu, sahip olan kuyruk yöneticisinin açmasına izin verir, ancak diğer kuyruk yöneticileri veri kümesi ACTIVE durumuna geri dönünceye kadar beklemelidir.

Hata nedeniyle kullanılamıyor
Bu, bir hata nedeniyle kaynağın otomatik olarak kullanılamaz kılındığı ve bazı onarım ya da kurtarma işlemleri gerçekleştirilinceye kadar kaynağın yeniden kullanılabilmesi beklenmediği anlamına gelir. Ancak, işletmen müdahalesi olmadan yeniden kullanılabilir duruma getirmeye izin verilir. Bu tür bir girişim, kaynağı etkin olarak işaretlemek için bir komut ya da kurtarma işleminin tamamlandığını gösterecek şekilde durumu değiştiren bir komut tarafından da tetiklenebilir.

Kaynağın kullanılamaz kılınmasının nedeni genellikle ilgili STATUS değerinden anlaşılır, ancak bazı durumlarda kaynağı kullanılamaz kılmak için başka nedenler de olabilir; bu durumda, nedeni belirtmek için ayrı bir REASON değeri sağlanır.

İşletmen komutu nedeniyle kullanılamıyor
Bu, kaynağa erişimin bir komut tarafından belirtik olarak geçersiz kılındığı anlamına gelir. Yalnızca, yeniden etkinleştirmek için bir komut kullanılarak kullanılabilir kılınabilir.

SMDS kullanılabilirliği

Paylaşılan SMDS nesnesi için, kullanılabilirlik ACCESS anahtar sözcüğüyle açıklanır; olası değerler ENABLED, SUSPENDED ve DISABLED.

Kullanılabilirlik, ACCESS (ENABLED) ya da ACCESS (DISABLED) ayarını tanımlamak için gruptaki herhangi bir kuyruk yöneticisinden ilgili paylaşılan nesne için **RESET SMDS** komutu kullanılarak güncellenebilir.

Kullanılabilirlik daha önce ACCESS (SUSPENDED) ise, ACCESS (ENABLED) olarak değiştirildiğinde, paylaşılan ileti veri kümesini kullanmak için yeni bir girişim tetiklenir, ancak önceki hata devam ediyorsa, kullanılabilirlik yeniden ACCESS (SUSPENDED) olarak sıfırlanır.

SMDSCONN kullanılabilirliği

Yerel bir SMDSCONN bilgi girişi için, kullanılabilirlik AVAIL anahtar sözcüğüyle açıklanır; olası değerler NORMAL, HATA ya da DURDURULDU. Kullanılabilirliği, bağlantısını etkinleştirmek ya da devre dışı bırakmak için belirli bir kuyruk yöneticisine gönderilen bir **START SMDSCONN** ya da **STOP SMDSCONN** komutu kullanılarak güncellenebilir.

Kullanılabilirlik daha önce AVAIL (ERROR) ise, AVAIL (NORMAL) olarak değiştirildiğinde, paylaşılan ileti veri kümesini kullanmak için yeni bir girişim tetiklenir, ancak önceki hata devam ederse, kullanılabilirlik yeniden AVAIL (ERROR) durumuna getirilir.

Paylaşılan ileti verileri kümesi paylaşılan durumu ve kullanılabilirliği

Her paylaşılan ileti veri kümesinin kullanılabilirliği, grup içinde, TYPE (SMDS) ile **DISPLAY CFSTATUS** komutu kullanılarak görüntülenebilecek paylaşılan durum bilgileri kullanılarak yönetilir. Bu, her bir yapı için bir veri kümesini etkinleştirmiş olan her bir kuyruk yöneticisine ilişkin durum bilgilerini görüntüler. Her veri kümesi aşağıdaki durumlardan birinde olabilir:

BULUNAMADİ
Bu, karşılık gelen veri kümesinin henüz etkinleştirilmediği anlamına gelir. Bu durum yalnızca belirli bir kuyruk yöneticisi belirtildiğinde görüntülenir; etkinleştirilmemiş veri kümeleri, tüm kuyruk yöneticileri seçildiğinde atlanır.

YENİ
Veri kümesi ilk kez açılıyor ve kullanıma hazırlanıyor, etkin hale getirilmeye hazır.

ETKİN
Bu, veri kümesinin tamamen kullanılabilir olduğu ve yapı için tüm etkin kuyruk yöneticileri tarafından ayrılması ve açılması gerektiği anlamına gelir.

BAŞARISIZ OLDU
Bu, veri kümesinin (kurtarma işlemi dışında) hiç kullanılamadığı ve tüm kuyruk yöneticileri tarafından kapatılması ve serbest bırakılması gerektiği anlamına gelir.

KURTARMA
Bu, bu veri kümesi için ortam kurtarma işleminin (RECOVER CFSTRUCT kullanılarak) devam etmekte olduğu anlamına gelir.

Tahsil Edildi
Bu, başarısız olan bir veri kümesini etkin duruma geri döndürmek için bir komut verildiğini, ancak henüz tamamlanmamış yeniden başlatma işleminin gerekli olduğunu ve bu nedenle veri kümesinin yalnızca yeniden başlatma işlemi için sahip kuyruk yöneticisi tarafından açılabileceğini gösterir.

EMPTY
Veri kümesi ileti içermiyor. Veri kümesi, herhangi bir ileti içermediğinde, sahip olan kuyruk yöneticisi tarafından olağan bir şekilde kapatılırsa bu duruma getirilir. Uygulama yapısı boşaltıldığından (TYPE PURGE ile **RECOVER CFSTRUCT** kullanılarak ya da yalnızca kurtarılamayan bir yapı için, yapının önceki örneği silinerek) önceki veri kümesi içeriği atıldığında da EMPTY durumuna getirilebilir. Veri kümesinin sahibi olan kuyruk yöneticisi tarafından bir sonraki açılışında, alan eşlemi boş duruma getirilir ve durum ACTIVE olarak değiştirilir. Önceki veri kümesi içeriği artık gerekli olmadığından, bu durumdaki bir veri kümesi, örneğin alan ayırmasını değiştirmek ya da başka bir birime taşımak için yeni ayrılmış bir veri kümesiyle değiştirilebilir.

Komut çıkışı, kurtarma günlük kaydının etkinleştirildiği tarih ve saati (varsa) ve veri kümesinin başarısız olduğu tarih ve saati (etkin değilse) içerir.

Paylaşılan ileti veri kümesi, **RESET SMDS** komutuyla ya da aşağıdaki hata tiplerinden biri saptandığında otomatik olarak FAILED durumuna getirilebiliyor:

- Veri kümesi, sahip olan kuyruk yöneticisi tarafından ayrılamıyor ya da açılamıyor.
- Veri kümesi üstbilgisinin doğrulanması, herhangi bir kuyruk yöneticisi tarafından başarıyla açıldıktan sonra başarısız olur.
- Sahip olan kuyruk yöneticisi veri okurken ya da veri yazarken kalıcı bir G/Ç hatası oluşur.
- Başka bir kuyruk yöneticisi, açık işlemeyi ve doğrulamayı başarıyla tamamlamış bir veri kümesinden veri okurken kalıcı bir G/Ç hatası oluşur.

Bir veri kümesi BAşARıSıZ ya da INRECOVER durumundaysa, olağan kullanım için kullanılamaz; kullanılabilirlik durumu ACCESS (ENABLED) ise, ACCESS (SUSPENDED) olarak değiştirilir.

Bir veri kümesi FAILED durumuna getirildiyse, ancak ortam kurtarma gerekmiyorsa (örneğin, veriler hala geçerli, ancak depolama aygıtı geçici olarak çevrimdışı olduğu için), **RESET SMDS** komutu durumu doğrudan RECOVERY durumuna değiştirmek için kullanılabilir.

Veri kümesi, kurtarma işlemi tamamlandığında ya da **RESET SMDS** komutunun sonucu olarak RECOVERY durumuna girdiğinde, yeniden başlatma işlemi tamamlandıktan sonra yeniden kullanılmaya hazır olur.

ACCESS (SUSPENDED) durumundaysa, otomatik olarak ACCESS (ENABLED) durumuna geri döner ve kuyruk yöneticisinin yeniden başlatma işlemi gerçekleştirmesini sağlar. Yeniden başlatma işlemi tamamlandığında, durum ACTIVE olarak değiştirilir ve diğer tüm kuyruk yöneticileri veri kümesine yeniden bağlanabilir.

Paylaşılan ileti veri kümesi bağlantı durumu ve kullanılabilirliği

Her kuyruk yöneticisi, kendisine ait olan her bir paylaşılan ileti veri kümesine ve gruptaki diğer kuyruk yöneticilerine olan bağlantısına ilişkin yerel durum ve kullanılabilirlik bilgilerini saklar. Bu bilgiler **DISPLAY SMDSCONN** komutu kullanılarak görüntülenebilir.

Başka bir kuyruk yöneticisine ait olan ACTIVE durumundaki bir paylaşılan ileti verilerine erişemezse, bağlantıyı kendi bakış açısından kullanılamaz olarak işaretler.

Hata kesinlikle veri kümesinin kendisiyle ilgili bir sorun olduğunu gösteriyorsa, kuyruk yöneticisi otomatik olarak paylaşılan durumu değiştirerek veri kümesinin BAŞARISIZ durumda olduğunu gösterir. Ancak, hata veri kümesini açma yetkisi olmaması gibi bir ortam sorunundan kaynaklanabiliyorsa, kuyruk yöneticisi hata iletileri yayınlar ve veri kümesini kullanılamıyor olarak işler, ancak paylaşılan veri kümesi durumunu değiştirmez. Ortam hatası veri kümesiyle ilgili bir sorun olarak ortaya çıkarsa (örneğin, bazı kuyruk yöneticilerinin erişemediği bir aygıtta ayrılmışsa), bir işletmen STATUS (FAILED) komutuyla veri kümesinin gerektiği şekilde kurtarılmasına ya da onarılmasına izin vermek için RESET SMDS komutunu kullanabilir.

Paylaşılan bir ileti veri kümesiyle bağlantı kurulamazsa, ancak veri kümesi geçerli görünüyorsa, sahip olan kuyruk yöneticisi için bir **START SMDSCONN** komutu verilerek yeni bir veri kümesi kullanma girişimi tetiklenebilir.

Belirli bir kuyruk yöneticisi ile bir veri kümesi arasındaki bağlantıyı geçici olarak sonlandırma gereksinimi varsa, ancak veri kümesi zarar görmediyse, veri kümesi kapatılabilir ve **STOP SMDSCONN** komutu kullanılarak veri kümesi serbest bırakılabilir. Veri kümesi kullanımdaysa, kuyruk yöneticisi bunu olağan şekilde kapatır (ancak, o veri kümesindeki veri istekleri bir dönüş koduyla reddedilir). Sahip olunan veri kümesiyse, kuyruk yöneticisi CLOSE işlemi sırasında alan eşlemini kaydedip yeniden başlatma işlemine gerek duymamasını sağlar.

If a data set needs to be taken out of service temporarily from all queue managers (for example to move it) but is not damaged, then it is best to use **STOP SMDSCONN** for the relevant data set with the option CMDSCOPE(*) to stop the queue managers using it first, as this will avoid the need for restart processing when the data set is brought back into service. Buna karşılık, veri kümesi FAILED olarak işaretlenirse, bu, kuyruk yöneticilerine bunu hemen kullanmayı bırakmaları gerektiğini bildirir; bu, alan eşleminin saklanmayacağı ve yeniden başlatma işlemiyle yeniden oluşturulması gerekeceği anlamına gelir.

Kuyruk yöneticisi yeniden başlatılırsa, daha önce ACCESS (SUSPENDED) durumunda olan paylaşılan ileti veri kümelerine erişim yeniden denenir.

Paylaşılan ileti veri kümesi kurtarma günlüğü

Kalıcı paylaşılan iletiler, ortam kurtarma amacıyla günlüğe kaydedilir. Bu, kurtarma günlüklerinin bozulmamış olması koşuluyla, bağlaşım olanağı yapılarında ya da paylaşılan ileti veri kümelerinde oluşan herhangi bir hatadan sonra iletilerin kurtarılabileceği anlamına gelir. Kalıcı iletiler, olağanüstü durumdan kurtarma amacıyla başka bir yerdeki kurtarma günlüklerinden de yeniden oluşturulabilir.

İleti verileri paylaşılan bir ileti veri kümesine yazıldığında, veri kümesine yazılan her blok ayrı olarak günlüğe kaydedilir ve bunu, bağlaşım olanağına yazılan ileti girişi (veri eşlemi de içinde olmak üzere) izler. Kurtarma işlemi her zaman bağlaşım olanağı yapısını kurtarır, ancak veri kümesi durumu FAILED dışında tek tek paylaşılan ileti veri kümelerini kurtarması gerekmez ya da durum ACTIVE (Etkin) olduğunda ancak veri kümesi üstbilgisi kaydı artık geçerli olmadığında, veri kümesinin yeniden yaratıldığını gösterir. Durumu ACTIVE ve veri kümesi üstbilgisi hala geçerliyse ya da durumu EMPTY ise, kurtarma için bir veri kümesi seçilmez; bu, hata sırasında içinde ileti saklanmadığını gösterir.

Paylaşılan ileti veri kümesi yedeklemeleri

BACKUP CFSTRUCT, bir uygulama yapısındaki paylaşılan iletilerin yedeğini almak için kullanıldığında, paylaşılan ileti veri kümelerinde saklanan kalıcı iletilere ilişkin veriler, daha önce veritabanında saklanan kalıcı paylaşılan iletilere ilişkin veriler aynı anda yedeklenir.

Paylaşılan ileti veri kümesi kurtarma

Paylaşılan bir ileti veri kümesi bozulduysa ya da kaybolduysa, kuyruk yöneticilerinin onarılıncaya kadar bunu kullanmasını durdurmak için FAILED (Başarısız) durumuna getirilmesi gerekir. Bu olağan durumda otomatik olarak gerçekleşir, ancak **RESET SMDS** komutu STATUS (FAILED) belirtilerek de yapılabilir.

Paylaşılan ileti veri kümesi kalıcı iletiler içeriyorsa, bunlar RECOVER CFSTRUCT komutu kullanılarak kurtarılabilir. Bu komut, önce o paylaşılan ileti verilerine ilişkin kalıcı ileti verilerini en son BACKUP CFSTRUCT komutundan geri yükler, daha sonra o zamandan bu yana günlüğe kaydedilen tüm değişiklikleri uygular. Veri kümesinin ilk etkinleştirildiği zamandan bu yana herhangi bir **BACKUP CFSTRUCT** komutu gerçekleştirilmediyse, etkinleştirme uygulandığından bu yana tüm değişiklikler boş olarak sıfırlanır.

CFSTRUCT içeriği ve paylaşılan ileti veri kümeleri kullanılamıyorsa (örneğin, olağanüstü durumdan kurtarma durumunda), bunların tümü tek bir **RECOVER CFSTRUCT** komutunda kurtarılabilir.

Paylaşılan bir ileti veri kümesi zarar gördüyse, ancak CFSTRUCT için kurtarma etkin değilse ya da en son BACKUP CFSTRUCT değerini içeren günlük kullanılamıyorsa ya da kullanılamıyorsa, o veri kümesine boşaltılan iletiler kurtarılamaz. Bu durumda, PURGE (TYPE) parametresiyle **RECOVER CFSTRUCT** komutu, paylaşılan ileti veri kümesini boş olarak işaretlemek ve o veri kümesinde saklanan verileri olan yapıdan iletileri silmek için kullanılabilir.

RECOVER CFSTRUCT komutu verildiğinde, paylaşılan ileti veri kümesi durumu FAILED değerinden INRECOVER değerine çevrilir. Kurtarma başarıyla tamamlanırsa, durum otomatik olarak SAKLANDI olarak değiştirilir, tersi durumda BAŞARISIZ olarak değişir.

Veri kümesi KURTULDU durumuna değiştirildiğinde, bu, sahip olan kuyruk yöneticisine artık veri kümesini açmayı ve yeniden başlatma işlemini gerçekleştirmeyi deneyebildiğini bildirir.

Paylaşılan ileti veri kümesi kurtarma ve eşitleme noktaları

Paylaşılan ileti veri kümesi kurtarma işlemi, eşitleme noktalarından bağımsız olarak, günlüğün sonuna kadar tüm günlük kayıtları için değişiklikleri yeniden uygular.

Eşitleme noktası içinde değişiklik yapıldıysa, CFSTRUCT için yeniden başlatma ya da kurtarma işlemi, kesinleştirilmemiş isteklerin geri alınmasına neden olabilir; bu nedenle, kurtarılan değişikliklerin bazıları gerçekten kullanılmayabilir, ancak bunların kurtarılmasında herhangi bir sakınca yoktur.

Kesinleştirilmemiş bir MQPUT iletisi yapıya yazılmış olabilir, ancak karşılık gelen veriler veri kümesine ya da günlüğe yazılmamış olabilir (G/Ç tamamlaması yalnızca eşitleme noktası işlemesinin başlangıcında zorlandığı için). Yeniden başlatma işlemi yapıdaki ileti girişini geri aldığından, kurtarılamayan verilere başvurması önemli olmadığından, bu zararsızdır.

Paylaşılan ileti veri kümesi yeniden başlatma işlemi

Bir CFSTRUCT ' ye yönelik kuyruk yöneticisi bağlantısı olağan bir şekilde sona ererse, kuyruk yöneticisi, veri kümesi kapatılmadan hemen önce, veri kümesi içindeki bir denetim noktası alanına ayarlanmış her bir paylaşılan ileti verileri için boş blok alanı haritasını yazar. Bundan sonra, CFSTRUCT ya da paylaşılan ileti veri kümesinin sonraki yeniden başlatmadan önce herhangi bir kurtarma işlemi gerektirmediği sürece, alan eşlemi bağlantı yeniden başlatma sırasında yeniden okunabilir.

Ancak, bir kuyruk yöneticisi olağandışı sonlanırsa ya da yapı ya da veri kümesi herhangi bir kurtarma işlemi gerektiriyorsa, yapıya yönelik kuyruk yöneticisi bağlantısı yeniden başlatıldığında alan eşlemini dinamik olarak yeniden oluşturmak için ek işlem gerekir.

Veri kümesinin kurtarılmasına gerek olmaması koşuluyla, kuyruk yöneticisi yeniden başlatması, yürürlükteki kuyruk yöneticisine ait ileti verilerine yönelik başvuruları bulmak için yapının yürürlükteki

içeriğini tarar ve ilgili veri bloklarını alan eşleminde olduğu gibi işaretler. Diğer kuyruk yöneticileri, alan eşlemi yeniden oluşturulurken yapıyı kullanmaya ve yeniden başlatma kuyruk yöneticisine ait verileri okumaya devam edebilir.

Kurtarmadan sonra paylaşılan ileti verileri kümesi yeniden başlatıltı

Paylaşılan bir ileti veri kümesinin yedekten kurtarılması gerektiyse, veri kümesinde saklanan tüm kalıcı olmayan iletiler kaybolur ve veri kümesi TYPE (PURGE) kullanılarak kurtarılırsa, veri kümesinde saklanan tüm iletiler kaybolur. Kurtarma işlemi tamamlanıncaya kadar, veri kümesi FAILED ya da INRECOVER olarak işaretlenir; böylece, etkilenen iletilerden birini başka bir kuyruk yöneticisinden okuma girişimi, veri kümesinin geçici olarak kullanılamadığını belirten bir hata kodu döndürür.

Veri kümesi kurtarıldığında, durum KURTULDU olarak değiştirilir; bu, sahip olan kuyruk yöneticisinin yeniden başlatma işlemi için açmasına izin verir, ancak veri kümesi diğer kuyruk yöneticileri tarafından kullanılamaz. Kuyruk yöneticisi yeniden başlatma, kalan iletiler için alan eşlemini yeniden oluşturmak üzere yapıyı tarar. Tarama ayrıca, verilerin kaybolduğu iletileri denetler ve bunları yapıdan siler (ya da gerekirse, daha sonra silinecek kayıp olarak işaretler).

Bu yeniden başlatma taraması tamamlandığında, veri kümesi durumu otomatik olarak KURTULDU değerinden ETKİN değerine değiştirilir; bu durumda diğer kuyruk yöneticileri yeniden kullanmaya başlayabilir.

Paylaşılan ileti veri kümesi kullanım bilgileri

DISPLAY USAGE komutu, şu anda açık olan paylaşılan ileti veri kümeleri için paylaşılan ileti veri kümesi alanı ve arabellek havuzu kullanımına ilişkin bilgileri de gösterir. Bu bilgiler, yeni seçenek TYPE (SMDS) ya da var olan seçenek TYPE (ALL) belirtilirse görüntülenir.

Paylaşılan ileti verileri başarımı ve kapasiteyle ilgili önemli noktalar

Veri kümesi kullanımını izleme

Sahip olunan her bir paylaşılan ileti veri kümesinin geçerli yüzdesi, **DISPLAY USAGE** komutuyla **TYPE(SMDS)** seçeneğiyle görüntülenebilir.

DSEXPAND(YES) seçeneğinin SMDS tanımlaması için geçerli olması koşuluyla, kuyruk yöneticisi, paylaşılan bir ileti veri kümesini %90 'a ulaştığında otomatik olarak genişletir. Bu, SMDS seçeneği **DSEXPAND(YES)** olarak ayarlandığında ya da SMDS seçeneği **DSEXPAND(DEFAULT)** olarak ayarlandığında ve CFSTRUCT varsayılan seçeneği **DSEXPAND(YES)** olarak ayarlandığında geçerlidir.

Veri kümesi yaratıldığında ikincil ayırma büyüklüğü belirtilmediği için genişletme girişimi başarısız olursa (203 neden koduyla IEC070I iletisi verilir) Kuyruk yöneticisi, yürürlükteki büyüklüğün yaklaşık %20 'si olan ikincil ayırmayı geçersiz kılarak genişletme isteğini yineler.

Bir veri kümesi genişletildiğinde, yeni veri kümesi kapsamları genişletme işleminin bir parçası olarak biçimlendirilir; bu on saniye ya da çok büyük kapsamlar için dakikalar sürebilir. Yeni alan, biçimlendirme tamamlandıktan ve katalog, yeni yüksek kullanılan denetim aralığını gösterecek şekilde güncellendikten sonra kullanılabilir olur.

Yeni iletiler çok hızlı bir şekilde oluşturulursa, genişletme işlemi tamamlanmadan önce var olan veri kümesinin dolması mümkündür. Bu durumda, alan ayıramayan tüm istekler, genişletme girişimi tamamlanıncaya ve yeni alan kullanılabilir oluncaya kadar geçici olarak askıya alınır. Genişletme başarılı olursa, istek otomatik olarak yeniden denenir.

Kullanılabilir alan olmaması ya da kapsam üst sınırına ulaşılmış olması nedeniyle bir genişletme girişimi başarısız olursa, hatanın nedenini bildiren bir ileti yayınlanır ve etkilenen SMDS için geçersiz kılma seçeneği, ek genişletme girişimlerini önlemek için otomatik olarak **DSEXPAND(NO)** olarak değiştirilir. Bu durumda, veri kümesinin dolması riski vardır; bu durumda, Veri kümesi dolu olurbaşlıklı konuda açıklandığı gibi daha fazla işlem gerekebilir.

Uygulama yapısı kullanımının izlenmesi

Uygulama yapısının kullanım düzeyi, uygulama yapısının tam adını (kuyruk paylaşım grubu öneki de içinde olmak üzere) belirten MVS **DISPLAY XCF,STRUCTURE** komutu kullanılarak görüntülenebilir. IXC360I yanıt iletisi, öğelerin ve girişlerin yürürlükteki kullanımını gösterir.

Yapı kullanımı CFRM ilkesinde belirtilen **FULLTHRESHOLD** değerini aştığında, sistem IXC585E iletisini yayınlar ve belirtildiyse otomatik **ALTER** işlemleri gerçekleştirebilir; bu, girdiyi öğe oranına değiştirebilir ya da yapı boyutunu artırabilir.

Arabellek havuzu büyüklüklerini eniyileme

Paylaşılan arabellek havuzundaki her arabellek, mantıksal blok büyüklüğüne kadar olan bir ileti için bitişik bir sayfa aralığını okumak ya da yazmak için kullanılır. İleti başka bloklara dökülecekse, ayrı bir bloktaki her sayfa aralığı ayrı bir arabellek gerektirir.

Bir yazma ya da okuma işleminden sonra ileti verilerini içeren arabellekler depoda tutulur ve en son kullanılan (LRU) önbellek şeması kullanılarak yeniden kullanılır; böylece, kısa bir süre sonra aynı verileri yeniden okuma isteği diske gitmez. Bu, paylaşılan iletiler yazıldığında ve daha sonra aynı sistemde çalışan uygulamalar tarafından okunduğunda önemli bir optimizasyon sağlar. Başka bir kuyruk yöneticisinin iyeliğindeki iletilere seçim amacıyla göz atılırsa, bu işlem iletiyi diskten yeniden okuma gereksinmesini de ortadan kaldırır.

Bu, her uygulama yapısı için gereken arabelleklerin sayısının, o uygulama yapısına ilişkin büyük iletileri okuyan ya da yazan her eşzamanlı API isteği için bir tane olduğu ve sonraki okuma erişimlerini optimize etmek için en son erişilen verileri kaydetmek üzere kullanılacak bazı ek arabellekler olduğu anlamına gelir.

Paylaşılan arabellek havuzları için yeterli arabellek yoksa, arabellek hemen kullanılamıyorsa API istekleri bekleyecek. Ancak, performansın önemli ölçüde düşmesine neden olacağından bu durum önlenmelidir.

Paylaşılan arabellek havuzlarına ilişkin **DISPLAY USAGE** komutundaki istatistikler, yürürlükteki istatistik aralığı içinde herhangi bir arabellek bekleme süresi olup olmadığını ve ayrıca en düşük boş arabellek sayısını (ya da herhangi bir zamanda arabellek bekleyen iş parçacığı sayısı üst sınırını gösteren negatif bir değeri), verileri saklayan arabelleklerin sayısını ve bir arabellek isteğinin LRU zincirinde kayıtlı verileri başarıyla bulup bulmadığını ("LRU eşleşmeleri") gösterir. okumak zorunda kalmaktansa ("LRU ıskalanır")[1].

- Herhangi bir bekleme varsa, arabelleklerin sayısı artırılmalıdır.
- Kullanılmayan çok sayıda arabellek varsa, bölgede başka amaçlarla daha fazla depolama alanı sağlamak için arabellek sayısı azaltılabilir.
- Kaydedilen verileri içeren çok sayıda arabellek varsa, ancak bu kaydedilen verilere karşı rastlanan okuma oranı çok azsa, depolama başka amaçlar için daha iyi kullanılacaksa arabellek sayısı azaltılabilir. Bununla birlikte, arabellek sayısı en düşük boş arabellek sayısından daha fazla azaltılmamalıdır, çünkü bu bekleme süresini tetikleyebilir ve tercihen en en düşük serbest arabellek sayısının normalde sıfırın üzerinde olması için yeterince yüksek olmalıdır.

Paylaşılan ileti veri kümelerini silme

DELETE CFSTRUCT komutu (yalnızca yapıdaki tüm paylaşılan kuyruklar boş ve kapalıyken kullanılabilir), paylaşılan ileti veri kümelerini kendileri silmez, ancak bu komut tamamlandıktan sonra olağan şekilde silinebilir. Aynı veri kümesi paylaşılan ileti veri kümesi olarak yeniden kullanılacaksa, boş duruma sıfırlamak için önce yeniden biçimlendirilmesi gerekir.

Paylaşılan ileti veri kümeleri için kural dışı durum durumları

Herhangi bir yazılım ya da donanım hatası olmasa da, normal kullanım sırasında ortaya çıkabilecek bazı özel durumlar vardır.

[1] `(Hits / (Hits+Misses))* 100`

Veri kümesi dolu olur

Bir veri kümesi dolursa, ancak genişletilemezse ya da genişletme girişimi başarısız olursa, ilgili uygulama yapısına büyük iletiler yazmak için ilgili kuyruk yöneticisini kullanan uygulamalar 2192, MQRC_STORAGE_MEDIUM_FULL (MQRC_PAGESET_FULL olarak da bilinir) hatasını alır.

Bir veri kümesi, verileri işlemesi beklenen ve büyük bir birikim birikmesine neden olması gereken uygulamadaki bir hata nedeniyle dolu olabilir. Bu durumda, veri kümesinin daha da genişletilmesi yalnızca geçici bir çözüm olacaktır ve işleme uygulamasının mümkün olan en kısa sürede yeniden işlemesi önemlidir.

Daha fazla yer kullanılabiliyorsa, **ALTER SMDS** komutu **DSEXPAND(YES)** ya da **DSEXPAND(DEFAULT)** ayarını tanımlamak için kullanılabilir (YES değerinin CFSTRUCT tanımına ilişkin **DSEXPAND** varsayılan değeri olarak belirlendiği ya da kabul edildiği varsayılarak). Hatanın nedeni kapsam sayısı üst sınırına ulaşıldıysa, yeni genişletme girişimi bir iletiyle reddedilir ve **DSEXPAND(NO)** yeniden ayarlanır. Bu durumda, daha fazla genişletmenin tek yolu, daha sonra açıklandığı gibi geçici olarak kullanılamaz hale getirmeyi içeren yeniden ayırmaktır.

Veri kümesinin taşınması ya da yeniden tahsis edilmesi gerekiyor

Bir veri kümesinin taşınması ya da genişletilmesi gerekiyorsa, ancak normal kullanımda değilse, taşınmasına ya da yeniden yerleştirilmesine izin vermek için geçici olarak kullanımdan kaldırılabilir. Veri kümesini kullanılamadığında kullanmayı deneyen her API isteği MQRC_DATA_SET_NOT_UNAVAILABLE neden kodunu alır.

1. Veri kümesini **ACCESS(DISABLED)** olarak işaretlemek için **RESET SMDS** komutunu kullanın. Bu, olağan bir şekilde kapatılmasına ve bağlı tüm kuyruk yöneticileri tarafından serbest bırakılmasına neden olur.
2. Eski içeriği yeni ayrılan veri kümesine (örneğin, Erişim Yöntemi Hizmetleri (AMS) **REPRO** komutunu kullanarak) kopyalayarak, veri kümesini gerektiği gibi taşıyın ya da yeniden ayırın.

   Eski verileri kopyalamadan önce yeni veri kümesini önceden biçimleme girişiminde bulunmayın; bu, kopyalanan verilerin biçimlendirilmiş veri kümesinin sonuna eklenmesine neden olur.
3. Veri kümesini yeniden **ACCESS(ENABLED)** olarak işaretlemek ve kullanıma geri getirmek için **RESET SMDS** komutunu kullanın.

   Eski içerik, yeni veri kümesinin boyutundan küçükse, yeni veri kümesi açıldığında alanın geri kalanı otomatik olarak önceden biçimlendirilir.

   Eski içerik, yeni veri kümesinin boyutundan büyükse, kuyruk yöneticisinin bağlaşım olanağı yapısındaki iletileri taraması ve etkin verilerin kaybolmadığından emin olmak için alan eşlemini yeniden oluşturması gerekir. Yeni kapsamların dışındaki bir veri bloğuna herhangi bir başvuru bulunursa, veri kümesi **STATUS(FAILED)** olarak işaretlenir ve veri kümesi doğru boyuttan biriyle değiştirilerek ve eski veri kümesi yeniden kopyalanarak ya da kalıcı iletileri kurtarmak için **RECOVER CFSTRUCT** kullanılarak onarılmalıdır.

Bağlaşım olanağı yapısı alan azaldı

Bağlaşım olanağı yapısında alan tükeniyorsa ve IXC585Eiletisine neden oluyorsa, bu durumda veri miktarı üst sınırının boşaltıldığından emin olmak için boşaltma kurallarının ayarlanıp ayarlanmadığını denetlemeye değer. Değilse, boşaltma kuralları **ALTER CFSTRUCT** komutu kullanılarak değiştirilebilir.

Paylaşılan ileti veri kümeleri için hata durumları

Dikkat edilmesi gereken bazı sorunlar vardır; bu sorunlar yalnızca hatalardan kaynaklanabilir ve olağan çalışma durumlarında ortaya çıkmaz.

Sahip olunan veri kümesi açılamıyor

Paylaşılan ileti veri kümesine sahip kuyruk yöneticisi bunu ayıramazsa ya da açamazsa ya da veri kümesi öznitelikleri desteklenmiyorsa, kuyruk yöneticisi uygun **SMDSCONN** durum değerini **ALLOCFAIL** ya da **OPENFAIL** olarak ayarlar ve **SMDSCONN** kullanılabilirliğini **AVAIL(ERROR)** olarak

ayarlar. Ayrıca SMDS kullanılabilirliğini **ACCESS(SUSPENDED)** olarak ayarlar. Hata düzeltildiğinde, bir yeniden denemeyi tetiklemek için **ACCESS(ENABLED)** ayarını tanımlamak için **RESET SMDS** komutunu kullanın ya da sahibi olan kuyruk yöneticisine **START SMDSCONN** komutunu verin.

Salt okunur veri kümesi açılamıyor

Bir kuyruk yöneticisi, başka bir kuyruk yöneticisinin sahip olduğu ve **STATUS(ACTIVE)** olarak işaretlenmiş bir paylaşılan ileti veri kümesini ayıramazsa ya da açamazsa, bunun büyük olasılıkla veri kümesinin kendisiyle ilgili bir sorun değil, veri kümesiyle (**SMDSCONN** nesnesi tarafından temsil edilen) bağlantısıyla ilgili bir sorun nedeniyle olduğunu varsayar.

SMDSCONN STATUS(ALLOCFAIL) ya da **STATUS(OPENFAIL)** öğesini uygun olarak işaretler ve **SMDSCONN** kullanılabilirliğini AVAIL (ERROR) olarak işaretler.

Sorun, veri kümesinin durumunu etkilemeden düzeltilebiliyorsa, yeniden denemeyi tetiklemek için **START SMDSCONN** komutunu kullanın.

Sorun veri kümesinin kendisiyle ilgili bir sorun olduğu ortaya çıkarsa, kurtarılana kadar veri kümesini **STATUS(FAILED)** olarak işaretlemek için **RESET SMDS** komutu kullanılabilir. Veri kümesi kurtarıldığında, durumu **STATUS(ACTIVE)** olarak değiştirme işlemi, diğer kuyruk yöneticilerine bildirim gönderilmesine neden olur. **SMDSCONN** , **AVAIL(ERROR)** olarak işaretliyse, veri kümesini açmak için yeni bir girişimi tetiklemek üzere otomatik olarak **AVAIL(NORMAL)** olarak yeniden değiştirilir.

Veri kümesi üstbilgisi bozuk

Veri kümesi başarıyla açıldıysa, ancak üstbilgi bilgilerinin biçimi yanlışsa, kuyruk yöneticisi veri kümesini kapatır ve serbest bırakır ve durum kümesini **STATUS(FAILED)** ve kullanılabilirliği **ACCESS(SUSPENDED)** olarak ayarlar. Bu, **RECOVER CFSTRUCT** ' in içeriği kurtarmak için kullanılmasını sağlar.

Hata, veri kümesinin başka bir kullanımdan kalan veriler içermesi ve daha sonra önceden biçimlendirilmemiş olması nedeniyle ortaya çıkarsa, veri kümesini önceden biçimlendirin ve durumu **STATUS(RECOVERED)** olarak değiştirmek için **RESET SMDS** komutunu kullanın.

Tersi durumda, veri kümesinin kurtarılması gerekir.

Veri kümesi beklenmeyen bir şekilde boş

Kuyruk yöneticisi **STATUS(ACTIVE)** olarak işaretli bir veri kümesini açarsa, ancak başlatılmamış ya da yeni biçimlendirilmiş ancak başka bir şekilde geçerli olduğunu bulursa, kuyruk yöneticisi, paylaşılan ileti veri kümesini kapatır ve serbest bıraktıktan sonra durumu **STATUS(FAILED)** ve kullanılabilirliği **ACCESS(SUSPENDED)** olarak ayarlar.

Veri kümesinde kalıcı G/Ç hataları var

OPEN işlemi başarılı olduktan sonra bir veri kümesinde kalıcı G/Ç hataları varsa, büyük olasılıkla kurtarma işlemi gerekir. Kuyruk yöneticisi, veri kümesini **STATUS(FAILED)** olarak işaretler; böylece, bağlı olan tüm kuyruk yöneticileri bu veri kümesini kapatır ve serbest bırakır.

Veri kümesinde kurtarılabilir G/Ç hataları var

Veri kümesiyle ilgili donanım sorunları varsa, bu, kuyruk yöneticisine geri yansıtılmayan, ancak başarımın önemli ölçüde düşmesine neden olan ve yakın gelecekte kalıcı G/Ç hataları riskini de gösteren kurtarılabilir G/Ç hatalarıyla sonuçlanabilir.

Bu durumda, veri kümesi, **STATUS(FAILED)** olarak işaretlemek için **RESET SMDS** komutu kullanılarak kurtarma için çevrimdışı duruma getirilir. Bu, tüm kuyruk yöneticileri tarafından kapatılmasına ve serbest bırakılmasına neden olur; bu nedenle, yeniden kullanılabilir kılınmadan önce yeni bir birime taşınabilir.

Bir veri kümesi bu şekilde kullanılamaz kılındığında, alan eşlemi saklanmaz; bu nedenle, kuyruk yöneticisi bağlantısını yeniden başlatma işleminin veri kümesindeki iletileri bulmak için bağlaşım olanağı yapısını taraması ve veri kümesinin yeniden kullanılabilir kılınması için alan eşlemini yeniden oluşturması gerekir. Diğer bir seçenek olarak, paylaşılan ileti veri kümesi hala kullanılabiliyorsa,

ACCESS(DISABLED) veri kümesini işaretlemek için **RESET SMDS** komutu kullanılarak, yeniden kullanılabilir kılınmaya hazır oluncaya kadar, bu küme daha nazik bir şekilde kullanılamaz kılınabilir.

Veri kümesi içeriği yanlış

Kuyruk yöneticisi, bir veri kümesinin yanlış veri içerdiğini ya da güncel olmadığını doğrudan saptayamıyor; örneğin, bu veri kümesini içeren bir birimin yedeklerden geri yüklenmesi gerektiğinden. Ancak, bu tür hataların uygulama programları tarafından yanlış ileti verilerinin görülmesine neden olma olasılığını çok düşük yapan bütünlük denetimleri gerçekleştirir.

Bütünlük denetimi amacıyla, veri kümesindeki her ileti bloğunun başında, ileti verileri kullanıcı programına geçirilmeden önce, ileti bloğu her okunduğunda denetlenen benzersiz bir zaman damgası da içinde olmak üzere, ilgili bağlaşım olanağı giriş tanıtıcısının bir kopyası bulunur. İleti öbeği öneki giriş tanıtıcısıyla eşleşmezse (ve bağlaşım olanağı girişi ortalama süre içinde silinmediyse), ileti öbeğinin zarar gördüğü ve kullanılamadığı varsayılır.

Hasarlı ileti kalıcıysa, veri kümesi **STATUS(FAILED)** olarak işaretlenir ve yapı içeriğinin **RECOVER CFSTRUCT** komutu kullanılarak kurtarılması gerekir. Hasarlı ileti kalıcı değilse, iletiyi kurtarmanın bir yolu yoktur; bu nedenle bir tanılama iletisi yayınlanır ve ilgili bağlaşım olanağı ileti girişi silinir.

Veri kümesi açıldığında kaydedilen alan eşlemi yoksa, veri kümesindeki verilere başvurular için bağlaşım olanağı yapısı taranarak yeniden oluşturulur. Bu tarama sırasında, kuyruk yöneticisi birkaç işlem gerçekleştirir:

1. Kuyruk yöneticisi, veri kümesinde kalan en son iletinin (varsa) yerini belirler.
2. Kuyruk yöneticisi, blok önekinin ileti girişi tanıtıcısıyla eşleştiğinden emin olmak için veri kümesinden bu iletiyi okur

Bu işlemler, kuyruk yöneticisinin veri kümesinin alt düzey olduğu herhangi bir durumu saptamasını ve veri kümesini FAILED olarak işaretlemesini sağlar. Ancak bu denetim, veri kümesinin önceki bir kopyadan geri yüklendiği ve o zamandan bu yana yeni ileti eklenmediği ya da o kopyanın daha sonra okunup silinmesinden bu yana eklenen tüm iletilerin eklenmediği durumu tolere eder.

Veri kümesinin olağan şekilde kapatıldığı durumlarda, düşük düzeyli verilere karşı koruma sağlamak için kuyruk yöneticisi bir dizi işlem gerçekleştirir:

1. Kuyruk yöneticisi, veri kümesi olağan bir şekilde kapatıldığında Db2 içindeki SMDS nesnesinde alan eşlemi zaman damgasının bir kopyasını kaydeder.
2. Kuyruk yöneticisi, veri kümesi yeniden açıldığında, alan eşlemi zaman damgasının aynı olup olmadığını denetler.

Zaman damgası eşleşmezse, bu, veri kümesinin alt düzey bir kopyasının kullanılmış olabileceği anlamına gelir; bu nedenle, kuyruk yöneticisi var olan alan eşlemini yoksayar ve yeniden oluşturur; bu da, hiçbir ileti verilerinin gerçekten kaybolmaması durumunda başarılı olur.

Not: Bu bütünlük kontrolleri, teorik olarak olası tüm durumlarda düşük seviyeli ya da hasarlı bir veri kümesini algılamayı garanti etmez. Örneğin, bir ileti öbeğinin başlangıcını geçerli olduğu, ancak verilerin geri kalanının kısmen üzerine yazıldığı bir durum algılanmaz.

Paylaşılan ileti veri kümeleri için kurtarma senaryoları

Bu bölümde, paylaşılan ileti veri kümesi kurtarma senaryoları açıklanmaktadır.

Veri kaybolmadığı durumlarda veri kümesi kurtarma

Bazı durumlarda, başarısız olan bir veri kümesinin doğru içeriği, gerçek kurtarma gerekmeden geri yüklenebilir. Bir örnek, bir veri kümesinin önceki bir kullanımdan kalan verileri içermesi ve yeniden önceden biçimlendirilmemiş olması ve bu verilerin önceden biçimlendirilmesiyle düzeltilebilmesi olabilir. Başka bir durum da, bir veri kümesi taşındığında, ancak verilerin kopyalanması sırasında bir hata oluştuğunda ortaya çıktı; bu hata, veriler yeniden doğru şekilde kopyalanarak düzeltilebilir.

Bu tür durumlarda, düzeltilen veri kümesi, **STATUS(RECOVERED)** değerini ayarlamak için **RESET SMDS** komutu kullanılarak yeniden kullanılabilir duruma getirilebilir. Kullanılabilirlik şu anda **ACCESS(SUSPENDED)** ise, otomatik olarak **ACCESS(ENABLED)** değerine ayarlanır.

Sahip olan kuyruk yöneticisine veri kümesinin kurtarıldığı bildirildiğinde, alan haritasını yeniden oluşturmak için yapı içeriğini tarar ve durumu **STATUS(ACTIVE)** olarak değiştirir. Diğer kuyruk yöneticileri daha sonra veri kümesini yeniden okumaya başlayabilir.

TYPE ile veri kümesi kurtarma (NORMAL)

Bir veri kümesinin içeriği kaybolduysa, ancak uygulama yapısı **RECOVER(YES)** ile tanımlandıysa ve uygun kurtarma günlükleri kullanılabiliyorsa, **RECOVER CFSTRUCT** komutu, paylaşılan ileti veri kümelerine boşaltılan kalıcı ileti verileri de dahil olmak üzere yapıda saklanan tüm kalıcı iletileri kurtarmak için kullanılabilir. Bu komut, **BACKUP CFSTRUCT** komutu tarafından günlüğe kaydedilen bilgilerin yanı sıra yedekleme zamanından bu yana kalıcı iletilerde yapılan tüm değişiklikleri kullanarak geçerli durumu geri yükler.

RECOVER CFSTRUCT komutu, Db2içinde depolanan boşaltılan ileti verileriyle birlikte bağlaşım olanağı yapısındaki tüm kalıcı iletileri her zaman kurtarır. Paylaşılan ileti veri kümelerinde depolanan boşaltılmış veriler için, her veri kümesi yalnızca **STATUS(FAILED)** olarak işaretlenmişse ya da kurtarma işlemi tarafından açıldığında beklenmeyen bir şekilde boş olduğu ya da başka bir şekilde geçersiz olduğu bulunursa kurtarma işlemi için seçilir. Var olan ileti verileri zaten doğru olduğundan, etkin olarak işaretlenen ve doğrulama denetimlerini geçen paylaşılan ileti verileri kümesinin kurtarılması gerekmez, ancak üstbilgi, kaydedilen alan eşlemlerinin kurtarıldıktan sonra yeniden oluşturulması gerektiğini gösterecek şekilde güncellenir.

Kurtarma işlemi, yapının tüm içeriğinin kurtarma işlemiyle yeniden oluşturulmasının gerekmesi nedeniyle, yapı başarısız olarak işaretlendiğinde gerçekleştirilebilir. Ancak, en az bir paylaşılan ileti veri kümesi başarısız olarak işaretlendiyse **RECOVER CFSTRUCT** komutu, kurtarma işleminin devam etmesine izin vermek için yapıyı otomatik olarak başarısız olarak işaretler.

Kurtarma, ilgili veri kümelerine yazma erişimi verilmiş olması koşuluyla, kuyruk paylaşım grubundaki herhangi bir kuyruk yöneticisinden gerçekleştirilebilir.

Yalnızca kalıcı iletiler yedeklenir ve günlüğe kaydedilir; bu nedenle, olağan kurtarma işlemi tüm kalıcı iletileri geri yükler, ancak yapıdaki kalıcı olmayan iletilerin kaybolmasına neden olur.

Kurtarma tamamlandığında, kurtarma için seçilen herhangi bir veri kümesi otomatik olarak **STATUS(RECOVERED)** olarak değiştirilir ve kullanılabilirlik **ACCESS(SUSPENDED)** ise, **ACCESS(ENABLED)** olarak değiştirilir. Kuyruk yöneticisi, bağlaşım olanağındaki iletileri tarayarak her veri kümesi için alan haritasını yeniden oluşturur ve daha sonra, yeniden kullanılabilmesi için veri kümesini **STATUS(ACTIVE)** olarak işaretler.

TYPE (PURGE) ile veri kümesi kurtarma

Kurtarılabilir bir yapıda, veri kümesi içeriği kaybolduysa, ancak bir nedenden ötürü kurtarma mümkün değilse, örneğin kurtarma günlükleri kullanılamadığından ya da kurtarma çok uzun süreceği için, **RECOVER CFSTRUCT** komutu **TYPE(PURGE)** ile yapıyı yeniden kullanılabilir duruma getirmek için kullanılabilir. Bu, yapıyı boş duruma getirir ve ilişkili tüm veri kümelerini **STATUS(EMPTY)** olarak işaretler.

Uygulama yapısı siliniyor

Kurtarılamayan bir uygulama yapısı MVS **SETXCF FORCE** komutu kullanılarak ya da yapı arızası sonucu silinirse, yapının bir sonraki bağlanması durumunda, yapının ilk durumuna getirildiğini ve var olan tüm iletilerin atılacağını ve var olan veri kümelerinin de otomatik olarak **STATUS(EMPTY)** olarak sıfırlandığını bildiren CSQE028I iletisi yayınlanır. Bu işlem, yapıda ya da ilişkili veri kümelerinden herhangi birinde veri kaybından sonra kurtarılamayan bir yapıyı yeniden kullanılabilir hale getirir.

Kurtarılabilir bir uygulama yapısı silinirse, yapı başarısız olmuş gibi işlem görür.

Veri kümesi kurtarma başarısız oldu

RECOVER CFSTRUCT herhangi bir nedenle tamamlanamazsa (örneğin, bir günlük veri kümesi artık kullanılamadığı için ya da kurtarma işlemi devam ederken kuyruk yöneticisi sonlandırıldığı için), kurtarma işleminin en az başlatıldığı veri kümesi, üstbilgide kısmi kurtarma girişiminde bulunulduğu ve veri kümesinin **STATUS(FAILED)** durumunda bırakılacağı şekilde işaretlenir.

Bu durumda, seçenekler özgün kurtarma isteğini yinelemek ya da var olan verileri atarak **TYPE(PURGE)** ile kurtarmak için kullanılabilir.

Veri kümesini kurtarmadan **STATUS(RECOVERED)** olarak işaretleme girişiminde bulunulursa, kuyruk yöneticisi bir sonraki açılışında üstbilginin eksik kurtarma gösterdiğini görür ve **STATUS(FAILED)** olarak yeniden işaretler.

Tesis dışında olağanüstü durumdan kurtarma

Müşteri yeri dışında olağanüstü durumdan kurtarma için, kalıcı paylaşılan iletiler yalnızca günlükler ve CFSTRUCT tanımlarını ve ilişkili SMDS durum bilgilerini içeren Db2 paylaşılan nesneleri kullanılarak yeniden oluşturulabilir.

Tanımlamaları içeren Db2 çizelgeleri ayarlandıktan sonra, uygulama yapısı ve paylaşılan ileti veri kümeleri boş olarak ayarlanabilir. Bir kuyruk yöneticisi bunlara bağlandığında ve beklenmedik bir şekilde boş olduklarını öğrendiğinde, bunları başarısız olarak işaretler ve bundan sonra etkilenen tüm yapılara ilişkin tüm kalıcı iletileri kurtarmak için tek bir **RECOVER CFSTRUCT** komutu kullanılabilir.

z/OS *SMDS ile ilgili komutlar*

Bu konuda, paylaşılan ileti veri kümeleriyle ilgili komutlara erişim sağlar ve bu komutlara erişim sağlar.

Büyük ileti boşaltma (**OFFLOAD** ve offload rules) ve paylaşılan ileti veri kümeleri ile ilgili **CFSTRUCT** seçeneklerini görüntüleyin ve değiştirin (**DSGROUP, DSBLOCK, DSBUFS, DSEXPAND)**:

- CFSTRUCT GöRüNTüSü
- CFSTRUCT DEFINE
- ALTER CFSTRUCT
- CFSTRUCT SIL

Büyük ileti boşaltma **(OFFLDUSE)** ile ilgili **CFSTRUCT** durumunu görüntüle:

- CFDURU GöRüNTüLE

Geçersiz kılma veri kümesi seçeneklerini görüntüle ve değiştir (**DSEXPAND** ve **DSBUFS**) Tek tek kuyruk yöneticileri için:

- SMDS GöRüNTüLE
- ALTER SMDS

Kuyruk paylaşım grubu içindeki veri kümelerinin durumunu ve kullanılabilirliğini görüntüler ya da değiştirir:

- CFSTATUS TYPE (SMDS) GöRüNTüLE
- SMDS ' YI YENIDEN

Bir kuyruk yöneticisine ilişkin SMDS veri kümesi alanı kullanımı ve arabellek kullanım bilgilerini görüntüler:

- GöRüNTüLEME TIPI (SMDS)

Bağlantıların durumunu ve kullanılabilirliğini görüntüle ya da değiştir (**SMDSCONN**) Tek bir kuyruk yöneticisinden veri kümelerine:

- SMDSCONN GöRüNTüLEYIN
- SMDSCONN BAşLı
- SDSCONN ' ı DURDUR

Gerekli olduğunda SMDS ' de büyük ileti verileri de dahil olmak üzere paylaşılan iletileri yedekleyin ve kurtarın:

- BACKUP CFSTRUCT
- CFSTRUCT ' I KURTAR

z/OS *Paylaşılan kuyruklar kullanmanın avantajları*

Paylaşılan kuyruk, IBM MQ uygulamalarının ölçeklenebilir, yüksek düzeyde kullanılabilir olmasını ve iş yükü dengelemenin gerçekleştirilmesini sağlar.

Paylaşılan kuyrukların yararları

Eşkopyalanmış sunucuların tek bir paylaşılan kuyruktan iş çevirdiği paylaşılan kuyruk mimarisi bazı yararlı özelliklere sahiptir:

- Sunucu uygulamasının yeni örnekleri eklenerek ya da kuyruk yöneticisiyle (kuyruk paylaşım grubunda) yeni bir z/OS görüntüsü ve uygulamanın bir kopyası eklenerek, bu ölçeklenir.
- Oldukça uygun.
- Bu, doğal olarak kuyruk paylaşım grubundaki her kuyruk yöneticisinin kullanılabilir işleme kapasitesine dayalı olarak *çekme* iş yükü dengelemesini gerçekleştirir.

Yüksek kullanılabilirlik için paylaşılan kuyruklar kullanma

Aşağıdaki örneklerde, uygulama kullanılabilirliğini artırmak için paylaşılan bir kuyruğu nasıl kullanabileceğiniz gösterilmektedir.

Ağ üzerinde çalışan istemci uygulamalarının, z/OSüzerinde çalışan sunucu uygulamaları isteklerini yapmak istediği bir IBM MQ senaryosını göz önünde bulundurun. İstemci uygulaması bir istek iletisi oluşturur ve bunu bir istek kuyruğuna yerleştirir. İstemci daha sonra, istek iletisinin ileti tanımlayıcısında belirtilen yanıtlama kuyruğuna gönderilen sunucudan yanıt bekliyor.

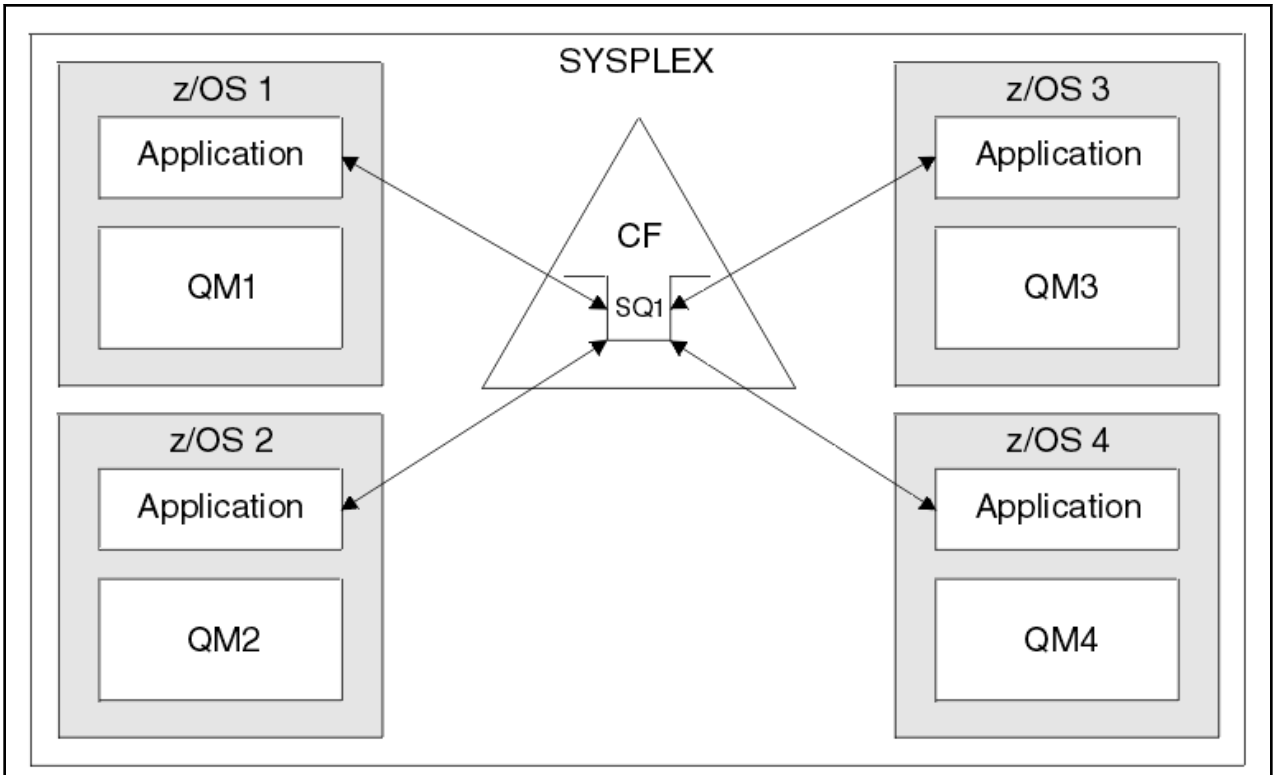
IBM MQ , istek iletisinin istemci makinesinden sunucunun z/OS üzerindeki giriş kuyruğuna ve sunucudan istemciye geri gönderilmesine ilişkin ulaşımı yönetir. Sunucunun giriş kuyruğunu paylaşılan bir kuyruk olarak tanımlayarak, kuyruğa gönderilen tüm iletiler, kuyruk paylaşım grubundaki herhangi bir kuyruk yöneticisinde alınabilir. Bu, sysplex içindeki her bir z/OS görüntüsünde bir kuyruk yöneticisini yapılandırabileceğiniz ve bunların tümünü aynı kuyruk paylaşım grubuna bağlayarak, bunların herhangi birinin sunucunun giriş kuyruğunda iletilere erişebildiği anlamına gelir.

Sunucunun giriş kuyruğunda bulunan iletiler, kuyruk yöneticilerinden biri olağandışı şekilde sona ererse ya da denetim nedeniyle bunu durdurmanız gerekse de, ileti kuyruğunda kullanılabilir. z/OS görüntüsünün tamamını çevrimdışı duruma getirebilirsiniz ve iletiler yine de kullanılabilir olacaktır.

To take advantage of this availability of messages on a shared queue, run an instance of the server application on each z/OS image in the sysplex to provide higher server application capacity and availability, as shown in Şekil 61 sayfa 181.

Sunucu uygulamasının bir eşgörünümü, paylaşılan kuyruktan bir istek iletisi alır ve içeriğe dayalı olarak, istemciye bir IBM MQ iletisi olarak geri gönderilen bir sonuç üreterek, bu iletiyi işlemeyi gerçekleştirir. Yanıt iletisi, istek iletisinin ileti tanımlayıcısında ve yanıt kuyruğunda ve istek iletisinin ileti tanımlayıcısında belirtilen yanıttan kuyruğa gönderilir.

Dönüş yolunu yapılandırmak için kullanabileceğiniz bir dizi seçenek vardır. Bu seçenekler hakkında daha fazla bilgi için bkz. “Dağıtılmış kuyruğa alma ve kuyruk paylaşım grupları” sayfa 199.



Şekil 61. Bir uygulamanın paylaşılan bir kuyruğa hizmet vermesinin birden çok eşgörünümü

Eş kurtarma

Bir kuyruk paylaşım grubundaki iletilerin kullanılabilirliğini daha da artırmak için, IBM MQ , gruptaki başka bir kuyruk yöneticisinin bağlaşım tesisinden bağlantıyı kesip bağlanmadığına karar verdi ve halen beklemekte olan kuyruk yöneticisine ilişkin çalışma birimlerini tamamlayıp tamamlamayı tamamlar. Bu özellik *eş kurtarma*olarak bilinir.

Bir kuyruk yöneticisinin, bir uygulamanın bir istek iletisini bir kuyruktan eşitleme noktasında aldığı, ancak henüz yanıt iletisini koymadığı ya da iş birimini kesinleştirmediği bir noktada olağan dışı bir şekilde sona erdirdiğini varsayalım. Kuyruk paylaşım grubundaki başka bir kuyruk yöneticisi, hatayı algılar ve başarısız olan kuyruk yöneticisinde gerçekleştirilmekte olan iş içi iş birimlerinden yedekler. Bu, istek iletisinin istek kuyruğuna geri konacağı ve diğer sunucu yönetim ortamlarından biri için, başarısız olan kuyruk yöneticisinin yeniden başlatılması beklenmeden, bu iletinin işleneceği anlamına gelir.

IBM MQ bir iş birimini otomatik olarak çözemiyorsa, kuyruk paylaşım grubunda başka bir kuyruk yöneticisinin çalışmayı sürdürmesini sağlamak için, paylaşılan bölümü el ile çözebilirsiniz.

z/OS *Paylaşılan kuyruklar ile depolama sınıfı belleğinin kullanılması*

IBM MQ for z/OS paylaşılan kuyruklarıyla kullanıldığında SCM (storage class memory; depolama sınıfı belleği) kullanımı avantajlı olabilir.

Önemli: IBM z16 , Coupling Facility görüntüleri için Virtual Flash Bellek (Depolama Sınıfı Bellek ya da SCM olarak da bilinir) kullanımını desteklemek üzere son nesil IBM Z ® olması planlanmıştır. Daha fazla bilgi için bkz. IBM Z ve IBM LinuxONE 4Q 2023 Yönelimin Bildirimleri.

Alternatif olarak, daha büyük yapılar kullanmalı ya da iletileri SMDS ' ye boşaltmalısınız.

z13, zEC12ve zBC12 makineleri, Flash Express kartlarının kuruluşuna izin verir. Bu kartlar, flaş yarıiletken sürücüler (SSD) içerir. Kuruluştan sonra kartlardaki flaş depolama, genellikle SCM olarak bilinen bir veya daha fazla LPAR ' a ayrılabilir.

SCM, G/Ç gecikme süresi ve maliyeti açısından gerçek depolama ile doğrudan erişim depolama aygıtı (DASD) arasında bulunur. SCM 'nin hareketli parçaları olmadığından, DASD' den çok daha düşük G/Ç gecikme süreleri gösterir.

SCM ayrıca gerçek depolamadan çok daha ucuzdur. Sonuç olarak, nispeten düşük bir maliyetle büyük miktarda depolama kurulabilir; örneğin, bir çift Flash Express kartı 1424 GB kullanılabilir depolama alanı içerir.

Bu özellikler, verilerin SCM 'ye DASD' ye yazılabildiğinden çok daha hızlı yazılabilmesi nedeniyle, SCM ' nin kısa bir süre içinde gerçek depolamadan alınması gerektiği zaman yararlı olacağı anlamına gelir. Bu özel nokta, IBM MQ paylaşılan kuyruklarını içeren bağlaşım olanağı (CF) liste yapıları kullanılırken çok yararlı olabilir.

Liste yapıları neden doluyor?

Bir CF yapısı tanımlandığında, yapının büyüklük üst sınırını tanımlayan bir SIZE özniteliğiyle yapılandırılır. CF yapıları her zaman gerçek depoda kalıcı olarak bulunduğundan, bir CF 'de tanımlanan yapıların SIZE özniteliklerinin toplamı, CF' ye ayrılan gerçek depolama miktarından az olmalıdır.

Sonuç olarak, daha fazla yapının CF ' ye sığması için herhangi bir yapının SIZE değerini mümkün olan en düşük değere kadar tutmak için sabit bir basınç vardır. Bununla birlikte, yapıların amaçlarına ulaşacak kadar büyük olmasını sağlamak çelişkili bir baskıyla sonuçlanabilir, çünkü bir yapının çok küçük olması, bu yapıyı kullanan uygulamaları veya altsistemleri bozabileceği anlamına gelir.

Beklenen kullanımına göre bir yapıyı doğru bir şekilde boyutlandırmak için güçlü bir ihtiyaç vardır. Ancak, iş yükleri zaman içinde değişebildiği ve dalgalanmalarının muhasebesi kolay olmadığı için bu görevi yapmak zordur.

IBM MQ paylaşılan kuyrukları, iletileri saklamak için CF liste yapılarını kullanır. IBM MQ , iletileri ve uygulama yapılarını içeren CF yapılarını çağırır.

Uygulama yapılarına, IBM MQ CFSTRUCT nesnelerinde saklanan bilgiler kullanılarak başvurulur. 63 KB ' den küçük bir ileti paylaşılan bir kuyruğa konduğunda, ileti tamamen bir uygulama yapısında tek bir liste girdisi ve sıfır ya da daha fazla liste öğesi olarak saklanır.

IBM MQ paylaşılan kuyrukları liste yapılarını kullandığından, açıklanan baskılar da paylaşılan kuyrukları etkiler. Bu durumda, paylaşılan bir kuyrukta saklanabilecek ileti sayısı üst sınırı, aşağıdaki işlevlerin bir işlevidir:

- Kuyruktaki iletilerin boyutu
- Yapının maksimum boyutu
- Yapıda kullanılabilir girdi ve öğe sayısı

512 'ye kadar paylaşılan kuyruk aynı yapıyı kullanabildiği ve girdiler ve öğeler için etkili bir şekilde rekabet edebildiği için, bu durum daha da karmaşık hale geliyor.

IBM MQ kuyrukları, uygulamalar arasında veri aktarımı için kullanılır; bu nedenle, ortak uygulama iletileri kuyruğa koyan bir uygulamadır ve bu iletileri almalıdır.

Bu durumda, aşağıdaki durumlardan biri ya da birkaçı ortaya çıkıncaya kadar kuyruktaki iletilerin sayısı zaman içinde artar:

- Koyma uygulaması ileti koymayı durdurur.
- Alan uygulama iletileri almaya başlar.
- Kuyrukta var olan iletilerin süresi doluyor ve kuyruktan kaldırılıyor.
- Kuyruk derinlik üst sınırına ulaşır; bu durumda, koyma uygulamasına bir MQRC_Q_FULL neden kodu döndürülür.
- Paylaşılan kuyruğu içeren yapı büyüklük üst sınırına ulaştı ya da yapıyı içeren CF ' nin kullanılabilir saklama alanı kalmadı. Her iki durumda da, koyma uygulamasına bir MQRC_STORAGE_MEDIUM_FULL neden kodu döndürülür.

Son üç durumda kuyruk dolu. Bu noktada, konan uygulamanın bir sorunu var çünkü iletilerinin gidecek yeri yok. Koyma uygulaması genellikle aşağıdaki çözümlerden birini ya da birkaçını kullanarak bu sorunu çözer:

- İsteğe bağlı olarak yeniden denemeler arasında bir gecikme ile iletiyi yerleştirmeyi yeniden deneyin.
- İletileri veritabanı ya da dosya gibi başka bir yere koyun. İletilere daha sonra erişilebilir ve kuyruğa normal şekilde yerleştirilebilir.
- Kalıcı değilse iletiyi atın.

Ancak, bazı uygulama sınıfları için, örneğin, büyük hacimli gelen iletilere sahip olanlar ya da bir dosya sistemine erişimi olmayanlar için, bu çözümler pratik değildir. Kuyrukların hiçbir zaman doldurulmamasını veya doldurulmamasını sağlamak için gerçek bir ihtiyaç vardır ve bu özellikle paylaşılan kuyruklar için uygun bir durumdur.

SMDS ve offload kuralları

IBM WebSphere MQ 7.1 içinde tanıtılan boşaltma kuralları, bir uygulama yapısının dolma olasılığını azaltmanın bir yolunu sağlar.

Her uygulama yapısıyla ilişkili, üç anahtar sözcük çifti kullanılarak belirtilen üç kural vardır:

- OFFLD1SZ ve OFFLD1TH
- OFFLD2SZ ve OFFLD2TH
- OFFLD3SZ ve OFFLD3TH

Her kural, ileti verilerinin uygulama yapısıyla ilişkili depolama mekanizmasına boşaltılması için karşılanması gereken koşulları belirtir. Şu anda iki tip depolama mekanizması vardır:

- Db2
- IBM MQ tarafından paylaşılan bir ileti veri kümesi (SMDS) çağıran Sanal Depolama Erişim Yöntemi (Virtual Storage Access Method; VSAM) doğrusal veri kümeleri grubu.

Aşağıdaki örnek, DEFINE CFSTRUCT komutunu kullanarak LIST1adlı bir uygulama yapısı yaratmak için kullanılan MQSC komutunu göstermektedir.

Bu yapı, varsayılan boşaltma kurallarını yerinde içerir ve boşaltma mekanizması olarak SMDS ' yi kullanır. Başka bir deyişle, yapı %70 doluyken (OFFLD1TH), 32 KB ya da daha büyük tüm iletiler (OFFLD1SZ) SMDS ' ye boşaltılır.

Benzer şekilde, yapı %80 doluyken (OFFLD2TH) 4 KB ya da daha büyük (OFFLD2SZ) tüm iletiler boşaltılır. Yapı %90 dolu olduğunda (OFFLD3TH) tüm iletiler (OFFLD3SZ) boşaltılır.

```
DEFINE CFSTRUCT(LIST1)
CFLEVEL(5)
OFFLOAD(SMDS)
OFFLD1SZ(32K) OFFLD1TH(70)
OFFLD2SZ(4K) OFFLD2TH(80)
OFFLD3SZ(0K) OFFLD3TH(90)
```

Boşaltılan bir ileti boşaltma ortamında saklanır ve iletiye ilişkin bir işaretçi yapıda saklanır. Boşaltma kuralları, depolama alanı tükenirken yapıya daha az ileti verisi koyarak yapının dolma olasılığını azaltırken, bazı veriler her ileti için yapıya yazılır. Yani, boşaltılan iletinin işaretçisi.

Ek olarak, boşaltma kuralları bir performans maliyeti ile birlikte gelir. Bir yapıya mesaj yazmak nispeten hızlıdır ve büyük ölçüde yazma isteğini CF ' ye göndermek için harcanan süre tarafından hakimdir. Yapıya gerçek yazılar hızlı, gerçek depolama hızlarında gerçekleşiyor.

SMDS 'ye bir ileti yazmak çok daha yavaştır, çünkü ileti işaretçisinin yapısına yazmayı ve ileti verilerini SMDS' ye yazmayı içerir. Bu ikinci yazma işlemi DASD hızında yapılır ve gecikme süresi ekleme potansiyeline sahiptir. Boşaltma mekanizması olarak Db2 kullanılırsa, performans maliyeti çok daha yüksektir.

z/OS *Depolama sınıfı belleğinin IBM MQ for z/OS ile nasıl çalıştığını*

IBM MQ for z/OS paylaşılan kuyruklarıyla depolama sınıfı belleğinin (SCM) kullanımına genel bakış.

Önemli: IBM z16 , Coupling Facility görüntüleri için Virtual Flash Bellek (Depolama Sınıfı Bellek ya da SCM olarak da bilinir) kullanımını desteklemek üzere son nesil IBM Z ® olması planlanmıştır. Daha fazla bilgi için bkz. IBM Z ve IBM LinuxONE 4Q 2023 Yönelimin Bildirimleri.

Alternatif olarak, daha büyük yapılar kullanmalı ya da iletileri SMDS ' ye boşaltmalısınız.

CFLEVEL 19 ya da üstü bir bağlaşım olanağı (CF) için SCM ayrılabilir. Daha sonra bu CF 'de tanımlanan yapılar, yapıların dolma olasılığını azaltmak için SCM' den yararlanacak şekilde yapılandırılabilir (tam bir yapı koşulu olarak bilinir). SCM 'yi kullanacak şekilde yapılandırılmış bir yapı sistem tarafından belirlenen bir noktayı geçtiğinde, CF verileri yapıdan SCM' ye taşımaya başlar ve bu da yeni veriler için yapıdaki alanı boşaltır.

Not: SCM 'nin kendisi doldurabildiği için, SCM' nin bir yapıya ayrılması, yalnızca bir yapının tam olarak olma olasılığını azaltır, ancak oluşma olasılığını tamamen ortadan kaldırmaz.

Bir yapı, CFRM (coupling facility resource manager; bağlaşım olanağı kaynak yöneticisi) ilkesinde hem **SCMALGORITHM** hem de **SCMMAXSIZE** anahtar sözcüklerini belirterek SCM ' yi kullanacak şekilde yapılandırılır.

Bu anahtar sözcükler belirtildikten ve CFRM ilkesi uygulandıktan sonra, yürürlüğe girmeleri için yapının yeniden oluşturulması ya da serbest bırakılması gerektiğini unutmayın.

SCMALGORITHM anahtar sözcük

SCM 'nin giriş/çıkış hızı gerçek depolamadan daha yavaş olduğu için CF, SCM' ye yazmanın ya da SCM ' den okumanın etkisini azaltmak için yapının beklenen kullanımına uyarlanmış bir algoritma kullanır.

Algoritma, *KEYPRIORITY1* değeri kullanılarak yapıya ilişkin CFRM ilkesinde **SCMALGORITHM** anahtar sözcüğü tarafından yapılandırılır. *KEYPRIORITY1* değerini yalnızca IBM MQ paylaşılan kuyrukları tarafından kullanılan liste yapılarıyla kullanmanız gerektiğini unutmayın.

KEYPRIORITY1 algoritması, çoğu uygulamanın iletileri paylaşılan bir kuyruktan öncelik sırasına göre alacağı varsayılarak çalışır; yani, bir uygulama bir ileti aldığında, en yüksek önceliğe sahip en eski iletiyi alır.

Bir yapı, sistem tarafından tanımlanan %90 eşiğini aşmaya başladığında, CF, bir sonraki alma olasılığı en düşük olan iletileri zamanuyumsuz olarak geçirmeye başlar. Bunlar, kuyruğa daha yakın zamanda konan daha düşük önceliklere sahip iletilerdir.

İletilerin yapıdan SCM ' ye zamanuyumsuz geçişi "ön hazırlık" olarak bilinir.

Ön hazırlık, SCM 'ye zamanuyumlu giriş/çıkış oluşması sırasında bir uygulamanın engellenme olasılığını azalttığı için SCM' yi kullanma performans maliyetini azaltır.

Ön hazırlığa ek olarak, *KEYPRIORITY1* algoritması, SCM ' den gelen iletileri zamanuyumsuz olarak geri getirir ve yeterli boş alan olduğunda yapıya geri getirir. *KEYPRIORITY1* algoritması için bu, yapının %70 'ten az ya da %70 'e eşit olduğu anlamına gelir.

SCM ' den gelen mesajları yapıya getirme eylemi "ön getirme" olarak bilinir.

Önceden getirme, bir uygulamanın SCM ' ye önceden hazırlanmış bir iletiyi alma ve CF iletiyi yapıya zamanuyumlu olarak geri getirirken beklemek zorunda kalma olasılığını azaltır.

SCMMAXSIZE anahtar sözcük

SCMMAXSIZE anahtar sözcüğü, bir yapı tarafından kullanılabilecek SCM miktarı üst sınırını tanımlar. SCM gerektiğinde CF tarafından yapıya ayrıldığı için, kullanılabilir toplam boş SCM miktarından büyük bir **SCMMAXSIZE** belirtilabilir. Bu "aşırı kesinleştirme" olarak bilinir.

Önemli: SCM ' yi asla aşırı kesinleştirme. Bunu yaparsanız, bu uygulamaya güvenen uygulamalar bekledikleri davranışı elde edemezler. Örneğin, paylaşılan kuyrukları kullanan IBM MQ uygulamaları beklenmeyen MQRC_STORAGE_MEDIUM_FULL neden kodlarını alabilir.

CF, SCM kullanımını izlemek için çeşitli veri yapılarını kullanır. Bu veri yapıları, CF ' ye ayrılan gerçek depolamada bulunur ve sonuç olarak, yapılar tarafından kullanılabilecek gerçek depolama miktarını azaltır. Bu veri yapıları tarafından kullanılan depolama alanı "artırılmış alan" olarak bilinir.

Bir yapı SCM ile yapılandırıldığında, CF ' den sabit artırılmış alan olarak bilinen yapıya az miktarda gerçek depolama ayrılır. Bu, yapı hiçbir zaman herhangi bir SCM kullanmasa da ayrılır. Yapıdaki veriler SCM 'de saklandığı için, CF' deki yedek gerçek depodan fazladan dinamik artırılmış alan ayrılır.

Veriler SCM 'den kaldırıldığında, dinamik artırılmış alan CF' ye döndürülür. Artırılmış alan, sabit ya da dinamik, hiçbir zaman bir yapıya ayrılan gerçek depodan alınmaz.

Artırılmış depolamaya ek olarak, bir yapı SCM kullanacak şekilde yapılandırıldığında, bu yapı tarafından kullanılan denetim depolama alanı miktarı artar. Bu, SCM ile yapılandırılan bir liste yapısının, SCM yapılandırılmadan aynı büyüklükteki bir yapıdan daha az giriş ve öğe içerebileceği anlamına gelir.

SCM ' nin yeni ya da var olan yapılar üzerindeki etkisini anlamak için CFSizer aracını kullanın.

Son önemli nokta, veriler yapıdan SCM ' ye taşındıktan ve dinamik artırılmış alan kullanıldıktan sonra yapının el ile ya da otomatik olarak değiştirilememesidir.

Yani, yapıya ayrılan depolama miktarı artırılamaz veya azaltılamaz, yapı tarafından kullanılan giriş-öğe oranı değiştirilemez ve bu şekilde devam eder. Yapıyı yeniden değiştirilebilecek hale getirmek için, yapının SCM ' de saklanan herhangi bir veriye sahip olmaması ve dinamik artırılmış depolamadan yararlanmaması gerekir.

z/OS *SCM neden kullanılıyor?*

Acil durum depolaması ve gelişmiş performans, IBM MQ for z/OSile SCM kullanımı için iki kullanım senaryodur.

Önemli: IBM z16 , Coupling Facility görüntüleri için Virtual Flash Bellek (Depolama Sınıfı Bellek ya da SCM olarak da bilinir) kullanımını desteklemek üzere son nesil IBM Z ® olması planlanmıştır. Daha fazla bilgi için bkz. IBM Z ve IBM LinuxONE 4Q 2023 Yönelimin Bildirimleri.

Alternatif olarak, daha büyük yapılar kullanmalı ya da iletileri SMDS ' ye boşaltmalısınız.

Bu bölümde iki olası senaryonun ardındaki teori tanıtılmaktadır. Senaryoları nasıl oluşturacağınıza ilişkin daha fazla ayrıntı için bkz:

- “Acil depolama-temel yapılandırma” sayfa 188
- “İyileştirilmiş performans-temel yapılandırma” sayfa 194

Önemli: SCM 'nin CF yapılarıyla kullanılması, IBM MQ' in belirli bir sürümüne bağlı değildir. Ancak, acil durum depolama senaryosu SMDS ve boşaltma kuralları gerektirdiğinden yalnızca IBM WebSphere MQ 7.1 ve sonraki sürümlerle çalışır.

Acil durum depolaması

SMDS ve ileti boşaltma, genişletilmiş bir kesinti sırasında bir MQRC_STORAGE_MEDIUM_FULL neden kodunun IBM MQ uygulamasına döndürülme olasılığını azaltmak için SCM ile birlikte kullanılabilir.

Genel bilgiler

Bir uygulama yapısında tek bir paylaşılan kuyruk yapılandırıldı. Koyma uygulaması iletileri paylaşılan kuyruğa koyar; alma uygulaması iletileri paylaşılan kuyruktan alır.

Normal çalışma sırasında kuyruk derinliğinin sıfıra yakın olması beklenir, ancak bir iş gereksinimi, sistemin alma uygulamasında iki saatlik bir kesintiye dayanması gerektiğini belirtir. Bu, paylaşılan kuyruğun koyma uygulamasından gelen iki saatlik iletileri içerebilmesi gerektiği anlamına gelir.

Şu anda bu işlem, varsayılan boşaltma kuralları ve SMDS kullanılarak gerçekleştirilir; böylece yapının boyutu en aza indirilir ve boşaltmayla ilişkili performans maliyeti azaltılır.

Paylaşılan kuyruğa gönderilen iletilerin hızının kısa ve orta vadede iki katına çıkması beklenir. Sistemin iki saatlik bir kesintiyi tolere edebilme gereksinimi hala var olsa da, CF ' de yapının boyutunu iki katına çıkaracak kadar gerçek depolama yoktur.

Uygulama yapısını içeren CF bir zEC12 makinesinde bulunduğundan, iki saatlik bir kesintiye izin verilebilmesi için yeterli SCM ' yi yapıyla yeterince ileti depolamak üzere ilişkilendirme olasılığı vardır.

Belirli bir süre içinde neler olduğunu göz önünde bulundurun:

1. Başlangıçta, sistem sabit durumdadır. Hem koyma hem de alma uygulaması olağan bir şekilde çalışıyor ve kuyruk derinliği sıfıra yakın ya da sıfıra yakın. Sonuç, uygulama yapısının büyük ölçüde boş olması.
2. Belirli bir zamanda, alma uygulaması beklenmeyen bir başarısızlığa uğrar ve durur. Koyma uygulaması iletileri kuyruğa yerleştirmeye devam eder ve uygulama yapısı dolmaya başlar.
3. Yapı %70 'e ulaştıktan sonra, ilk boşaltma kuralının koşulları karşılanır ve 32 KB 'den büyük ya da 32 KB' ye eşit tüm iletiler SMDS ' ye boşaltılır.

   Boşaltma kurallarına genel bakış için bkz. “SMDS ve offload kuralları” sayfa 183 .
4. İletiler paylaşılan kuyruğa konmaya devam ettikçe, yapı doldurmaya devam eder (ya yapıda depolanan ileti verileri ya da yapıda depolanan boşaltılmış iletilere ilişkin işaretçiler nedeniyle).

   Yapı %80 'e ulaştığında, ikinci boşaltma kuralı uygulanmaya başlar ve 4 KB ya da daha büyük iletiler SMDS ' ye boşaltılır.
5. Yapı %90 'ı geçtiğinde, tüm iletiler SMDS ' ye yüklenir ve yapıya yalnızca ileti işaretçileri yerleştirilir.

   Bu süre içinde, hazırlık öncesi algoritması çalışmaya başlar ve verileri yapıdan SCM ' ye taşımaya başlar. Kuyruktaki tüm iletilerin aynı öncelik olduğu varsayılarak, en yeni iletiler önceden hazırlandı.

   Tüm iletiler artık SMDS 'ye boşaltıldığından, SCM' ye taşınmakta olan veriler gerçek ileti verileri değil, bunun yerine SMDS ' deki iletilere işaretçiler.

   Sonuç olarak, yapı birleşiminde saklanabilecek iletilerin sayısı ve yapıyla ilişkili SCM ve SMDS çok büyüktür.

   Performans: Kesintinin bu aşamasında, uygulama SMDS ' ye yazmak zorunda kaldığından bir dereceye kadar performans düşüşüne uğrayabilir. Bu durumda, SCM ' nin kullanımı, performans açısından koyma uygulamasında sınırlayıcı bir faktör olmamalıdır. SCM, yapının dolmasını önlemek için ek alan sağlar.
6. Sonuç olarak, alma uygulaması yeniden kullanılabilir olur ve kesinti sona erer.

   Ancak SCM hala yapı tarafından kullanılıyor. Alma uygulaması, kuyruktaki iletileri okumaya başlar ve önce en eski, en yüksek öncelikli iletileri almaya başlar.

   Bu mesajlar yapı doldurmaya başlamadan önce yazıldığı için, tamamen yapının gerçek depolama kısmından ortaya çıkıyorlar.
7. Yapı boş olmaya başladığında, ön hazırlanma etkin olduğu eşiğin altına girer ve bu nedenle ön hazırlama durur.
8. Yapı kullanımı, boşaltma kurallarının yürürlüğe girdiği noktanın altında azalır, bu nedenle iletiler 63 KB 'den fazla olmadıkça artık SMDS' ye yüklenmez.

   Şu anda, önokuma algoritması verileri SCM ' den yapıya taşımaya başlar. Alma uygulaması iletileri SCM algoritmalarının beklediği sırayla kuyruktan aldığından, iletiler alma uygulaması için gerekli olmadan önce getirilir.

   Sonuç, alma uygulamasının SCM ' den zamanuyumlu olarak ileti getirilmesini hiçbir zaman beklemesi gerekmemesinden kaynaklanır.
9. Alma uygulaması kuyrukta aşağı doğru ilerlemeye devam ettikçe, SMDS ' ye boşaltılan iletileri almaya başlar.
10. Son olarak, sistem yine sabit bir durumda. SCM ya da SMDS ' de hiçbir ileti saklanmaz ve kuyruk derinliği sıfıra yakındır.

İyileştirilmiş performans

Bu senaryo, SMDS 'yi kullanma performans maliyetine maruz kalmadan paylaşılan bir kuyrukta saklanabilecek ileti sayısını artırmak için SCM' nin kullanılmasını açıklar.

Açıklama

Bu senaryoda, bir koyma ve alma uygulaması, uygulama yapısında saklanan paylaşılan bir kuyruk aracılığıyla iletişim kurar.

Koyma uygulaması, kısa bir süre içinde çok sayıda ileti koyduğunda, atım halinde çalışma eğilimindedir. Daha sonra, uzun bir süre içinde, hiç mesaj üretmez.

Alma uygulaması, her iletiyi sırayla işler ve her biri üzerinde karmaşık işleme gerçekleştirir. Sonuç olarak, koyma uygulaması çalışmaya başladığında, iletiler alındığından daha hızlı yerleştirildikçe kuyruk derinliğinin artmaya başladığı durumlar dışında, çoğu zaman kuyruk derinliği sıfırdır.

Uygulama duruncaya ve alma uygulamasının kuyruktaki tüm iletileri işlemek için yeterli zamanı bulununcaya kadar kuyruk derinliği artar.

Notlar:

1. Bu senaryoda, temel faktör performanstı. Kuyruğa gönderilen iletiler her zaman 63 KB 'den azdır ve bu nedenle hiçbir zaman SMDS' ye boşaltılması gerekmez.
2. Uygulama yapısı, tek bir "patlama" içine koyma uygulaması tarafından üzerine yerleştirilecek tüm iletileri içerecek kadar büyük olacak şekilde boyutlandırıldı.
3. Yapı doldurmaya başladığında bile iletilerin SMDS ' ye boşaltılmaması için boşaltma kurallarının tümü devre dışı bırakılmalıdır. Bunun nedeni, SMDS 'ye ileti yazılması ve SMDS' den ileti okunmasıyla ilişkili performans maliyetlerinin kabul edilemez olmasıdır.

Zaman içinde, bir ayırma uygulamasına gönderilen iletilerin sayısı birkaç büyüklük sırasıyla artmalıdır. Alma uygulamasının her iletiyi sıralı olarak işlemesi gerektiğinden, kuyruktaki ileti sayısı, yapının dolduğu noktaya kadar artar.

Bu noktada, koyma uygulaması bir ileti koyarken bir neden kodu (MQRC_STORAGE_MEDIUM_FULL) alır ve koyma işlemi başarısız olur. Koyma uygulaması, iletileri kuyruğa koyamadığı dönemleri yalnızca kısa bir süre tolere edebilir. Dönem çok uzunsa, uygulama sona erer.

Başvuruyu yeniden yazmak ya da uygulamayı almak için zamanınız ya da becerileriniz olmadığı varsayılarak, bu sorunun üç olası çözümü vardır:

1. Uygulama yapısının boyutunu artırın.
2. Kuyruk dolmaya başladığında iletilerin SMDS ' ye boşaltılması için uygulama yapısına boşaltma kuralları ekleyin.
3. SCM ' yi yapıyla ilişkilendirin.

İlk çözüm hızlı uygulanır, ancak CF ' de yeterli gerçek depolama yoktur.

İkinci çözümün uygulanması da hızlı olabilir, ancak SMDS ' ye boşaltmanın performans üzerindeki etkisi bu seçeneği kullanmak için çok önemli olarak kabul edilir.

SCM ' yi yapıyla ilişkilendiren üçüncü çözüm, kabul edilebilir bir maliyet ve performans dengesi sağlar.

SCM 'nin bir yapıyla ilişkilendirilmesi, kullanılan işlemleri alan artırılmış depolama alanı nedeniyle CF' de gerçek depolamanın daha yüksek kullanımıyla sonuçlanır. Ancak, gerçek gerçek depolama miktarı, ilk seçenekte kullanılan miktardan daha az olacaktır.

Bir diğer husus da SCM ' nin maliyetidir. Ancak bu maliyet gerçek depolamadan çok daha ucuz. Bu faktörler birleşerek üçüncü seçeneği ilk seçenekten daha ucuz hale getirir.

Üçüncü seçenek, olası olarak birinci seçeneğin yanı sıra gerçekleştirilemese de, CF tarafından kullanılan ön getirme ve ön hazırlama algoritmaları, performans farklarını kabul edilebilir ya da bazı durumlarda göz ardı edilebilir hale getirmek için birleşebilir.

Performans, iletileri boşaltmak için SMDS ' yi kullanmaktan çok daha iyi olabilir.

Belirli bir süre içinde neler olduğunu göz önünde bulundurun:

1. Başlangıçta, alma uygulaması etkindir ve iletilerin paylaşılan kuyruğa teslim edilmesini bekler. Koyma uygulaması etkin değil ve paylaşılan kuyruk boş.

2. Belirli bir zamanda, koyma uygulaması etkin duruma gelir ve paylaşılan kuyruğa çok sayıda ileti yerleştirmeye başlar. Alma uygulaması iletileri almaya başlar, ancak alma uygulaması koyma uygulamasından daha yavaş olduğundan kuyruk derinliği hızla artmaya başlar.

   Sonuç olarak, uygulama yapısı dolmaya başlar.
3. Zaman arttıkça, koyma uygulaması etkin olmaya devam ediyor. Uygulama yapısı yaklaşık %90 'a kadar doldurur.

   Bu, SCM ön hazırlama algoritmasının iletileri yapıdan SCM ' ye taşımaya başladığı ve yapıdaki alanı boşaltmaya başladığı andır.

   Alma uygulaması kuyruktan en eski, en yüksek öncelikli iletileri ilk önce aldığından, her zaman yapıdan ileti alır ve SCM ' den yapıya zamanuyumlu olarak ileti getirilmesini beklemesi gerekmez.
4. Koyma uygulaması hala etkin ve iletileri paylaşılan kuyruğa koyuyor. Ancak, SCM ' de yapıya sığmayan tüm iletileri saklamak için yeterli yer bulunduğundan, uygulama hiçbir zaman bir MQRC_STORAGE_MEDIUM_FULL neden kodu almaz.
5. Sonunda, koyacak başka ileti olmadığı için koyma uygulaması durur.

   Yapı kullanımda %90 'ın altına düştüğü ve alma uygulaması kuyruktaki iletileri işlemeye devam ettiği için ön hazırlama algoritması durur.
6. Alma uygulaması yapıda yer açmaya başladığında, ön getirme algoritması SCM ' den yapıya iletileri geri getirmeye başlar.

   Alma uygulaması iletileri önokuma algoritması tarafından beklenen sırayla işlediğinden, alma uygulaması ileti verilerinin SCM ' den yapıya zamanuyumlu olarak getirilmesini beklerken hiçbir zaman engellenmez.
7. Son olarak, alma uygulaması paylaşılan kuyruktaki tüm iletileri işler ve sonraki ileti kullanılabilir oluncaya kadar bekler. Yapı ve SCM ' de ileti yok.

z/OS *Acil depolama-temel yapılandırma*

IBM MQüzerinde acil durum depolaması için temel senaryoyu ayarlama.

Bu görev hakkında

Önemli: IBM z16 , Coupling Facility görüntüleri için Virtual Flash Bellek (Depolama Sınıfı Bellek ya da SCM olarak da bilinir) kullanımını desteklemek üzere son nesil IBM Z ® olması planlanmıştır. Daha fazla bilgi için bkz. IBM Z ve IBM LinuxONE 4Q 2023 Yönelimin Bildirimleri.

Alternatif olarak, daha büyük yapılar kullanmalı ya da iletileri SMDS ' ye boşaltmalısınız.

SMDS ve ileti boşaltma, genişletilmiş bir kesinti sırasında bir MQRC_STORAGE_MEDIUM_FULL neden kodunun IBM MQ uygulamasına döndürülme olasılığını azaltmak için SCM ile birlikte kullanılabilir.

Örneğin, işletmenizin kuyruğa ileti koyan bir uygulaması ve kuyruktan ileti alan bir uygulaması vardır. Olağan çalışma sırasında, kuyruk derinliğinin sıfıra yakın olmasını beklersiniz, ancak bir iş gereksinimi, sistemin, iletileri alan uygulamanın iki saatlik bir kesintisine izin verebileceğini gösterir.

Bu, kullanılmakta olan paylaşılan kuyruğun, koyma uygulamasından iki saatlik iletiler içerebilmesi gerektiği anlamına gelir. Şu anda bunu varsayılan boşaltma kurallarını ve SMDS ' yi kullanarak başarıyorsunuz.

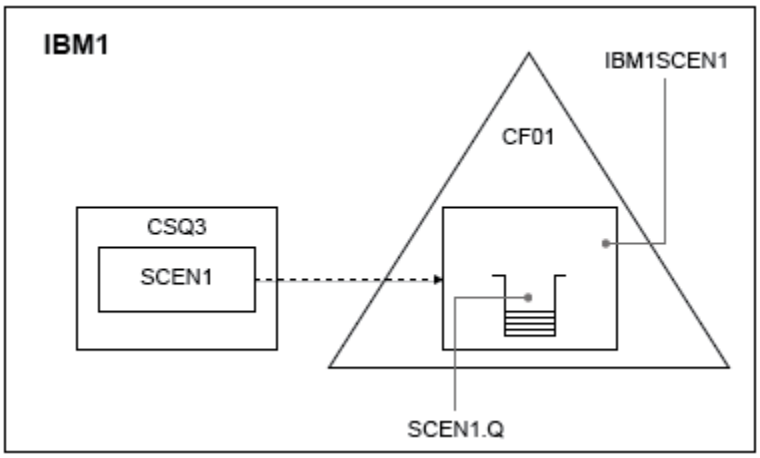
Paylaşılan kuyruğa gönderilen iletilerin hızının kısa ve orta vadede iki katına çıkmasını beklersiniz. Sistemin iki saatlik bir kesintiyi tolere edebilme gereksinmeniz olmasına rağmen, CF ' de yapının boyutunu iki katına çıkaracak kadar gerçek depolama bulunmamaktadır. Uygulama yapısını içeren CF bir zEC12 makinesinde bulunduğundan, yeterli SCM ' yi yeterli iletilerin saklanacağı yapıyla ilişkilendirme yeteneğiniz vardır; böylece iki saatlik bir kesinti tolere edilebilir.

Bu ilk senaryoda aşağıdakiler kullanılır:

- Tek bir kuyruk yöneticisi içeren IBM1kuyruk paylaşım grubu, CSQ3. Denetim yapısına ek olarak, kuyruk paylaşım grubu tek bir uygulama yapısı tanımladı: SCEN1.

- SCEN1 uygulama yapısının IBM1SCEN1 yapısı olarak saklandığı bağlaşım olanağı (CF) CF01. Bu yapı en fazla 1 GB boyutuna sahiptir.
- Uygulama yapısının kullandığı tek bir paylaşılan kuyruk (SCEN1.Q).

Bu yapılandırma, Şekil 62 sayfa 189içinde gösterilmektedir.



Şekil 62. Temel yapılandırma

Ayrıca, CSQ3 kuyruk yöneticisinin IBM1kuyruk paylaşım grubunun tek üyesi olduğunu varsayın.

IBM1SCEN1 yapısına ilişkin tanımı, bağlaşım olanağı kaynak yöneticisi (CFRM) ilkesine eklemeniz gerekir. Basitlik için yapı, PREFLIST(CF01) belirtilerek yalnızca tek bir bağlaşım olanağında (CF01) yaratılabilmesi için tanımlanır.

Uyarı: Üretim sisteminizde yüksek düzeyde kullanılabilirlik sağlamak için, IBM MQtarafından kullanılan yapılar için PREFLIST içine en az iki CF eklemelisiniz.

Yordam

1. Aşağıdaki komutu kullanarak CFRM ilkesini yenileyin:

```
SETXCF START,POLICY,TYPE=CFRM,POLNAME=IBM1SCEN1
```

   IBM1SCEN1yapısı için örnek CFRM ilkesi:

```
STRUCTURE
NAME(IBM1SCEN1)
SIZE(1024M)
INITSIZE(512M)
ALLOWAUTOALT(YES)
FULLTHRESHOLD(85)
PREFLIST(CF01)
ALLOWREALLOCATE(YES)
DUPLEX(DISABLED)
ENFORCEORDER(NO)
```

2. Aşağıdaki komutu kullanarak yapının doğru oluşturulduğunu doğrulayın:

```
D XCF,STR,STRNAME=IBM1SCEN1
```

   Bu noktada, STATUS hattı tarafından gösterilen yapınız kuyruk paylaşım grubuna ayrılmadı.
3. CFRM ilkesinde tanımlanan yapıdan yararlanmak için IBM MQ ' i yapılandırın.

a. Bir IBM MQ CFSTRUCT nesnesi yaratmak için DEFINE CFSTRUCT komutunu SCEN1 yapı adıyla birlikte kullanın:

```
DEFINE CFSTRUCT(SCEN1)
CFCONLOS(TOLERATE)
CFLEVEL(5)
DESCR('Structure for SCM scenario 1')
RECOVER(NO)
RECAUTO(YES)
OFFLOAD(DB2)
OFFLD1SZ(64K) OFFLD1TH(70)
OFFLD2SZ(64K) OFFLD2TH(80)
OFFLD3SZ(64K) OFFLD3TH(90)
```

.

b. DISPLAY CFSTRUCT komutunu kullanarak yapıyı doğrulayın.

c. Aşağıdaki MQSC komutunu kullanarak SCEN1 yapısını kullanmak için SCEN1.Q paylaşılan kuyruğunu tanımlayın:

```
DEFINE QLOCAL(SCEN1.Q) QSGDISP(SHARED) CFSTRUCT(SCEN1) MAXDEPTH(999999999)
```

4. Kuyruğa tek bir ileti koymak için IBM MQ Explorer komutunu kullanın SCEN1.Q ve iletiyi yeniden kapatın.
5. Yapının ayrılmış olup olmadığını denetlemek için aşağıdaki komutu verin:

```
D XCF,STR,STRNAME=IBM1SCEN1
```

STATUS satırının ALLOCATEDgösterdiğini, komutun çıkışını geri verin.

Sonuçlar

Temel yapılandırmayı yarattınız. Artık, seçtiğiniz yöntemi kullanarak yapılandırmanızın temel çizgisi performansına ilişkin bir fikir edinebilirsiniz.

Sonraki adım

SMDS ve SCM ' yi ilk yapıya ekle

İlgili kavramlar

“Paylaşılan kuyruklar ile depolama sınıfı belleğinin kullanılması” sayfa 181

IBM MQ for z/OS paylaşılan kuyruklarıyla kullanıldığında SCM (storage class memory; depolama sınıfı belleği) kullanımı avantajlı olabilir.

z/OS *SMDS ve SCM ' nin ilk yapıya eklenmesi*

IBM MQüzerinde acil durum depolaması için SMDS ve SCM ' yi ekleme.

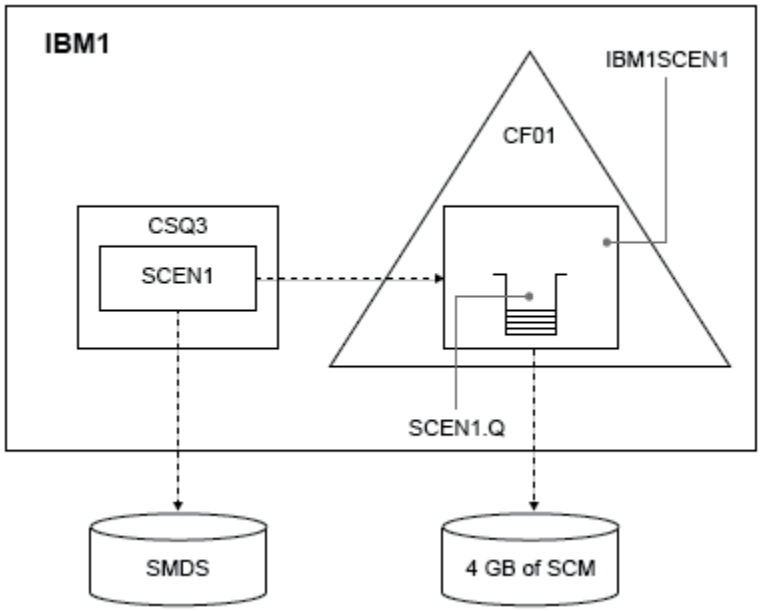
Bu görev hakkında

Önemli: IBM z16 , Coupling Facility görüntüleri için Virtual Flash Bellek (Depolama Sınıfı Bellek ya da SCM olarak da bilinir) kullanımını desteklemek üzere son nesil IBM Z ® olması planlanmıştır. Daha fazla bilgi için bkz. IBM Z ve IBM LinuxONE 4Q 2023 Yönelimin Bildirimleri.

Alternatif olarak, daha büyük yapılar kullanmalı ya da iletileri SMDS ' ye boşaltmalısınız.

Görevin bu bölümü, “Acil depolama-temel yapılandırma” sayfa 188içinde açıklanan temel yapılandırmayı kullanır. Senaryo, paylaşılan ileti veri kümelerinin (SMDS) ve daha sonra SCM ' nin ilk yapıya eklenmesini açıklar.

Bu son yapılandırma Şekil 63 sayfa 191içinde gösterilmektedir.



Şekil 63. Acil depolama için SMDS ve SCM ekleme yapılandırması

Yordam

1. Aşağıda gösterildiği gibi **CSQ4SMDS** örnek JCL ' yi düzenleyerek SCEN1 uygulama yapısının kullandığı SMDS veri kümesini oluşturun:

```
//CSQ4SMDS JOB NOTIFY=&SYSUID
//*
//* Allocate SMDS
//*
//DEFINE EXEC PGM=IDCAMS,REGION=4M
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
DEFINE CLUSTER           -
(NAME(CSQSMDS.SCEN1.CSQ3.SMDS) -
MEGABYTES(5000 3000)     -
LINEAR            -
SHAREOPTIONS(2 3) )      -
DATA                -
(NAME(CSQSMDS.SCEN1.CSQ3.SMDS.DATA) )
/*
//*
//* Format the SMDS
//*
//FORM   EXEC PGM=CSQJUFMT,COND=(0,NE),REGION=0M
//STEPLIB DD DSN=MQ800.SCSQANLE,DISP=SHR
//     DD DSN=MQ800.SCSQAUTH,DISP=SHR
//SYSUT1  DD DISP=OLD,DSN=CSQSMDS.SCEN1.CSQ3.SMDS
//SYSPRINT DD SYSOUT=*
```

2. SCEN1 uygulama yapısını değiştirmek, boşaltma için SMDS ' yi kullanmak ve varsayılan boşaltma kurallarını uygulamak için ALTER CFSTRUCT komutunu verin:

```
ALTER CFSTRUCT(SCEN1) OFFLOAD(SMDS) OFFLD1SZ(32K) OFFLD2SZ(4K) OFFLD3SZ(0K)
DSGROUP('CSQSMDS.SCEN1.*.SMDS') DSBLOCK(1M)
```

   Aşağıdakileri unutmayın:

   - SCEN1.Q , SCEN1 uygulama yapısındaki tek paylaşılan kuyruk olduğundan **DSBLOCK** değeri, mümkün olan en büyük değer olan 1Molarak ayarlandı. Bu, senaryomuz için en verimli ayar olmalı.
   - Koyma uygulaması tarafından gönderilen iletiler 30 KB olduğundan, ikinci boşaltma kuralı yerine getirilinceye kadar SMDS ' ye boşaltma işlemi başlamaz, yapı %80 dolu olduğunda.

3. Test uygulamanızı yeniden çalıştırın.

Kuyruktaki iletilerin artan saklama alanına dikkat edin.

4. Aşağıdaki yordamı gerçekleştirerek IBM1SCEN1 yapısına 4 GB SCM ekleyin:
   a) Aşağıdaki komutu vererek, CF01'e ne kadar SCM kurulduğunu ve bu SCM' ye ne kadar ayrıldığını denetleyin:

```
D CF,CFNAME=CF01
```

   b) Kullanılabilir depolama alanını görmek için görüntülenen çıkışın STORAGE CONFIGURATION bölümündeki STORAGE-CLASS MEMORY şekillerini denetleyin.
   c) CFRM ilkesini SCMMAXSIZE ve SCMALGORITHM anahtar sözcükleriyle aşağıda gösterildiği gibi güncelleyin:

```
STRUCTURE
NAME(IBM1SCEN1)
SIZE(1024M)
INITSIZE(512M)
ALLOWAUTOALT(YES)
FULLTHRESHOLD(85)
PREFLIST(CF01)
ALLOWREALLOCATE(YES)
DUPLEX(DISABLED)
ENFORCEORDER(NO)
SCMMAXSIZE(4G)
SCMALGORITHM(KEYPRIORITY1)
```

5. Aşağıdaki komutu girerek CFRM ilkesini etkinleştirin:

```
SETXCF START,POLICY,TYPE=CFRM,POLNAME=polname
```

6. IBM1SCEN1 yapısını yeniden oluşturun.
   Önceki değişiklikleri yaptığınızda yapı ayrıldığı için bu yordamı gerçekleştirmeniz gerekir.

   Yapıyı yeniden oluşturmak için aşağıdaki komutu verin:

```
SETXCF START,REBUILD,STRNM=IBM1SCEN1
```

Sonuçlar

Yapılandırmanıza SCM ' yi başarıyla eklediniz.

Sonraki adım

Sisteminizin performansını optimize edin. Ek bilgi için bkz. “Depolama sınıfı bellek kullanımının eniyilenmesi” sayfa 192 .

z/OS *Depolama sınıfı bellek kullanımının eniyilenmesi*

Depolama sınıfı bellek (SCM) kullanımınızı nasıl geliştirdiğiniz.

Önemli: IBM z16 , Coupling Facility görüntüleri için Virtual Flash Bellek (Depolama Sınıfı Bellek ya da SCM olarak da bilinir) kullanımını desteklemek üzere son nesil IBM Z ® olması planlanmıştır. Daha fazla bilgi için bkz. IBM Z ve IBM LinuxONE 4Q 2023 Yönelimin Bildirimleri.

Alternatif olarak, daha büyük yapılar kullanmalı ya da iletileri SMDS ' ye boşaltmalısınız.

Aşağıdaki komutu çalıştırın:

```
D XCF,STR,STRNAME=IBM1SCEN1
```

Önceki testler nedeniyle yapı zaten ileti verileriyle dolu olduğundan, yeniden oluşturma işleminin bir kısmı yapıdan SCM ' ye bazı iletileri önceden hazırlama işlemini içeriyordu. Bu işlem önceki komut kullanılarak başlatıldı.

Bu komutun ürettiği çıkış, örneğin:

```
ACTIVE STRUCTURE
----------------
ALLOCATION TIME: 06/17/2014 09:28:50
CFNAME : CF01
COUPLING FACILITY: 002827.IBM.02.00000000B8D7
PARTITION: 3B CPCID: 00
STORAGE CONFIGURATION ALLOCATED MAXIMUM %
ACTUAL SIZE: 1024 M 1024 M 100
AUGMENTED SPACE: 3 M 142 M 2
STORAGE-CLASS MEMORY: 88 M 4096 M 2
ENTRIES: 120120 1089536 11
ELEMENTS: 240240 15664556 1
SPACE USAGE IN-USE TOTAL %
ENTRIES: 84921 219439 38
ELEMENTS: 2707678 3149050 85
EMCS: 2 282044 0
LOCKS: 1024
SCMHIGHTHRESHOLD : 90
SCMLOWTHRESHOLD : 70
ACTUAL SUBNOTIFYDELAY: 5000
PHYSICAL VERSION: CD5186A0 2BD8B85C
LOGICAL VERSION: CD515C50 CE2ED258
SYSTEM-MANAGED PROCESS LEVEL: 9
XCF GRPNAME : IXCLO053
DISPOSITION : KEEP
ACCESS TIME : NOLIMIT
MAX CONNECTIONS: 32
# CONNECTIONS : 1
CONNECTION NAME ID VERSION  SYSNAME JOBNAME  ASID STATE
--------------- -- --------  ------- -------  ---- ----------------
CSQEIBM1CSQ301  01 00010059 SC61    CSQ3MSTR 0091 ACTIVE
```

Komutun çıkışında aşağıdakilere dikkat edin:

- Bu STORAGE_CLASS MEMORY , yapıya 4096 MB SCM ' lik bir **MAXIMUM** eklendiğini doğrular.
- Ön hazırlık için kullanılan STORAGE-CLASS MEMORY miktarının ALLOCATED şekli. Şu anda SCM eklenmeden önce hiçbir şey olmadığı yapıda boş alan var.
- SCM kullanımını izlemek için kullanılan AUGMENTED SPACE miktarı.
- Ön aşama algoritmasının verileri yapıdan SCM ' ye taşımaya başladığı nokta, yapının %90 dolu olduğu andır. Bu, yapılandırılamaz **SCMHIGHTHRESHOLD** özelliğiyle gösterilir.
- Ön getirme algoritmasının SCM ' den yapıya veri taşımaya başladığı nokta, yapının %70 dolu olduğu zamandır. Bu, yapılandırılamaz **SCMLOWTHRESHOLD** özelliğiyle gösterilir.

Artık SCM kullanımını eniyilemek için çeşitli yolları sınayabilirsiniz. Aşağıdakileri unutmayın:

- İletileri saklamak için SCM kullanıldıktan sonra, SCM ' den tüm verileri kaldırıncaya kadar yapıyı değiştiremezsiniz.

  Bu durumda bu, giriş-öğe oranının SCM ilk kullanıldığında yerinde olan değerde dondurulduğu anlamına gelir. Hazırlık öncesi algoritması verileri SCM ' ye taşımaya başlamadan önce, yapının istediğiniz durumda olduğundan emin olmanız gerekir.
- SCM kullanılmadan önce geçerli yapı boyutu doğru mu?

  Örneğin, **INITSIZE** değerini 512 MB 'den 1 GB' ye çıkardınız mı?

  Bunu yapmazsanız, otomatik değişiklik için yapınızı etkinleştirmiş olsanız da, hazırlık öncesi algoritması, değişiklik başlamadan önce verileri SCM ' ye taşımaya başlayacaktır. Sonuç olarak, yapı 512 MB gerçek depolama alanı kullanılarak dondurulur.
- SCM kullanılmadan önce giriş-öğe oranı doğru mu?

  Bu senaryonun amacı, yapıda ve SCM ' de bir bütün olarak saklanabilecek boşaltılan ileti işaretçisinin sayısını artırmanın yanı sıra, yapı deposunda mümkün olduğunca çok ileti bulundurmaktır. Bu iletilere erişmek, SMDS ' deki iletilere erişmekten daha hızlıdır.

  Bu nedenle, iletilerin depolanması için iyi olan bir giriş-öğe oranıyla başlayan bir yapıya ve daha sonra, prestage algoritması ilk başlamadan önce ileti işaretçileri depolamak için iyi bir orana geçiş yapmanız gerekir. Bu geçiş, kısmen IBM MQ boşaltma kurallarından yararlarak gerçekleştirilebilir.

Aşağıdaki komutu vererek boşaltma kurallarını değiştirin:

```
ALTER CFSTRUCT(SCEN1) OFFLD1SZ(0K)
```

Giriş-öğe oranlarını eniyilemek için birkaç çalıştırma gerçekleştirmeniz gerekebilir.

Aşağıdaki çizelgede, acil durum depolama senaryosunun farklı aşamalarında kuyruğa konan ileti sayısındaki olası iyileştirmeler gösterilmektedir.

Çizelge 17. Acil durum depolama senaryosuna ilişkin sonuçların karşılaştırılması

Test açıklaması	**İleti sayısı**	**Kuyruğu doldurma süresi (saniye)**
Temel yapılandırma	27.850	3.2
Varsayılan boşaltma kurallarına sahip SMDS	205.000	158
Varsayılan boşaltma kurallarına sahip SCM	828.610	469
Ayarlı boşaltma kuralları olan SCM	1.135.775	679

Tablodaki son satır, boşaltma kurallarının ayarlanmasında gerekli etkinin olduğunu gösterir.

Bu şekillerde herhangi bir iyileşme olup olmadığını görmek için sisteminizi incelemeniz gerekir. Örneğin, kullanılabilir SMDS depolamanız tükenebilir. Daha fazla SMDS saklama alanı ayırabiliyorsanız, kuyruktaki ileti sayısını önemli ölçüde artırabilirsiniz.

z/OS *İyileştirilmiş performans-temel yapılandırma*

IBM MQüzerinde paylaşılan kuyrukları kullanarak daha yüksek performans için temel bir senaryo ayarlama.

Bu görev hakkında

Önemli: IBM z16 , Coupling Facility görüntüleri için Virtual Flash Bellek (Depolama Sınıfı Bellek ya da SCM olarak da bilinir) kullanımını desteklemek üzere son nesil IBM Z ® olması planlanmıştır. Daha fazla bilgi için bkz. IBM Z ve IBM LinuxONE 4Q 2023 Yönelimin Bildirimleri.

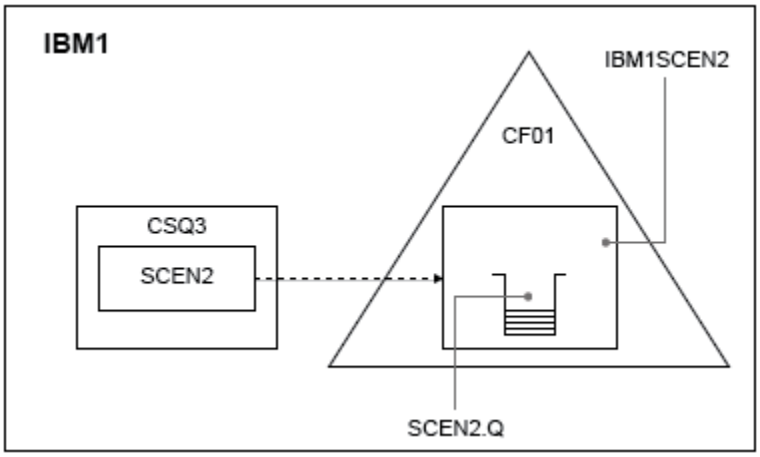
Alternatif olarak, daha büyük yapılar kullanmalı ya da iletileri SMDS ' ye boşaltmalısınız.

Bu senaryoda, SMDS ' yi kullanma performans maliyetine maruz kalmadan paylaşılan bir kuyrukta saklanabilecek iletilerin sayısını artırmak için SCM kullanımı açıklanmaktadır.

Bu ilk senaryo, acil durum depolaması için kullanılana çok benzer ve aşağıda belirtilenleri kullanır:

- Tek bir kuyruk yöneticisi içeren `IBM1`kuyruk paylaşım grubu, `CSQ3`. Denetim yapısına ek olarak, kuyruk paylaşım grubu tek bir uygulama yapısı tanımladı: `SCEN2`.
- `SCEN2` uygulama yapısının `IBM1SCEN2` yapısı olarak saklandığı bağlaşım olanağı (CF) `CF01`. Bu yapının maksimum boyutu 2 GB 'dir.
- Uygulama yapısını kullanmak üzere yapılandırılan tek bir paylaşılan kuyruk (`SCEN2.Q`).

Bu yapılandırma, Şekil 64 sayfa 195içinde gösterilmektedir.



Şekil 64. Temel yapılandırma

Ayrıca, CSQ3 kuyruk yöneticisinin IBM1kuyruk paylaşım grubunun tek üyesi olduğunu varsayın.

IBM1SCEN2 yapısına ilişkin tanımı, bağlaşım olanağı kaynak yöneticisi (CFRM) ilkesine eklemeniz gerekir. Basitlik için yapı, PREFLIST(CF01) belirtilerek yalnızca tek bir bağlaşım olanağında (CF01) yaratılabilmesi için tanımlanır.

IBM1SCEN2yapısı için örnek CFRM ilkesi:

```
STRUCTURE
NAME(IBM1SCEN2)
SIZE(2048M)
INITSIZE(2048M)
ALLOWAUTOALT(YES)
FULLTHRESHOLD(85)
PREFLIST(CF01)
ALLOWREALLOCATE(YES)
DUPLEX(DISABLED)
ENFORCEORDER(NO)
```

Yapının yeniden boyutlandırılamamasını sağlayan **INITSIZE** ve **SIZE** anahtar sözcüklerinin her ikisi de 2048M değerine sahiptir.

Yordam

1. Aşağıdaki komutu kullanarak CFRM ilkesini yenileyin:

   ```
   SETXCF START,POLICY,TYPE=CFRM,POLNAME=IBM1SCEN2
   ```

2. Aşağıdaki komutu kullanarak yapının doğru oluşturulduğunu doğrulayın:

   ```
   D XCF,STR,STRNAME=IBM1SCEN2
   ```

   Önceki komutun verilmesi aşağıdaki çıktıyı verir:

   ```
   RESPONSE=SC61
   IXC360I 07.58.51 DISPLAY XCF 581
   STRNAME: IBM1SCEN2
   STATUS: NOT ALLOCATED
   POLICY INFORMATION:
   POLICY SIZE : 2048 M
   POLICY INITSIZE: 2048 M
   POLICY MINSIZE : 1536 M
   FULLTHRESHOLD : 85
   ALLOWAUTOALT : YES
   REBUILD PERCENT: N/A
   DUPLEX : DISABLED
   ALLOWREALLOCATE: YES
   ```

```
PREFERENCE LIST: CF01
ENFORCEORDER : NO
EXCLUSION LIST IS EMPTY

EVENT MANAGEMENT: MESSAGE-BASED   MANAGER SYSTEM NAME: SC53
MANAGEMENT LEVEL : 01050107
```

Bu noktada, STATUS hattı tarafından gösterilen yapınız kuyruk paylaşım grubuna ayrılmadı.

3. CFRM ilkesinde tanımlanan yapıdan yararlanmak için IBM MQ ' i yapılandırın.
   a. Bir IBM MQ CFSTRUCT nesnesi yaratmak için, yapı adı SCEN2 olan DEFINE CFSTRUCT komutunu kullanın.

```
DEFINE CFSTRUCT(SCEN2)
CFCONLOS(TOLERATE)
CFLEVEL(5)
DESCR('Structure for SCM scenario 2')
RECOVER(NO)
RECAUTO(YES)
OFFLOAD(DB2)
OFFLD1SZ(64K) OFFLD1TH(70)
OFFLD2SZ(64K) OFFLD2TH(80)
OFFLD3SZ(64K) OFFLD3TH(90)
```

   b. DISPLAY CFSTRUCT komutunu kullanarak yapıyı denetleyin.
   c. Aşağıdaki MQSC komutunu kullanarak SCEN2 yapısını kullanmak için SCEN2.Q paylaşılan kuyruğunu tanımlayın:

```
DEFINE QLOCAL(SCEN2.Q) QSGDISP(SHARED) CFSTRUCT(SCEN2) MAXDEPTH(999999999)
```

4. Kuyruğa tek bir ileti koymak için IBM MQ Gezgin SCEN2.Q ' i kullanın ve iletiyi yeniden kapatın.
5. Yapının ayrılmış olup olmadığını denetlemek için aşağıdaki komutu verin:

```
D XCF,STR,STRNAME=IBM1SCEN2
```

Bir kısmı gösterilecek şekilde komutun çıkışını gözden geçirin ve STATUS satırının ALLOCATEDgösterdiğinden emin olun.

```
RESPONSE=SC61
IXC360I 08.31.27 DISPLAY XCF 703
STRNAME: IBM1SCEN2
STATUS: ALLOCATED
EVENT MANAGEMENT: MESSAGE-BASED
TYPE: SERIALIZED LIST
POLICY INFORMATION:
POLICY SIZE : 2048 M
POLICY INITSIZE: 2048 M
POLICY MINSIZE : 1536 M
FULLTHRESHOLD : 85
ALLOWAUTOALT : YES
REBUILD PERCENT: N/A
DUPLEX : DISABLED
ALLOWREALLOCATE: YES
PREFERENCE LIST: CF01
ENFORCEORDER : NO
EXCLUSION LIST IS EMPTY
```

Ayrıca, SPACE USAGE bölümündeki alanların değerlerini de göz önünde bulundurun:

- Girişler
- öğeler
- EMCS
- kilitler

Aşağıdaki değerlerin bir örneği:

```
SPACE USAGE   IN-USE      TOTAL    %
ENTRIES:       344686     345242   99
ELEMENTS:     6548455    6548467   99
EMCS:               2     780318    0
LOCKS:           1024
```

Sonuçlar

Temel yapılandırmayı yarattınız. Artık, seçtiğiniz yöntemi kullanarak yapılandırmanızın temel çizgisi performansına ilişkin bir fikir edinebilirsiniz.

Sonraki adım

Temel senaryoyu test etmelisin. Örnek olarak, uygulamaları gösterilen sırayla başlatarak ve eşzamanlı olarak çalıştırarak aşağıdaki üç uygulamayı kullanabilirsiniz.

1. Yürürlükteki derinliği istemek için bir PCF uygulaması kullanın (**CURDEPTH**) her beş saniyede bir SCEN2.Q değeri. Çıkış, zaman içinde kuyruğun derinliğini çizmek için kullanılabilir.
2. Tek bir iş parçacıklı uygulama, sonsuz bekleme ile get (alma) işlevini kullanarak SCEN2.Q' den sürekli olarak ileti alır. Kaldırılan iletilerin işlenmesinin benzetimini yapmak için, uygulamanın alınması, kaldırdığı her on ileti için dört milisaniye süreyle duraklatılır.
3. Tek iş parçacıklı bir koyma uygulaması, SCEN2.Q' e toplam bir milyon 4 KB kalıcı olmayan ileti koyar. Bu uygulama, her iletiyi koyma arasında duraksamaz, böylece iletiler SCEN2.Q ' e alma uygulamasının alabildiğinden daha hızlı yerleştirilebilir.

   Sonuç olarak, koyma uygulaması çalışırken SCEN2.Q derinliği artar.

   IBM1SCEN2 yapısı doldurulduğunda ve koyan uygulama bir MQRC_STORAGE_MEDIUM_FULL neden kodu aldığında, koyma uygulaması sonraki iletiyi kuyruğa yerleştirme girişiminde bulunmadan önce beş saniye boyunca uyur.

CURDEPTH uygulamasının sonuçlarını belirli bir süre içinde çizebilirsiniz. Bir çeşit testere-diş dalga çıkışı elde edilir ve koyma uygulaması kuyruğun kısmen boş kalmasına izin vermek için duraklar.

“İlk yapıya SCM eklenmesi” sayfa 197başlıklı konuya geçin.

İlgili kavramlar

“Paylaşılan kuyruklar ile depolama sınıfı belleğinin kullanılması” sayfa 181

IBM MQ for z/OS paylaşılan kuyruklarıyla kullanıldığında SCM (storage class memory; depolama sınıfı belleği) kullanımı avantajlı olabilir.

z/OS *İlk yapıya SCM eklenmesi*

IBM MQüzerinde daha yüksek performans için SCM ekleme.

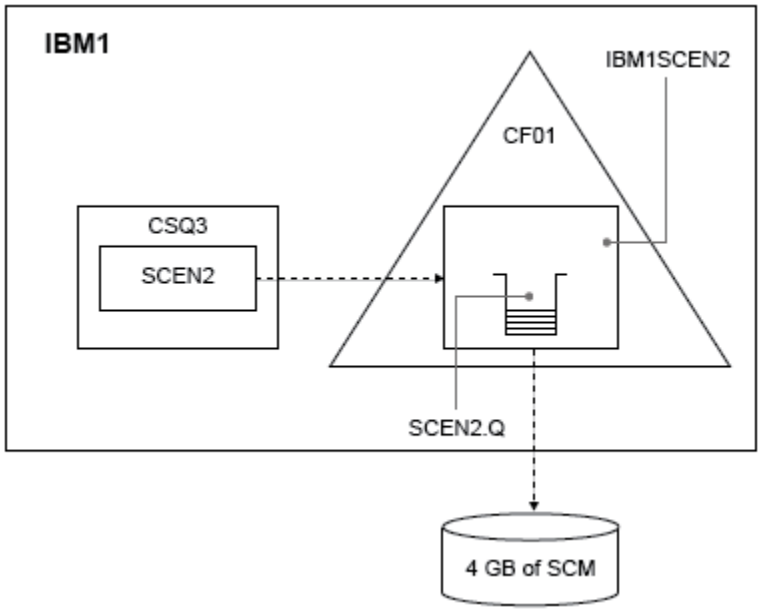
Bu görev hakkında

Önemli: IBM z16 , Coupling Facility görüntüleri için Virtual Flash Bellek (Depolama Sınıfı Bellek ya da SCM olarak da bilinir) kullanımını desteklemek üzere son nesil IBM Z ® olması planlanmıştır. Daha fazla bilgi için bkz. IBM Z ve IBM LinuxONE 4Q 2023 Yönelimin Bildirimleri.

Alternatif olarak, daha büyük yapılar kullanmalı ya da iletileri SMDS ' ye boşaltmalısınız.

Görevin bu bölümü, “İyileştirilmiş performans-temel yapılandırma” sayfa 194içinde açıklanan temel yapılandırmayı kullanır. Senaryo, SCM ' nin ilk yapıya eklenmesini açıklar.

Bu son yapılandırma Şekil 65 sayfa 198içinde gösterilmektedir.



Şekil 65. Daha yüksek performans için SCM ekleme yapılandırması

Yordam

1. Aşağıdaki yordamı gerçekleştirerek IBM1SCEN2 yapısına 4 GB SCM ekleyin:
   a) Aşağıdaki komutu vererek, CF01'e ne kadar SCM kurulduğunu ve bu SCM' ye ne kadar ayrıldığını denetleyin:

   ```
   D CF,CFNAME=CF01
   ```

   b) Kullanılabilir depolama alanını görmek için görüntülenen çıkışın STORAGE CONFIGURATION bölümündeki STORAGE-CLASS MEMORY şekillerini denetleyin.
   c) CFRM ilkesini SCMMAXSIZE ve SCMALGORITHM anahtar sözcükleriyle aşağıda gösterildiği gibi güncelleyin:

   ```
   STRUCTURE
   NAME(IBM1SCEN2)
   SIZE(2048M)
   INITSIZE(2048M)
   ALLOWAUTOALT(YES)
   FULLTHRESHOLD(85)
   PREFLIST(CF01)
   ALLOWREALLOCATE(YES)
   DUPLEX(DISABLED)
   ENFORCEORDER(NO)
   SCMMAXSIZE(4G)
   SCMALGORITHM(KEYPRIORITY1)
   ```

2. Aşağıdaki komutu girerek CFRM ilkesini etkinleştirin:

   ```
   SETXCF START,POLICY,TYPE=CFRM,POLNAME=IBM1SCEN2
   ```

3. IBM1SCEN2 yapısını yeniden oluşturun.
   Önceki değişiklikleri yaptığınızda yapı ayrıldığı için bu yordamı gerçekleştirmeniz gerekir.

   Yapıyı yeniden oluşturmak için aşağıdaki komutu verin:

   ```
   SETXCF START,REBUILD,STRNM=IBM1SCEN2
   ```

4. Yapının yeni yapılandırmasını onaylamak için aşağıdaki komutu verin:

```
D XCF,STR,STRNAME=IBM1SCEN2
```

Komutun çıkışını gözden geçirin:

```
SPACE USAGE     IN-USE       TOTAL      %
ENTRIES:          33         342684     0
ELEMENTS:         48        6503697     0
EMCS:              2         575600     0
LOCKS:                         1024
```

Sonuçlar

SCM kullanmak için gereken denetim depolaması artışı ile gerçek depolama kullanımında yapılan değişikliği hesaplayın.

- Yapıya SCM eklenmeden önce, yapı “İyileştirilmiş performans-temel yapılandırma” sayfa 194içinde gösterildiği gibi bu toplamlara sahiptir:
  - 345,242 giriş
  - 6.548,467 öğe
  - 780,318 EMCS
- Yapıya SCM eklendikten sonra yapı şu toplamlara sahiptir:
  - 342.684 giriş
  - 6.503.697 öğe
  - 575.600 EMCS

Bu şekiller kullanılarak, SCM eklendikten sonra yapı aşağıdaki şekilde küçültülür:

- 2558 giriş
- 44.770 öğe
- 204.718 EMCS

SCM 'yi yönetmek için kullanılan yapı depolama alanı miktarı, 4 GB SCM ayrılmış 2 GB' lik bir yapı için aşağıdaki gibidir:

```
(2558 + 44,770 + 204,718) * 256 = 61.5 MB
```

Daha fazla SCM eklenmesinin, SCM ' yi izlemek için kullanılan kontrol depolama miktarı arttığından ve ayrılan SCM miktarı arttığından, yapının boyutunda yalnızca marjinal bir azalma elde edebileceğini unutmayın.

Sonraki adım

“İyileştirilmiş performans-temel yapılandırma” sayfa 194' in son bölümünde açıklanan sınamaları yineleyin.

Düzeltilen uygulamanın sonuçlarını belirli bir süre içinde çizebilirsiniz. Çizimi daha önce elde edilenle karşılaştırarak, şimdi bir testere dişi dalgası olmadan bir çıkış elde edebilirsiniz, çünkü koyma uygulaması artık kuyruğun kısmen boşalmasını beklemek zorunda değildir.

Daha fazla bilgi için bkz. MP16: WebSphere MQ for z/OS -Kapasite planlama ve ayarlama.

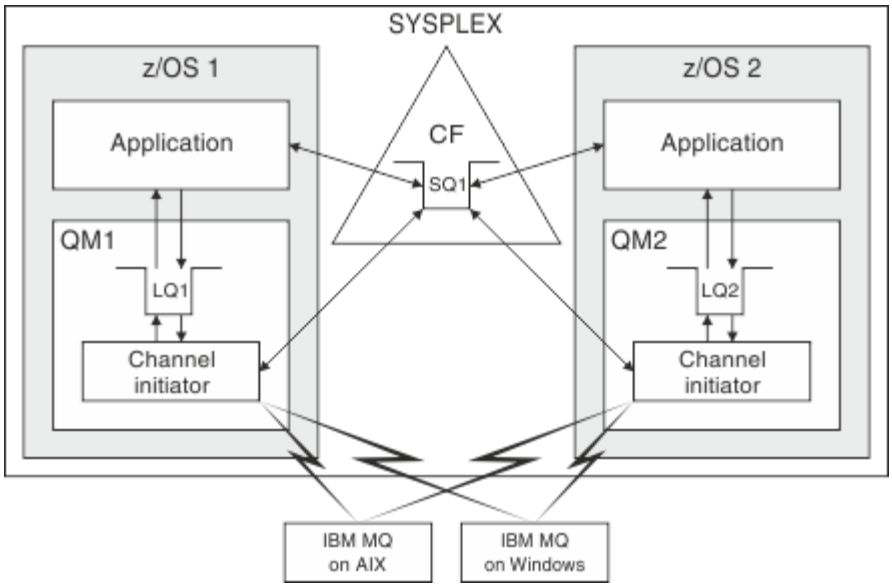
z/OS Dağıtılmış kuyruğa alma ve kuyruk paylaşım grupları

Dağıtılmış kuyruklama ve kuyruk paylaşım grupları, uygulama sistemlerinizin kullanılabilirliğini artırmak için kullanabileceğiniz iki yöntemdir. Bu tekniklerle ilgili daha fazla bilgi bulmak için bu konuyu kullanın.

To complement the high availability of messages on shared queues, the distributed queuing component of IBM MQ has additional functions to provide the following:

- Ağ üzerinde daha yüksek kullanılabilirlik.
- Kuyruk paylaşım grubuna gelen ağ bağlantıları için artırılmış kapasite.

Şekil 66 sayfa 200 , dağıtılmış kuyruğa alma ve kuyruk paylaşım gruplarını gösterir. Bir sistem birleşimi içinde iki kuyruk yöneticisini gösterir; bunların her ikisi de aynı kuyruk paylaşım grubuna ait olur. Her ikisi de SQ1paylaşılan kuyruğuna erişebilirler. Ağdaki kuyruk yöneticileri (örneğin, AIX ve Windows ' de), kuyruk yöneticisinin kanal başlatıcısı aracılığıyla bu kuyruğa ileti yerleştirebilir. Kuyruğun her iki kuyruk yöneticisinde de eşkopyalanmış uygulamalar.



Şekil 66. Dağıtılmış kuyruğa alma ve kuyruk paylaşım grupları

İlgili kavramlar

“Paylaşılan kanallar” sayfa 200
IBM MQ for z/OSile paylaşılan kanalların kavramlarını ve bunların kullanımını anlamak için bu konuyu kullanın.

“Grup içi kuyruğa alma” sayfa 202
Grup içi kuyruğa alma, bir kuyruk paylaşım grubundaki kuyruk yöneticileri arasında iletilerin aktarılmasına olanak sağlar.

“Kümeler ve kuyruk paylaşım grupları” sayfa 203
Kuyruk paylaşım gruplarını kümelerle nasıl kullanabileceğinin anlaşılması için bu konuyu kullanın.

z/OS *Paylaşılan kanallar*

IBM MQ for z/OSile paylaşılan kanalların kavramlarını ve bunların kullanımını anlamak için bu konuyu kullanın.

Bir dizi ağ ürünü, sunucu hatalarını ağdan gizlemek ya da bir dizi hak kazanan sunucu arasında gelen ağ isteklerini dengelemek için bir mekanizma sağlar. Ağ ürünleri, gelen ağ bağlantısı istekleri için bir *genel kapı* sağlar ve gelen istek, uygun sunuculardan birine bağlanarak yerine getirilebilir.

Bu ağ ürünleri aşağıdakileri içerir:

- VTAM soysal kaynakları
- SYSPLEX Distribütörü

Kanal başlatıcı, paylaşılan kuyrukların yeteneklerini kullanmak için bu ürünlerden yararlanır

İki tip paylaşılan kanal vardır: *paylaşılan gelen kanal*ve *paylaşılan giden kanal*.

- Paylaşılan gelen kanallar
- Paylaşılan giden kanallar

Kanallar hakkında daha fazla bilgi için bkz.

- Paylaşılan kanal özeti
- Paylaşılan kanal durumu

Paylaşılan gelen kanallar

Kuyruk paylaşım grubundaki her kanal başlatıcı, *soysal bir kapıda*dinleme için ek bir dinleyici görevi başlatır. Bu soysal kapı, destekleyen teknolojilerden (VTAM, TCP/IP) biri tarafından ağ tarafından kullanılabilir hale getirilir. Genel kapıya gelen ağ bağlantısı istekleri, genel kapıda dinleyen kuyruk paylaşım grubundaki (QSG) dinleyicilerden herhangi birine ağ teknolojisi tarafından dağıtılır.

Kanal başlatıcının bu ada sahip bir kanala ilişkin bir kanal tanımına erişimi varsa, kanal başlatıcısında, gelen eklemenin yönlendirildiği bir kanal başlatabilirsiniz. Bir kanal tanımlamasını bir kuyruk yöneticisine özel olacak ya da paylaşılan havuzda saklanacak ve herhangi bir yerde (genel tanımlama) saklanacak şekilde tanımlayabilirsiniz. Bu, bir kanal tanımlamasını genel tanım olarak tanımlayarak, kuyruk paylaşım grubundaki herhangi bir kanal başlatıcısında kullanılabilir kılabileceğiniz anlamına gelir.

Soysal kapı üzerinden bir kanal başlatılırken ek bir fark vardır; kanal eşitlemesi, tek bir kuyruk yöneticisiyle değil, kuyruk paylaşım grubuyla olur. Örneğin, soysal kapı üzerinden bir kanal başlatan uzak kuyruk yöneticisini düşünün. Kanal ilk kez başlatıldığında, QM1 kuyruk yöneticisinde ve ileti akışında başlatılabilir. Kanal durursa ve QM2kuyruk yöneticisinde yeniden başlatılırsa, eşitlemenin kuyruk paylaşım grubuyla birlikte olması nedeniyle, akışı olan iletilerin sayısıyla ilgili bilgiler hala doğrudur.

Herhangi bir kuyruğa ileti koymak için soysal kapı üzerinden başlatılan bir gelen kanalı kullanabilirsiniz. Uzak kuyruk yöneticisi, hedef kuyruğun paylaşılıp paylaşılmadığını bilmiyor. Hedef kuyruk paylaşılan bir kuyruksa, uzak kuyruk yöneticisi kullanılabilir herhangi bir kanal başlatıcısı üzerinden yük dengeli bir şekilde bağlanır ve iletiler paylaşılan kuyruğa yerleştirilir.

Hedef kuyruk özel bir kuyruksa, iletiler, kanalın yürürlükteki eşgörünümünün bağlı olduğu kuyruk yöneticisinin sahip olduğu özel kuyruğa yerleştirilir. *Eşlenmiş yerel kuyruklar*olarak bilinen bu ortamda, her kuyruk yöneticisinin tanımlanmış aynı özel kuyruk kümesi olmalıdır.

Kuyruk paylaşım grubu için SVRCONN kanallarının yapılandırılması

Bir kuyruk paylaşım grubundaki SVRCONN kanalları için en uygun yapılandırma, noktadan noktaya iletişim kanallarından farklı bir kapı numarası kullanan her CHINIT ' de özel dinleyiciler oluşturmaktır. Daha sonra bu dinleyici kapıları, Sanal IP adreslerini (VIPA) kullanan Sysplex Distributor gibi yeni bir iş yükü dağıtım mekanizması için 'arka uç' kaynakları olarak kullanılır. Daha sonra, dış VIPA adresi ağdaki CLNTCONN tanımlamaları için hedef adres olarak kullanılır. SVRCONN kanalı QSGDISP (GROUP) ile tanımlanabilir; bu nedenle, QSG ' deki tüm kuyruk yöneticileri için aynı tanımlama kullanılabilir. Bu yapılandırma, paylaşılan bir dinleyicinin kullanılmasını önler ve bu nedenle, istemci/sunucu kanalları için gerekli olmayan paylaşılan kanal durumunu koruyan kuyruk paylaşım grubunun performans etkisini azaltır.

Paylaşılan giden kanallar

Paylaşılan bir iletim kuyruğundan ileti alıyorsa, giden kanal paylaşılan kanal olarak kabul edilir. Paylaşılıyorsa, kuyruk paylaşım grubu düzeyinde eşitleme bilgilerini tutar. Bu, iletişim altsistemi, kanal başlatıcısı ya da kuyruk yöneticisi başarısız olursa, kanal, kuyruk paylaşım grubu içindeki farklı bir kuyruk yöneticisi ve kanal başlatıcısı yönetim ortamında yeniden başlatılabileceği anlamına gelir. Başarısız kanalların bu şekilde yeniden başlatılması, *eş kanal kurtarma*adı verilen paylaşılan kanalların bir özelliğidir.

Paylaşılan giden kanallar için iş yükü dengeleme

Giden paylaşılan kanal, belirli bir kanal başlatıcısında başlatılmasını istemediğinizi belirttiyseniz, kuyruk paylaşım grubu içindeki herhangi bir kanal başlatıcısında başlatılabilir. IBM MQ tarafından seçilen kanal başlatıcısı aşağıdaki ölçütler kullanılarak belirlenir:

- İletişim altsistemi şu anda kanal başlatıcısı tarafından kullanılabilir mi?
- Kanal başlatıcısı için bir Db2 bağlantısı var mı?
- Hangi kanal başlatıcı en düşük iş yüküne sahip? İş yükü, etkin olan ve yeniden deneyen kanalları içerir.

Paylaşılan kanal özeti

Paylaşılan kanallar aşağıdaki şekillerde özel kanallardan farklıdır:

Özel kanal
Tek bir kanal başlatıcısına bağlı.

- Giden kanal yerel bir iletim kuyruğu kullanıyor.
- Gelen kanal yerel bir kapıdan başlatıldı.
- SYSTEM.CHANNEL.SYNCQ kuyruğu.

Paylaşılan Kanal
Yüksek düzeyde kullanılabilirlik ile dengelenmiş iş yükü.

- Giden kanal paylaşılan bir iletim kuyruğu kullanıyor.
- Gelen kanal soysal bir kapıyla başlatıldı.
- SYSTEM.QSG.CHANNEL.SYNCQ kuyruğu.

Kanalı başlattığınızda, START CHANNEL komutuyla CHLDISP seçeneklerini kullanarak bir kanalın özel mi, yoksa paylaşımlı mı olduğunu belirleyebilirsiniz. Paylaşılan bir kanal, özel bir kanal gibi tetiklenerek başlatılabilir. Ancak, paylaşılan bir kanal başlatıldığında, IBM MQ iş yükü dengeleme işlemini gerçekleştirir ve kanalı, kuyruk paylaşım grubu içindeki en uygun kanal başlatıcısında başlatır. (Gerekiyorsa, belirli bir kanal başlatıcısında paylaşılan bir kanalın başlatılacağını belirtebilirsiniz.)

Paylaşılan kanal durumu

Bir kuyruk paylaşım grubundaki kanal başlatıcıları, Db2içinde paylaşılan bir kanal durumu tablosu sağlar. Bu, hangi kanalların hangi kanal başlatıcılarında etkin olduğunu kaydeder. Bir kanal başlatıcı ya da iletişim sistemi hatası varsa, paylaşılan kanal durumu çizelgesi kullanılır. Kuyruk paylaşım grubundaki farklı bir kanal başlatıcısında hangi kanalların yeniden başlatılması gerektiğini gösterir.

z/OS *Grup içi kuyruğa alma*

Grup içi kuyruğa alma, bir kuyruk paylaşım grubundaki kuyruk yöneticileri arasında iletilerin aktarılmasına olanak sağlar.

Kanal tanımlamadan kuyruk paylaşım grubundaki kuyruk yöneticileri arasında hızlı ileti aktarımı gerçekleştirebilirsiniz. Bu, SYSTEM.QSG.TRANSMIT.QUEUE, paylaşılan bir iletim kuyruğu. Kuyruk paylaşım grubundaki her kuyruk yöneticisi, kuyruk yöneticisi için yazılmış olan bu kuyruğa ulaşmak için iletilerin gelmesini bekleyen grup içi kuyruğa alma aracısı (intra-group queuing Agent) adlı bir görevi başlatır. Böyle bir ileti algılandığında, kuyruktan kaldırılır ve doğru hedef kuyruğa yerleştirilir.

Standart ad çözme kuralları kullanılır, ancak, grup içi kuyruğa alma (IGQ) etkinleştirilmişse ve hedef kuyruk yöneticisi kuyruk paylaşım grubu içindeyse, SYSTEM.QSG.TRANSMIT.QUEUE , ileti, iletim kuyruğu ve kanal kullanmak yerine doğru hedef kuyruk yöneticisine aktarmak için kullanılır.

Bir kuyruk yöneticisi özniteliği aracılığıyla grup içi kuyruğa alma işlemini geçerli kılmanızı sağlar. Grup içi kuyruklama, kalıcı olmayan iletileri eşitleme noktası dışında ve kalıcı iletiler eşitleme noktası içinde

taşır. Hedef kuyruğa ileti gönderimi sırasında bir sorun bulursa, grup içi kuyruklama onları ölüme mektup kuyruğuna yerleştirmeyi dener. Ölü-harf kuyruğu dolu ya da tanımsız ise, kalıcı olmayan iletiler atılır, ancak kalıcı iletiler yedeklenir ve SYSTEM.QSG.TRANSMIT.QUEUE(Kuyruk) ve IGQ aracısı, iletileri başarılı oluncaya kadar teslim etmeye çalışır.

Kuyruk paylaşım grubundaki farklı bir kuyruk yöneticisinde kuyruğa yollanan bir iletiyi alan gelen paylaşılan kanal, iletiyi doğru hedefe *sekme noktası* yapmak için grup içi kuyruğa alma işlemi kullanabilir.

Hedef kuyruk hedef kuyruk yöneticisine aktarılmakta olan ileti yerine, hedef kuyruk paylaşılan bir kuyruksa, yerel kuyruk yöneticisinin hedef kuyruğa doğrudan bir ileti göndermesini istemeniz gerekebilir. Bu durumu denetlemek için SQQMNAME kuyruk yöneticisi özniteliğini kullanabilirsiniz. SQQMNAME değerini kullanacak şekilde ayarladıysanız, MQOPEN komutu ObjectQMgrAdı tarafından belirtilen kuyruk yöneticisesinde gerçekleştirilir. Ancak, hedef kuyruk paylaşılan bir kuyruksa ve SQQMNAME değerini IGNORE olarak ayarladıysanız ve ObjectQMgrAdı, kuyruk paylaşım grubundaki başka bir kuyruk yöneticisiniyse, yerel kuyruk yöneticisinde paylaşılan kuyruk açılır. Yerel kuyruk yöneticisi hedef kuyruğu açamıyorsa ya da kuyruğa bir ileti koyarsa, ileti belirtilen ObjectQMgrAdına IGQ ya da MQ kanalı yoluyla aktarılır.

Grup içi kuyruklama (IGQ) büyük iletileri, en büyük 100 MB ' lik *eksi* iletim kuyruğu üstbilgisinin uzunluğunu destekler.

Bu özelliği kullanırsanız, kullanıcıların kuyruk paylaşım grubundaki her kuyruk yöneticisinde bulunan kuyruklara aynı erişime sahip olması gerekir.

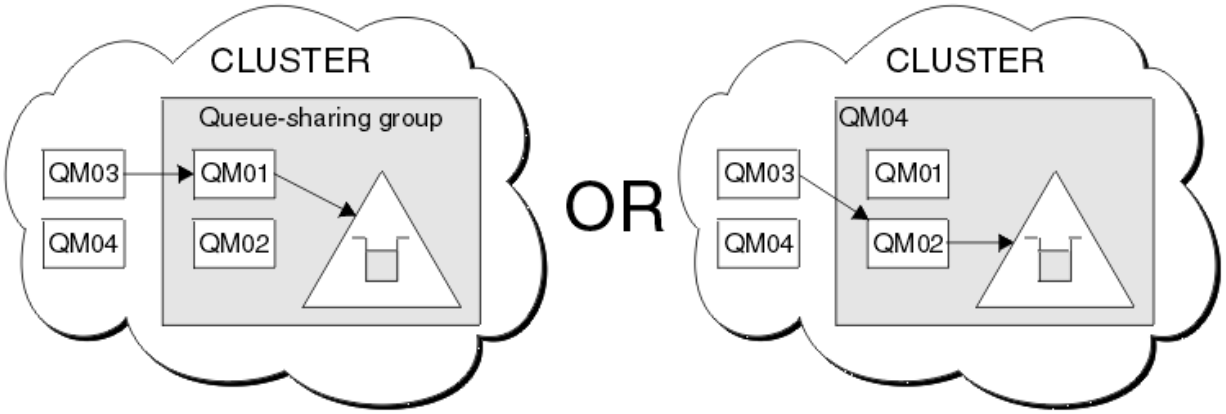
z/OS *Kümeler ve kuyruk paylaşım grupları*

Kuyruk paylaşım gruplarını kümelerle nasıl kullanabileceğinin anlaşılması için bu konuyu kullanın.

Paylaşılan kuyruklarınızı tek bir tanımlamadaki bir kümede kullanılabilir kılabilir. Bunu yapmak için, paylaşılan kuyruğu tanımladığınızda kümenin adını belirtin.

Ağdaki kullanıcılar, kuyruk paylaşım grubundaki her bir kuyruk yöneticisi tarafından barındırılmakta olan paylaşılan kuyruğu görürler (paylaşılan kuyruk, kuyruk paylaşım grubunun barındırıldığı gibi tanıtılmaz). İstemciler, iletileri aynı paylaşılan kuyruğa yerleştirmek için kuyruk paylaşım grubunun herhangi bir üyesiyle oturum başlatabilir.

Şekil 67 sayfa 203 , bir kümenin üyelerinin, kuyruk paylaşım grubunun herhangi bir üyesi aracılığıyla paylaşılan bir kuyruğa nasıl erişebileceğini gösterir.



Şekil 67. Bir kümenin parçası olarak bir kuyruk paylaşım grubu

z/OS Paylaşılan kuyruklarla iş yükü dağılımını etkileme

Bir kuyruk paylaşım grubundaki paylaşılan kuyruklarla iş yükü dağılımını etkileyen faktörleri anlamak için bu konuyu kullanın.

IBM MQ paylaşılan kuyruklar için iş yükü dengelemesi sağlamaz. Ancak, bir kuyruk paylaşım grubunda (QSG) iş yükü dağılımı *çekme tabanlı bir moda*etkisinde olabilir. Kuyruk yöneticisi hizmetlerinin bir kuyruğun (paylaşılan bir kuyruğa yazılan bir iletiyi alan) seçimi, kuyruk paylaşım grubundaki her kuyruk yöneticisinin kullanılabilir işleme kapasitelerinden ve sistem şebekesi genelinde tanımlanan iş yükü yönetimi hedeflerinden etkilenir.

Ancak, bir iletinin MQPUT değerini gerçekleştiren kuyruk yöneticisinin, iletiyi hangi kuyruk yöneticisinin alacağına karar verme konusunda büyük bir etkisi de olabilir.

Yerel kuyruk yöneticisi, MQGET işlemini gerçekleştirmeye daha yatkın

Bir MQPUT gerçekleştiren bir uygulama için, yerel kuyruk yöneticisinin uygulamanın bağlı olduğu kuyruk yöneticisi olduğu söylenmektedir.

Bir uygulama alma uygulaması adına bir MQGET işlemi gerçekleştirerek, MQPUT 'un bir MQPUT' ye ilişkin MQPUT ' u tam olarak hangi kuyruk yöneticisi hizmetleri tarafından etkilenir: aşağıdaki noktalar etkilenir.

Bir ileti boş bir paylaşılan kuyruğa konduğunda, yerel kuyruk yöneticisi genellikle kuyruk paylaşım grubundaki diğer kuyruk yöneticilerinden herhangi birinin bildirilmeden önce postalanır. Yerel kuyruk yöneticisi iletiyi işlemek için bir konumdaysa, QSG ' deki başka bir kuyruk yöneticisinden önce, bağlaşım tesisinden (CF) bir liste geçiş bildirimi alır. (Bir liste geçişi bildirimi, paylaşılan kuyruğun boş olan durumu boş olmayan bir duruma getirdiğine ilişkin bir bildirimdir.)

Bu durumda, olası senaryolar aşağıdaki gibidir:

1. MQPUT, kalıcı olmayan iletiyi eşitleme noktasından ve *hızlı bir şekilde getter beklemeye al*değerine koydu.

   Kuyruğa ilişkin yerel kuyruk yöneticisiyle ilgili bir *Beklemeyle MQGET* bulunan bir uygulama varsa, iletinin MQPUT değeri doğrudan uygulamanın arabelle§ine ge‡irilir ve kuyru§a yazilmaz. Bu, paylaşılan ve paylaşılmayan kuyruklar için geçerlidir. Bu özellik genellikle *hızlı bekleme getter takması* mekanizmasını içerir. Paylaşılan kuyruklar durumunda, kuyruğun boş olmaması için boş bir geçiş olmadığı için, QSG ' de başka bir kuyruk yöneticisi bilgilendirilmedi. Bu, örneğin, bu kuyruk yöneticisinin bu uygulamadan tüm yerleştirmeleri hizmet edebilmesi ve başka hiçbir uygulamanın kuyruğa ileti koymamasını sağlamak anlamına gelir; daha sonra, kuyruk paylaşım grubunda başka bir kuyruk yöneticisi bu kuyruğun boşaltılmasına yardımcı olmaz. Ancak, yerel kuyruk yöneticisinde bekleyen bir MQGET işlemi yoksa ve paylaşılan kuyruğa bir ileti konursa, CF, kuyruk paylaşım grubundaki diğer kuyruk yöneticilerine, liste geçişleri bildirimlerine ilişkin kurallarına göre bildirim gönderecektir.
2. Kalıcı ya da eşitleme noktası iletisi MQPUT.

   Bu durumda, yerel kuyruk yöneticisinde *MQGET with wait* içeren bir uygulama varsa, ileti paylaşılan kuyruğa konursa ve CF, kuyruk paylaşım grubundaki diğer kuyruk yöneticilerine liste geçişleri bildirimlerine ilişkin kurallarına göre bildirir. Ancak, yerel kuyruk yöneticisi, CF ' den bir geçiş bildirimi beklemez, ancak herhangi bir yerel *MQGET with wait* özelliğini kabul eder ve kuyruk paylaşım grubundaki diğer herhangi bir kuyruk yöneticisinin bir CF bildirimine yanıt verebilmesi için önce bu iletinin alma işlemini genellikle uygulama adına gerçekleştirir. Bu, yerel kuyruk yöneticisinin ne kadar meşgul olduğuna bağlı olarak değişir. Tersi durumda, ileti CF tarafından boş kuyruğa varılması nedeniyle bildirilen herhangi bir kuyruk yöneticisi önce alma işlemi için hizmet almaya çalışır. Yanıt veren ilk kuyruk yöneticisi, yeni iletiyi işleme sokardı.
3. Son olarak, kuyruğun iletileri boşaltılmamışsa, CF ' nin kuyrukta boş olmayan bir durum değişikliği bildirimi göndermesi durumunda, bağlı olan tüm kuyruk yöneticilerinin kuyruğun işlenmesine yardımcı olması için bir fırsat bulunur. Bu olayda, iş yükünün *temelli olarak çekeceği*olduğu söyleniyor.

Bu tasarım, tamamen çekme tabanlı bir iş yükü dağılımı üzerinde iyileştirilmiş performansa olanak sağlar. Hedef, CF ' de tutulan kuyruklar tarafından sunulan yüksek kullanılabilirlik avantajından yararlanarak, mümkün olduğunca kuyruk yöneticisine başvurarak MQGET işlemini gerçekleştirmeyi ve ileti iş yükünün mümkün olduğunca verimli bir şekilde işlenmesini sağlar.

Alternatif yaklaşımlar, iş yükünün dengesinin daha önce açıklanan performans geliştirmelerinden daha önemli olduğu durumlarda benimsenebilir. Örneğin, alma uygulamalarından hiçbirinin, koyma

uygulamasının bağlı olduğu kuyruk yöneticisine bağlı olmamasını sağlayın. Bu tasarımın kullanılması, tüm iletilerin kuyruğa konması ve QSG ' deki tüm kuyruk yöneticilerinin, bu tür geçişleri işlemek için kullanılan CF algoritmasına uygun bir şekilde boş olmayan bir kuyruktan boş duruma geçmesiyle ilgili bilgilendirilmeleri. Ayrıca, *fast put to bekleme getter* mekanizması geçerli değildir.

z/OS Paylaşılan kuyruklar ve kuyruk paylaşım grupları hakkında daha fazla bilgi bulabileceğiniz yerler

IBM MQ for z/OS ' in paylaşılan kuyrukları ve kuyruk paylaşım gruplarını nasıl kullandığı hakkında daha fazla bilgi bulmak için bu konudaki tabloyu kullanın.

Çizelge 18. Paylaşılan kuyruklar ve kuyruk paylaşım grupları hakkında daha fazla bilgi bulabileceğiniz yerler

Konu	**Nereye bakılacağı**
Kuyruk paylaşım grubu kurtarma	“Recovery and restart on z/OS” sayfa 244
Kuyruk paylaşım grubu güvenliği	“IBM MQ for z/OSiçindeki güvenlik kavramları” sayfa 260
Özel ve genel nesne tanımlamaları Komutlar farklı kuyruğa yönlendiriliyor Yöneticiler	Komut verme
Bağlaşım tesisinizi planlama Ortam	Bağlantı olanağı kaynaklarının tanımlanması
SMDS ortamınızın planlanması	Paylaşılan ileti veri kümesi (SMDS) ortamınızı planlama
Planlama Db2 Ortam	Db2 ortamınızın planlanması
Paylaşılan kuyruklarınızın ayarlanması Sistem değiştirbeleri	“Paylaşılan kuyruklar ve kuyruk paylaşım grupları” sayfa 160
Yardımcı programlar Kuyrukların geçirilmesi	IBM MQ Yardımcı Programlarının Kullanılması
Konsol iletileri	IBM MQ for z/OS İletileri
MQSC komutları	MQSC komutları
IBM MQ Kümeler	Kuyruk yöneticisi kümesinin yapılandırılması
IBM MQ dağıtımlı kuyruğa alma Kanal adları	Dağıtımlı kuyruk yönetimine giriş
Uygulamalar yazılıyor	Uygulama tasarımına genel bakış
MQCONNX çağrısı	MQCONNX

z/OS Grup içi kuyruğa alma

Bu bölümde grup içi kuyruğa alma, z/OS platformuna özgü bir IBM MQ for z/OS işlevi ele alınmıştır. Bu işlev yalnızca, bir kuyruk paylaşım grubuna tanımlı kuyruk yöneticilerine kullanılabilir.

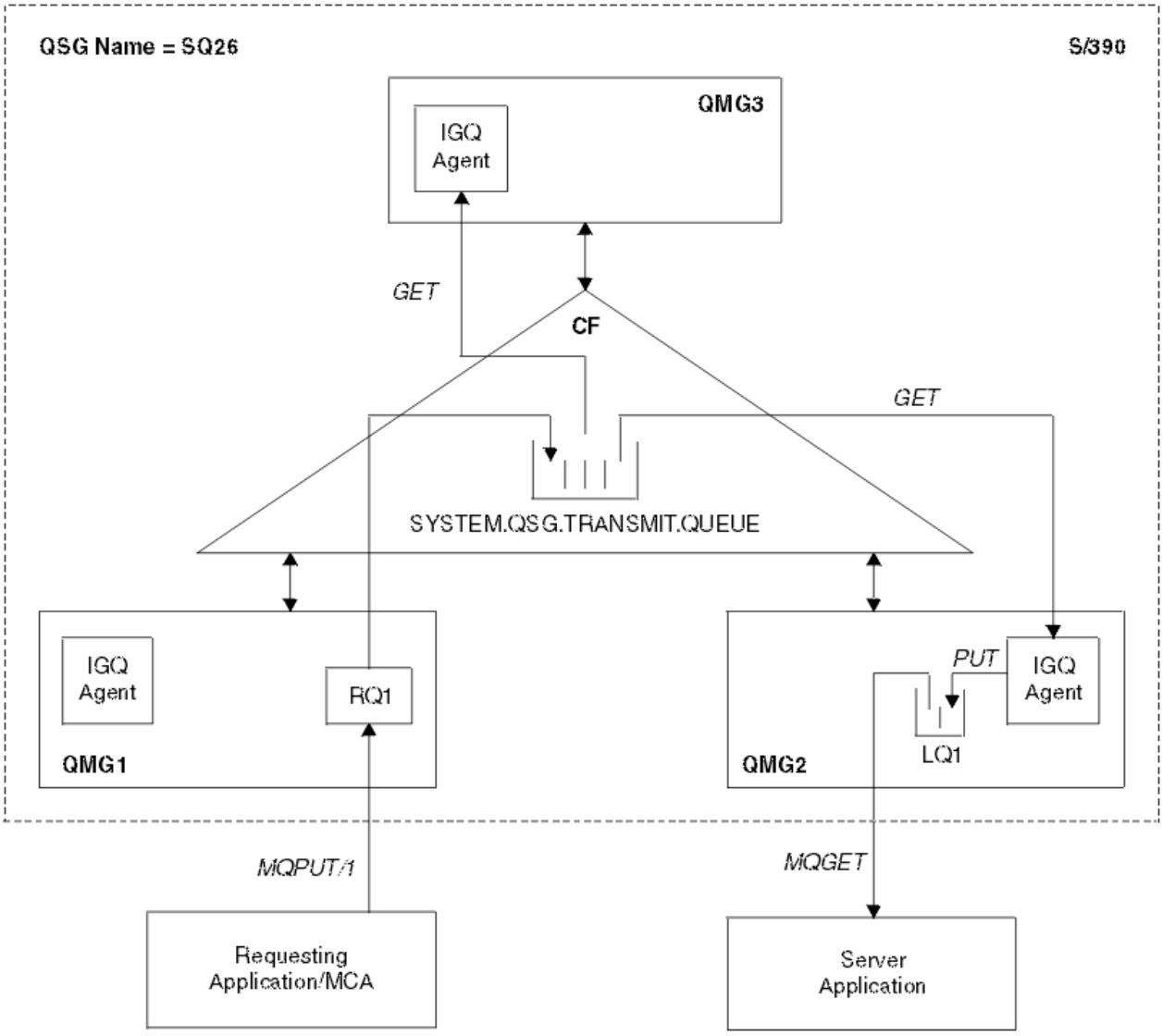
Kuyruk paylaşım gruplarıyla ilgili bilgi için bkz. “Paylaşılan kuyruklar ve kuyruk paylaşım grupları” sayfa 160.

z/OS Grup içi kuyruklama kavramları

Grup içi kuyruklama, bir kuyruk paylaşım grubu içindeki uzak kuyruk yöneticilerindeki kuyruklara daha verimli, daha verimli, küçük iletiler sunmak için kullanılabilir.

Grup içi kuyruğa alma (IGQ), kuyruk yöneticileri arasında kanal tanımlama gereksinimi olmadan, bir kuyruk paylaşım grubundaki (QSG) kuyruk yöneticileri arasında potansiyel olarak hızlı ve daha az maliyetli küçük ileti aktarımını etkileyebilir.

Aşağıdaki çizge, grup içi kuyruğa alma örneğinin tipik bir örneğini göstermektedir.



Şekil 68. Grup içi kuyruğa alma örneği

Çizge şunları gösterir:

- Üç kuyruk yöneticisi (QMG1, QMG2ve QMG3) üzerinde çalışan IGQ aracıları, SQ26adlı bir kuyruk paylaşım grubuna tanımlanır.
- Paylaşılan iletim kuyruğu SYSTEM.QSG.TRANSMIT.QUEUE (bağlaşım olanağı) içinde tanımlı olan QUEUE (kuyruk).
- A remote queue definition that is defined in queue manager QMG1.
- A local queue that is defined in queue manager QMG2.
- İstekte bulunan bir uygulama (bu uygulama, QMG1kuyruk yöneticisine bağlı bir Message Channel Agent (MCA) olabilir).
- Kuyruk yöneticisine (QMG2) bağlı bir sunucu uygulaması.
- SYSTEM.QSG.TRANSMIT.QUEUE.

Grup içi kuyruklama ve grup içi kuyruğa alma aracısı

Kuyruk yöneticisi başlatma işlemi sırasında bir IGQ aracısı başlatılır. Uygulamalar açıldığında ve uzak kuyruklara ileti yerleştirirken, yerel kuyruk yöneticisi ileti aktarımı için grup içi kuyruğa alma kullanılıp kullanılmayacağını belirler. Grup içi kuyruğa alma işlemi kullanılacaksa, yerel kuyruk yöneticisi iletiyi SYSTEM.QSG.TRANSMIT.QUEUE. Hedef uzak kuyruk yöneticisininde IGQ aracısı iletiyi alır ve hedef kuyruğa yerleştirir.

z/OS Grup içi kuyruğa alma terminolojisi

terminolojinin açıklamaları: grup içi queuing, grup içi queuing tarafından kullanılmak üzere paylaşılan iletim kuyruğu ve grup içi queuing ajanda.

Grup içi kuyruğa alma

Grup içi kuyruğa alma, kanal tanımlama gereksinimi olmadan kuyruk paylaşım grubundaki kuyruk yöneticileri arasında hızlı ve daha az maliyetli ileti aktarımını etkileyebilir.

Grup içi kuyruğa alma işlemi tarafından kullanılmak üzere paylaşılan iletim kuyruğu

Her kuyruk paylaşım grubunun, SYSTEM.QSG.TRANSMIT.QUEUE (kuyruk içi kuyruklama) tarafından kullanılmak üzere QUEUE (kuyruk) Grup içi kuyruğa alma geçerli kılındıysa, SYSTEM.QSG.TRANSMIT.QUEUE görüntülenir. Uygulamalar (Message Channel Agents (MCA ' lar) da içinde olmak üzere), iletileri uzak bir kuyruğa koyduğunda, yerel kuyruk yöneticisi iletilerin hızlı aktarıma uygunluğu ve bunları SYSTEM.QSG.TRANSMIT.QUEUE.

Grup içi kuyruklama aracısı

IGQ aracısı, SYSTEM.QSG.TRANSMIT.QUEUE. IGQ aracısı, bu kuyruktan uygun iletileri alır ve bunları hedef kuyruklara teslim eder.

Her kuyruk yöneticisi için IGQ aracısı her zaman başlatılır; kuyruk içi kuyruklama, kendi iç işlemesi için kuyruk yöneticisinin kendisi tarafından kullanılır.

z/OS Grup içi kuyruğa alma yararları

Grup içi kuyruğa alma olanağının yararları şunlardır: sistem tanımlarının azaltılması, sistem yönetiminin azaltılması, başarımın artırılması, geçişin desteklenmesi ve kuyruk paylaşım grubundaki kuyruk yöneticileri arasında çok sekmelerin olması, iletilerin sağlanmasını sağlar.

grup içi queuing ' in faydaları şunlardır:

Azaltılmış sistem tanımlamaları
Grup içi kuyruklama, kuyruk paylaşım grubundaki kuyruk yöneticileri arasında kanal tanımlama gereksinmesini ortadan kaldırır.

Azaltılmış sistem denetimi

Kuyruk paylaşım grubundaki kuyruk yöneticileri arasında tanımlı kanal olmadığı için, kanal denetimi için herhangi bir gereksinim yoktur.

Daha iyi performans

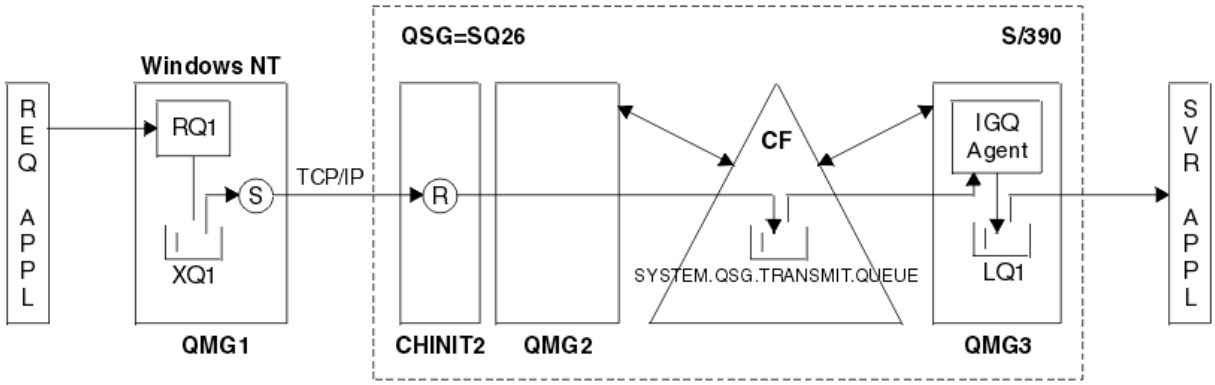
Bir iletinin hedef kuyruğa teslim edilmesi için gereken tek bir IGQ aracı olduğundan (iki ara gönderici ve alıcı aracısı yerine), grup içi kuyruğa alma kullanan iletilerin teslim edilmesi, kanalları kullanan iletilerin tesliminden daha az maliyetli olabilir. Grup içi kuyruğa alma işlemi, gönderme bileşeninin gereksinmesi kaldırıldığı için yalnızca bir giriş bileşenidir. Bu kaydetme işlemi, yerel kuyruk yöneticisinde yapılan put işlemi tamamlandıktan sonra ve ileti eşitleme noktası kapsamı olarak sunulan iletiler durumunda, ileti hedef kuyruk yöneticisinde hedef kuyruk yöneticisinde bulunan IGQ aracısının iletilmesinden kaynaklanır ve kesinleştirilir.

Geçişi destekler

Kuyruk paylaşım grubunun dışındaki uygulamalar, kuyruk paylaşım grubundaki herhangi bir kuyruk yöneticisinde bulunan bir kuyruğa ileti gönderirken, yalnızca kuyruk paylaşım grubundaki belirli bir kuyruk yöneticisine bağlanılıyor. Bunun nedeni, bir günlük nesnesi kanalına gelen iletilerin, uzak kuyruk yöneticilikteki bir kuyruğa yollanması, grup içi kuyruğa alma kullanılarak hedef kuyruğa saydam bir şekilde gönderilebilmesinden kaynaklanır. Bu olanak, kuyruk paylaşım grubu dışındaki tüm sistemlerin değiştirilmesi gerekmeksizin, uygulamaların kuyruk paylaşım grubu arasında konuşlandırılmasına olanak sağlar.

Tipik bir yapılanış aşağıdaki çizge tarafından gösterilir:

- A requesting application connected to queue manager QMG1 needs to send a message to a local queue on queue manager QMG3.
- Queue manager QMG1 is connected only to queue manager QMG2.
- Queue managers QMG2 and QMG3, which were previously connected using channels, are now members of queue sharing group SQ26.



Şekil 69. Geçiş desteği örneği

İşlemlerin akışı aşağıdaki gibidir:

1. The requesting application puts a message, destined for local queue LQ1 at remote queue manager QMG3, on to remote queue definition RQ1.
2. Windows NT iş istasyonunda çalışan kuyruk yöneticisi QMG1, iletiyi iletim kuyruğuna (XQ1) yerleştirir.
3. Sender MCA (S) on QM1 transmits the message, using TCP/IP, to the receiver MCA (R) on channel initiator CHINIT2.
4. Receiver MCA (R) on channel initiator CHINIT2 places the message on to the shared transmission queue SYSTEM.QSG.TRANSMIT.QUEUE.
5. IGQ agent on queue manager QMG3 retrieves the message from the SYSTEM.QSG.TRANSMIT.QUEUE and places it on to the target local queue LQ1.
6. Sunucu uygulaması iletiyi hedef yerel kuyruktan alır ve bu iletiyi işler.

Bir kuyruk paylaşım grubundaki kuyruk yöneticileri arasında çoklu atlama sırasında iletilerin teslimi
Migration migration (Geçişi destekle) içindeki önceki çizge, bir kuyruk paylaşım grubundaki kuyruk yöneticileri arasında çok seksli geçiş yaparken iletilerin sağlananını da gösterir. Kuyruk paylaşım grubundaki bir kuyruk yöneticisine gelen, ancak kuyruk paylaşım grubundaki başka bir kuyruk yöneticisinde bulunan bir kuyruğa yollanan iletiler, hedef kuyruk yöneticisinde hedef kuyruğa kolayca iletilebilir; grup içi kuyruğa alma işlemi kullanılarak bu iletiler kolayca iletilebilir.

z/OS Grup içi kuyruğa alma sınırlamaları

Grup içi kuyruğa alma sınırlamaları şunlardır: grup içi kuyruğa alma kullanılarak aktarıma uygun iletiler, kuyruk yöneticisi başına grup içi kuyruğa alma aracısı sayısı, grup içi kuyruğa alma aracısının başlatılması ve durdurulması.

Bu konuda, grup içi kuyruğa alma sınırlamalarıyla ilgili bilgiler yer alır.

Grup içi kuyruğa alma kullanılarak aktarıma uygun iletiler
Grup içi kuyruğa alma, bağlaşım tesisinde (CF) tanımlanmış paylaşılan bir iletim kuyruğu kullandığından, grup içi kuyruğa alma işlemi, paylaşılan kuyruklar için desteklenen ileti uzunluğu üst sınırı eksi iletim kuyruğu üstbilgisinin (MQXQH) uzunluğu için ileti teslim etmek üzere sınırlandırılır.

Kuyruk yöneticisi başına grup içi kuyruğa alma aracıları sayısı
Kuyruk paylaşım grubunda kuyruk yöneticisi başına tek bir IGQ aracısına başlanır.

Grup içi kuyruğa alma aracısının başlatılması ve durdurulması
Kuyruk yöneticisi kullanıma hazırlanması sırasında IGQ aracısı başlatılır ve kuyruk yöneticisi sona erdirilirse sonlandırılır. Uzun bir çalışma, kendini kurtarma (olağandışı sonlandırma durumunda), görev olarak tasarlanmıştır. SYSTEM.QSG.TRANSMIT.QUEUE (örneğin, bu kuyruk engellenirse) IGQ aracısı yeniden denenmeye devam eder. IGQ aracısı, kuyruk yöneticisi hala etkin durumdayken, aracının olağan şekilde sonlandırılması sonucu ortaya çıkan bir hatayla karşılaşırsa, bir ALTER QMGR IGQ (ENABLED) komutu yayınlanarak yeniden başlatılabilir. Bu komut, kuyruk yöneticisinin geri dönüştürülmesine gerek duyacak şekilde önlenir.

IGQ kuyruk yöneticisi özniteliğinin IGQ ENABLED ya da DISABLE değeri belirleniyor
Kuyruk yöneticisi özniteliği IGQ ENABLED ya da DISABLE değerine ayarlıysa, varolan nesne tanıtıcıları geçersiz kılınabilir; neden kodu MQRC_OBJECT_CHANGED neden kodu. Ek bilgi için bkz. Grup içi kuyruğa alımıyla çalışmaya başlama .

z/OS Grup içi kuyruklama ile çalışmaya başlama

Bu konuda açıklandığı gibi grup içi kuyruklama özelliğini etkinleştirebilir, geçersiz kılabilir ve kullanabilirsiniz.

Grup içi kuyruğa alma etkinleştirmesi
Kuyruk yöneticilerinizde grup içi kuyruğa alma işlevini etkinleştirmek için aşağıdaki işlemi yapmanız gerekir:

- SYSTEM.QSG.TRANSMIT.QUEUEadlı bir paylaşılan iletim kuyruğu tanımlayın. Bu kuyruğun tanımı thlqual.SCSQPROC(CSQ4INSS) içinde, kuyruk paylaşım grupları için SYSTEM nesneleri için CSQINP2 örneğinde bulunabilir. Bu kuyruk, thlqual.SCSQPROC(CSQ4INSS) içinde belirtildiği gibi, grup içi kuyruklama için doğru şekilde çalışmak üzere doğru özniteliklerle tanımlanmalıdır.
- IGQ aracısı her zaman kuyruk yöneticisi kullanıma hazırlamaya başlandığından, gelen ileti işleme için grup içi kuyruğa alma her zaman kullanılabilir olur. IGQ aracısı, SYSTEM.QSG.TRANSMIT.QUEUE. Ancak, giden işleme için grup içi kuyruğa alma işlevini etkinleştirmek için, IGQ kuyruk yöneticisi özniteliği ENABLED olarak ayarlanmış olmalıdır.

  Önemli: Kuyruk yöneticisi özniteliği IGQ ENABLED olarak ayarlandıysa, varolan nesne tanıtıcıları geçersiz kılınabilir; neden kodu MQRC_OBJECT_CHANGED neden kodu. Ek bilgi için “Grup içi kuyruğa alma özel özellikleri” sayfa 217 başlıklı konuya bakın. Bu neden koduna ilişkin 'Programcı yanıtı' kısmında açıklandığı gibi, uygulamaların bu durumla başa çıkabilmek için kodlanmalıdır (ayrıntılı bilgi için 2041 (07F9) (RC2041): MQRC_OBJECT_CHANGED).

Ayrıca, IGQ, başlatma sırasında başlayan ve sona erdirme ile sona erdiren, uzun bir çalışma ve kendini kurtarma görevi olarak tasarlandığı için, ek bilgi için “Grup içi kuyruğa alma sınırlamaları” sayfa 209 konusuna bakın.

Grup içi kuyruklama devre dışı bırakılıyor

Giden ileti aktarımı için grup içi kuyruğa alma işlemini geçersiz kılmak için, IGQ kuyruk yöneticisi özniteliğini DISABLE olarak ayarlayın. Belirli bir kuyruk yöneticisi için grup içi kuyruğa alma geçersiz kılındıysa, o kuyruk yöneticindeki IGQ aracısı, SYSTEM.QSG.TRANSMIT.QUEUE (kuyruk yöneticisi) tarafından, giden aktarım için grup içi kuyruğa alma etkinleştirilmiş olan bir kuyruk yöneticisi tarafından kullanılabilir.

Önemli: Kuyruk yöneticisi özniteliği IGQ ENABLED olarak ayarlandıysa, varolan nesne tanıtıcıları geçersiz kılınabilir; neden kodu MQRC_OBJECT_CHANGED neden kodu. Ek bilgi için “Grup içi kuyruğa alma özel özellikleri” sayfa 217 başlıklı konuya bakın. Bu neden koduna ilişkin 'Programcı yanıtı' kısmında açıklandığı gibi, uygulamaların bu durumla başa çıkabilmek için kodlanmalıdır (ayrıntılı bilgi için 2041 (07F9) (RC2041): MQRC_OBJECT_CHANGED).

Ayrıca, IGQ, başlatma sırasında başlayan ve sona erdirme ile sona erdiren, uzun bir çalışma ve kendini kurtarma görevi olarak tasarlandığı için, ek bilgi için “Grup içi kuyruğa alma sınırlamaları” sayfa 209 konusuna bakın.

Grup içi kuyruklama kullanılması

Grup içi kuyruğa alma etkinleştirildikten sonra, kullanılabilir durumda ve kuyruk yöneticisi tarafından mümkün olduğunda bunu kullanır. Yani, bir uygulama uzak kuyruk tanımlamasına, tam olarak nitelenmiş bir uzak kuyruğa ya da bir küme kuyruğuna ileti koyduğunda, kuyruk yöneticisi, iletinin grup içi kuyruğa alma kullanılarak teslim edilme uygun olup olmadığını ve varsa, iletiyi SYSTEM.QSG.TRANSMIT.QUEUE' a göndereceğini belirler. Kuyruk yöneticisinin SYSTEM.QSG.TRANSMIT.QUEUE, başka bir iletim kuyruğunu tercih ediyor.

z/OS Grup içi kuyruğa alma yapılandırmaları

Tipik grup içi kuyruğa alma yapılanışına ek olarak, diğer yapılanışlar da mümkündür.

Şekil 68 sayfa 206 , tipik yapılandırmayı açıklar.

İlgili kavramlar

“Grup içi kuyruklama ile dağıtımlı kuyruğa alma (birden çok teslim yolu)” sayfa 210
Kısa iletileri işleyen uygulamalar için, yalnızca bir kuyruk paylaşım grubundaki kuyruk yöneticileri arasında ileti almak için grup içi kuyruğa alma işleminin yapılandırılması uygulanabilir.

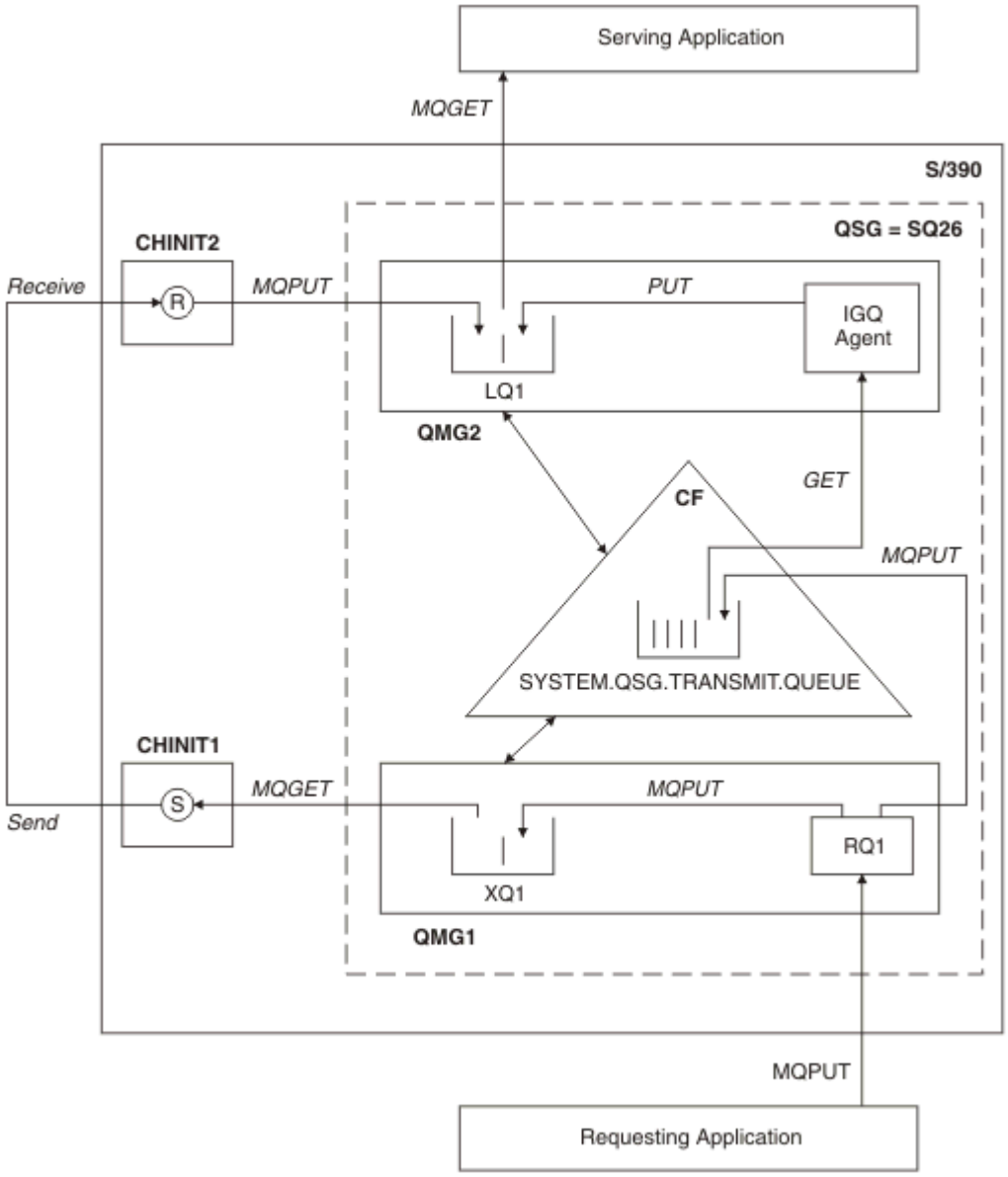
“Grup içi kuyruğa alma ile kümeleme (birden çok teslim yolu)” sayfa 212
Kuyruk yöneticilerini, bir kuyruk paylaşım grubunda olduğu kadar bir kümede olacak şekilde yapılandırmak mümkün.

“Kümeleme, grup içi kuyruğa alma ve dağıtılmış kuyruklama” sayfa 214
Bir kümenin yanı sıra bir kuyruk paylaşım grubu olan ve gönderen/alıcı kanal çifti kullanılarak dağıtımlı bir kuyruk yöneticisine bağlı bir kuyruk yöneticisi yapılandırmak mümkündür.

z/OS *Grup içi kuyruklama ile dağıtımlı kuyruğa alma (birden çok teslim yolu)*

Kısa iletileri işleyen uygulamalar için, yalnızca bir kuyruk paylaşım grubundaki kuyruk yöneticileri arasında ileti almak için grup içi kuyruğa alma işleminin yapılandırılması uygulanabilir.

Kanal iletişimleri üzerinde kuyruğa alma içi kuyruğa alma seçeneği, CFSTRUCT tip düzeyi tarafından denetlenebilir. (4 ya da 5 yerine 3). SYSTEM.QSQ.TRANSMIT.QUEUE.



Şekil 70. Örnek bir yapılandırma

Açma/Koy işleme

1. It is important to note that when the requesting application opens remote queue RQ1, name resolution occurs for both the non-shared transmission queue XQ1 and the shared transmission queue SYSTEM.QSG.TRANSMIT.QUEUE.
2. İstekte bulunan uygulama uzak kuyruğa bir ileti yerleştirdiğinde, kuyruk yöneticisine giden aktarma için grup içi kuyruğa alma özelliğinin etkinleştirilip etkinleştirilmediğine ve ileti özelliklerine ilişkin ileti, ileti iletim kuyruğuna (XQ1) ya da iletim kuyruğuna (SYSTEM.QSG.TRANSMIT.QUEUE. Kuyruk yöneticisi, tüm büyük iletileri XQ1iletim kuyruğuna ve tüm küçük iletileri SYSTEM.QSG.TRANSMIT.QUEUEiletim kuyruğuna yerleştirir.
3. İletim kuyruğu XQ1 dolduysa ya da kullanılamazsa, büyük iletiler için koyma istekleri uygun bir dönüş ve neden koduyla zamanuyumlu olarak başarısız olur. Ancak, küçük iletilere ilişkin istekler başarılı olmaya devam eder ve iletim kuyruğu SYSTEM.QSG.TRANSMIT.QUEUE.
4. İletim kuyruğu SYSTEM.QSG.TRANSMIT.QUEUE tam ya da yerleştirilemiyor, küçük iletiler için koyma istekleri uygun bir iade ve neden koduyla zamanuyumlu olarak başarısız oldu. However, put requests

for large messages continue to succeed and are placed on transmission queue XQ1. Bu durumda, küçük iletileri bir iletim kuyruğuna yerleştirmek için girişimde bulunulmaz.

Büyük iletiler için akış

1. İstekte bulunan uygulama, büyük iletileri RQ1uzak kuyruğuna yerleştiriyor.
2. Kuyruk yöneticisi QMG1 , iletileri XQ1iletim kuyruğuna yerleştiriyor.
3. Sender MCA (S) on queue manager QMG1 retrieves the messages from transmission queue XQ1 and sends them to queue manager QMG2.
4. Receiver MCA (R) on queue manager QMG2 receives the messages and places them on to destination queue LQ1.
5. Hizmet veren uygulama, LQ1kuyruğundan gelen iletileri alır ve işler.

Küçük iletiler için akış

1. İstekte bulunan uygulama, küçük iletileri RQ1uzak kuyruğuna yerleştiriyor.
2. Kuyruk yöneticisi QMG1 , iletileri SYSTEM.QSG.TRANSMIT.QUEUEiletim kuyruğuna yerleştiriyor.
3. IGQ on kuyruk yöneticisi QMG2 , iletileri alır ve hedef kuyruğun LQ1' e yerleştirir.
4. The serving application retrieves the messages from queue LQ1.

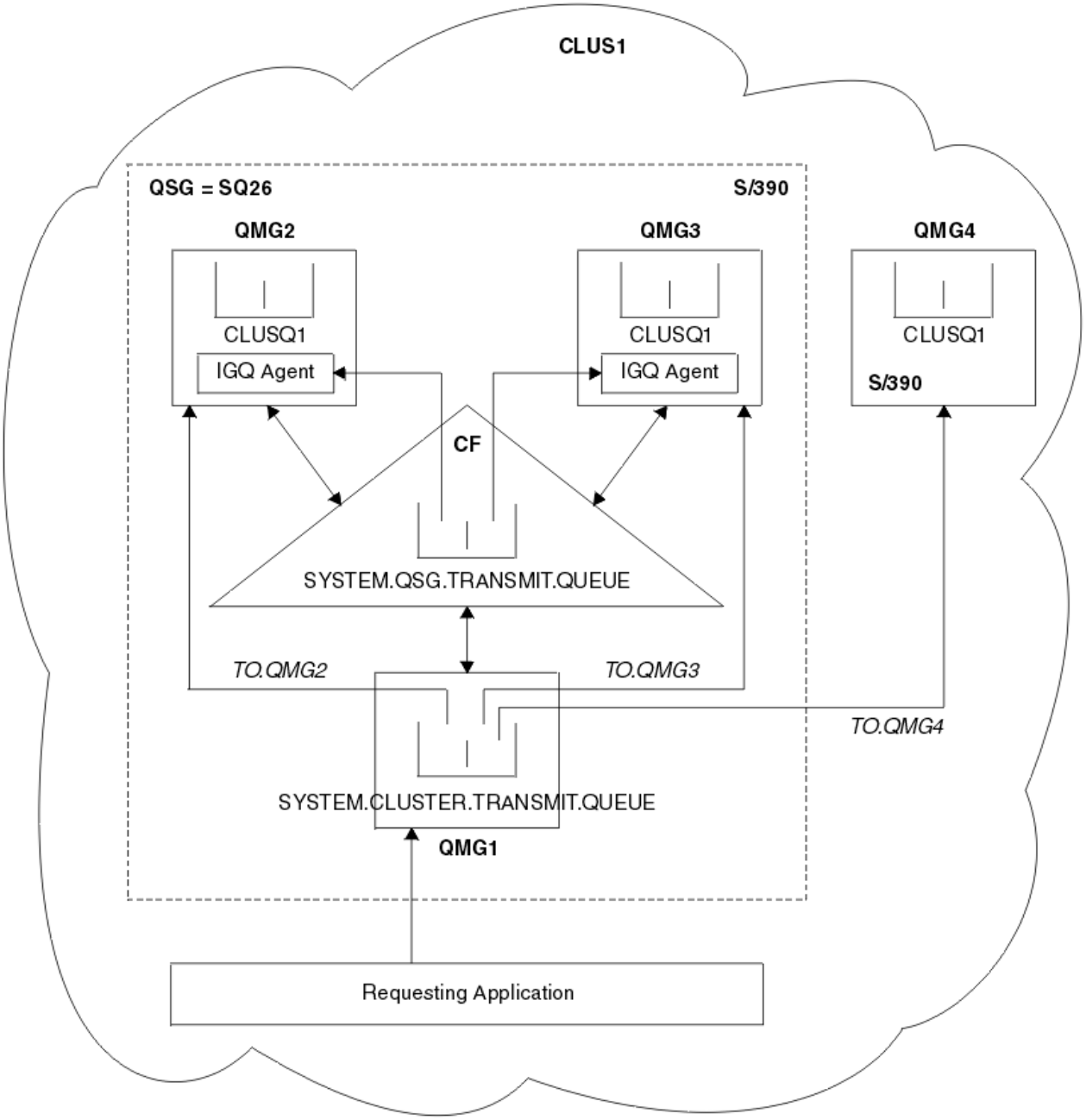
Notu gösteren noktalar

1. İstekte bulunan uygulamanın, iletilerin teslimi için kullanılan temel mekanizmanın farkında olması gerekmez.
2. Küçük iletiler için daha hızlı bir ileti sağlama mekanizması elde edilebilir.
3. İleti teslimi için birden çok yol vardır (yani, olağan kanal rotası ve grup içi kuyruğa alma rotanıdır).
4. Grup içi kuyruğa alma rotayı, potansiyel olarak daha hızlı olmak üzere, normal kanal rotasına tercih edilir. İleti özelliklerine bağlı olarak, ileti teslimi iki yol arasında bölünebilir. Bu nedenle, iletiler sıradan teslim edilebilir (ancak, iletiler yalnızca olağan kanal rotaları kullanılarak teslim edildiyse, bu teslim de olanaklıdır).
5. Bir rota seçildiyse ve iletiler iletim kuyruklarına yerleştirildiyse, ileti teslimi için yalnızca seçilen rota kullanılır. SYSTEM.QSG.TRANSMIT.QUEUE ile ilgili işlenmemiş iletiler, XQ1iletim kuyruğuna yönlendirilmez.

z/OS *Grup içi kuyruğa alma ile kümeleme (birden çok teslim yolu)*

Kuyruk yöneticilerini, bir kuyruk paylaşım grubunda olduğu kadar bir kümede olacak şekilde yapılandırmak mümkün.

İletiler bir küme kuyruğuna gönderildiğinde ve yerel ve uzak hedef kuyruk yöneticileri aynı kuyruk paylaşım grubunda yer alıyorsanız, küçük iletilerin teslimi için grup içi kuyruklama kullanılır (SYSTEM.QSG.TRANSMIT.QUEUE) ve grup içi kuyruğa alma işlemi iletinin boyutunu destekliyorsa, büyük iletilerin teslimi. Ayrıca, SYSTEM.CLUSTER.TRANSMIT.QUEUE değeri, iletilerin, küme içindeki herhangi bir kuyruk yöneticisine teslim edilmesi için kullanılır, ancak kuyruk paylaşım grubunun dışında. Aşağıdaki şemada bu yapılandırma gösterilir (kanal başlatıcıları gösterilmez).



Şekil 71. Grup içi kuyruğa alma işlemi içeren kümeleme örneği

Çizge şunları gösterir:

- Four z/OS queue managers QMG1, QMG2, QMG3, and QMG4 configured in a cluster CLUS1.
- Queue managers QMG1, QMG2, and QMG3 configured in a queue sharing group SQ26.
- Kuyruk yöneticileri QMG2 ve QMG3üzerinde çalışan IGQ aracıları.
- QMG1içinde tanımlanan yerel SYSTEM.CLUSTER.TRANSMIT.QUEUE .

  Not: Netlik sağlamak için, SYSTEM.CLUSTER.TRANSMIT.QUEUE gösterilmeyen diğer kuyruk yöneticilerininde KUYRUK.
- The shared SYSTEM.QSG.TRANSMIT.QUEUE defined in the CF, which is in an IBM MQ structure configured with the CFLEVEL(3) RECOVER(YES) attribute.
- Küme kanalları TO.QMG2 (QMG1 ile QMG2arasında bağlantı kuruluyor), TO.QMG3 (QMG1 ile QMG3arasında bağlantı kurulması) ve TO.QMG4 (QMG1 ile QMG4arasında bağlantı kuruluyor).

- Cluster queue CLUSQ1 being hosted on queue managers QMG2, QMG3, and QMG4.

İstekte bulunan uygulamanın, küme kuyruğuna ilişkin hedef kuyruk yöneticisinin bir kerede seçilebilmesi için, istenen uygulamanın MQOO_BIND_NOT_FIXED seçeneğiyle açıldığını varsayın.

Seçilen hedef kuyruk yöneticisi QMG2:ise

- İstekte bulunan uygulama tarafından gönderilen tüm büyük iletiler şunlardır:
  - SYSTEM.QSG.TRANSMIT.QUEUE , CFLEVEL (3) yapısında olduğu için QMG1'ta SYSTEM.CLUSTER.TRANSMIT.QUEUE ' a yerleştirin; bu nedenle, yalnızca 63 KB ' ye kadar olan iletileri destekler.
  - TO.QMG2küme kanalını kullanarak QMG2 üzerindeki CLUSQ1 küme kuyruğuna aktarıldı
- İstekte bulunan uygulama tarafından gönderilen tüm küçük iletiler
  - Paylaşılan iletim kuyruğuna (SYSTEM.QSG.TRANSMIT.QUEUE. Bu kuyruk, RECOVER (YES) öznitelikle yapılandırılmış bir yapıda, bu nedenle hem kalıcı, hem de kalıcı olmayan küçük iletiler için kullanılır.
  - QMG2üzerinde IGQ aracısı tarafından alındı
  - QMG2üzerindeki CLUSQ1 küme kuyruğuna kont

Seçilen hedef kuyruk yöneticisi QMG4:ise

- QMG4 , kuyruk paylaşım grubunun SQ26üyesi olmadığından, istekte bulunan uygulamanın koyduğu tüm iletiler
  - QMG1'ta SYSTEM.CLUSTER.TRANSMIT.QUEUE ' a
  - TO.QMG4küme kanalını kullanarak QMG4 üzerindeki CLUSQ1 küme kuyruğuna aktarıldı

Notu gösteren noktalar

- İstekte bulunan uygulamanın, iletilerin teslimi için kullanılan temel mekanizmanın farkında olması gerekmez.
- Bir kuyruk paylaşım grubundaki kuyruk yöneticileri arasında (aynı kuyruk yöneticilerinin bir kümede olması durumunda bile) küçük, kalıcı olmayan iletilerin aktarılması için potansiyel olarak daha hızlı bir teslim mekanizması elde edilir.
- İleti teslimi için birden çok yol vardır (yani, hem kümeleme rotası hem de grup içi kuyruğa alma rotanıdır).
- Küme rotasına tercihte daha hızlı olmak üzere, grup içi kuyruğa alma rotası seçilidir. İleti özelliklerine bağlı olarak, ileti teslimi iki yol arasında bölünebilir. Bu nedenle, iletiler sıradan teslim edilebilir. Uygulama tarafından belirtilen MQOO_BIND_ * seçeneği dikkate alınmadan bu teslimata dikkat edilmesi önemlidir. Grup içi kuyruğa alma, iletileri kümeleme olarak aynı şekilde dağıtır; bu iletiler, MQOO_BIND_NOT_FIXED, MQOO_BIND_ON_XX_ENCODE_CASE_CAPS_LOCK_ON open, mqoo_bind_on_group ya da MQOO_BIND_AS_Q_DEF ' nin açık olarak belirtilip belirtilmediğine bağlı olarak, iletileri dağıtır.
- Bir rota seçildiyse ve iletiler iletim kuyruklarına yerleştirildiyse, ileti teslimi için yalnızca seçilen rota kullanılır. SYSTEM.QSG.TRANSMIT.QUEUE , SYSTEM.CLUSTER.TRANSMIT.QUEUE.

z/OS *Kümeleme, grup içi kuyruğa alma ve dağıtılmış kuyruklama*

Bir kümenin yanı sıra bir kuyruk paylaşım grubu olan ve gönderen/alıcı kanal çifti kullanılarak dağıtımlı bir kuyruk yöneticisine bağlı bir kuyruk yöneticisi yapılandırmak mümkündür.

Bu yapılandırma, grup içi kuyruğa alma ve grup içi kuyruğa alma ile kümeleme işlemi ile dağıtılmış kuyruklama birleşimidir.

Grup içi kuyruklama “Grup içi kuyruklama ile dağıtımlı kuyruğa alma (birden çok teslim yolu)” sayfa 210içinde açıklanmıştır.

Grup içi kuyruğa alma ile kümeleme, “Grup içi kuyruğa alma ile kümeleme (birden çok teslim yolu)” sayfa 212içinde açıklanmıştır.

z/OS Grup içi kuyruklama iletileri

This section describes the messages put to the SYSTEM.QSG.TRANSMIT.QUEUE.

İleti yapısı
İletim kuyruklarına yapılan diğer tüm iletiler gibi, SYSTEM.QSG.TRANSMIT.QUEUE öneki, iletim kuyruğu üstbilgisiyle (MQXQH) önlenir.

İleti kalıcılığı

IBM WebSphere MQ 5.3 ve üstü, paylaşılan kuyruklar hem kalıcı, hem de kalıcı olmayan iletileri destekler.

IGQ aracısı kalıcı olmayan iletileri işlerken kuyruk yöneticisi sona ererse ya da IGQ aracısı, işlem iletilerinin ortasında olağandışı bir şekilde sona ererse, işlenmekte olan kalıcı olmayan iletiler kaybedilebilir. Kurtarma işlemleri gerekli olduğunda, uygulamaların kalıcı olmayan iletilerin kurtarılması için gerekli düzenlemeleri yapması gerekir.

IGQ aracısı tarafından yayınlanan, kalıcı olmayan bir ileti için bir put isteği beklenmedik şekilde başarısız olursa, işlenmekte olan ileti kaybedilir.

İletilerin teslim edilmesi
IGQ aracısı, eşitleme noktası kapsamı dışındaki tüm kalıcı olmayan iletileri ve eşitleme noktası kapsamı içindeki tüm kalıcı iletileri alır ve sunar. Bu durumda, IGQ aracısı, Sync Point Coordinator (Sync Point Coordinator) olarak işlev görür. IGQ aracısı, bu nedenle, hızlı ve kalıcı olmayan iletiler gibi kalıcı olmayan iletileri bir ileti kanalına işlemektedir. Bkz. Hızlı, kalıcı olmayan iletiler.

İletilerin batması
IGQ aracısı, 50 ileti için sabit bir toplu iş boyutu kullanır. Bir toplu iş içinde alınan tüm kalıcı iletiler, 50 ileti aralıklarla kesinleştirilir. SYSTEM.QSG.TRANSMIT.QUEUE' ta alma için kullanılabilir başka ileti olmadığında, aracı kalıcı iletilerden oluşan bir toplu iş olarak kesinleştirir.

İleti büyüklüğü
SYSTEM.QSG.TRANSMIT.QUEUE , paylaşılan kuyruklar için desteklenen ileti uzunluğu üst sınırı eksi iletim kuyruğu üstbilgisinin (MQXQH) uzunluğuna sahip.

Varsayılan ileti kalıcılığı ve varsayılan ileti önceliği
SYSTEM.QSG.TRANSMIT.QUEUE , açık bir zamanda kurulan kuyruk adı çözme yolunda, daha sonra varsayılan kalıcılık ve varsayılan önceliğe sahip iletiler (ya da varsayılan kalıcılık ya da varsayılan öncelik ile) için, olağan kurallar, kullanılan varsayılan öncelik ve kalıcılık değerlerine sahip olan kuyruğun seçiminde uygulanır. (Kuyruk seçimi kurallarıyla ilgili daha fazla bilgi için IBM MQ iletileri bölümüne bakın).

İlgili kavramlar

“Teslim edilmemiş/işlenmemiş iletiler” sayfa 215
Bu konuda, SYSTEM.QSG.TRANSMIT.QUEUE.

“Rapor iletileri-Intra Group Kuyruklama” sayfa 216
Bu konuda, rapor iletileri açıklanır: varış onayı, teslim onayı, süre bitimi raporu ve özel durum raporu.

z/OS *Teslim edilmemiş/işlenmemiş iletiler*

Bu konuda, SYSTEM.QSG.TRANSMIT.QUEUE.

Bir IGQ aracısı, hedef kuyruğa bir ileti sunamazsa, IGQ aracısı:

- MQRO_DISCARD_MSG rapor seçeneğini sunar (teslim edilmemiş ileti için MQMD ' nin Rapor seçenekleri alanı belirtildiğini gösteriyorsa) ve teslim edilmemiş iletiyi atar.
- Teslim edilmeyen iletiyi hedef kuyruk yöneticisi için, ileti önceden atılmamışsa, hedef kuyruk yöneticisine ilişkin ileti kuyruğuna yerleştirmeyi dener. IGQ aracısı, iletiyi bir ölü harf kuyruğu üstbilgisiyle (MQDLH) önekler.

Ölü bir mektup kuyruğu tanımlanmadıysa ya da teslim edilmeyen bir ileti gönderilemez ve gönderilmeyen ileti gönderilmezse:

- kalıcı olarak, IGQ aracısı, işlemekte olduğu yürürlükteki kalıcı ileti kümesini yedekler ve yeniden deneme durumuna girer. Daha fazla bilgi için bkz “Grup içi kuyruğa alma özel özellikleri” sayfa 217.
- Kalıcı olmayan IGQ aracısı, iletiyi atar ve sonraki iletiyi işlemeye devam eder.

Kuyruk paylaşım grubundaki bir kuyruk yöneticisi, ilişkili IGQ aracısının tüm iletilerini işleme zamanı bulunmadan sonlandırılırsa, işlenmemiş iletiler SYSTEM.QSG.TRANSMIT.QUEUE . Daha sonra IGQ aracısı, iletileri hedef kuyruklara alır ve gönderir.

Bağlaşım olanağı SYSTEM.QSG.TRANSMIT.QUEUE işlemi işlendi, işlenmeyen ve kalıcı olmayan iletiler kaybedildi.

IBM , uygulamaların iletileri doğrudan iletim kuyruklarına koymamasını önerir. Bir uygulama, iletileri doğrudan SYSTEM.QSG.TRANSMIT.QUEUE, IGQ aracısı bu iletileri işleyemeyebilir ve SYSTEM.QSG.TRANSMIT.QUEUE. Daha sonra kullanıcılar, bu işlenmemiş bu iletilerle başa çıkmak için kendi yöntemlerini kullanmalarını sağlar.

z/OS *Rapor iletileri-Intra Group Kuyruklama*

Bu konuda, rapor iletileri açıklanır: varış onayı, teslim onayı, süre bitimi raporu ve özel durum raporu.

Varışta onay (COA) /teslim onayı (COD) rapor iletileri
COA ve COD iletileri, grup içi kuyruğa alma işlemi kullanıldığında kuyruk yöneticisi tarafından üretilir.

Süre bitim raporu iletileri
Süre bitim raporu iletileri kuyruk yöneticisi tarafından oluşturulur.

Kural dışı durum raporu iletileri

Teslim edilmemiş iletiye ilişkin ileti tanımlayıcısının *Rapor seçenekleri* alanında belirtilen MQRO_EXCEPTION_ * rapor seçeneğine bağlı olarak, IGQ aracısı gereken kural dışı durum raporunu oluşturur ve bu raporu belirtilen yanıt kuyruğuna yerleştirir. Grup içi kuyruklama, kural dışı durum raporunu hedef yanıtlama kuyruğuna teslim etmek için kullanılabilir.

Rapor iletisinin sürekliliği, teslim edilmemiş iletinin kalıcıcığıyla aynıdır. IGQ aracısı, hedef yanıtlama kuyruğu adını çözemezse ya da yanıt iletisini bir iletim kuyruğuna yerleştiremezse (hedef yanıt kuyruğuna sonraki aktarma için), kural dışı durum raporunu, rapor iletisinin oluşturulduğu kuyruk yöneticisinin ölüme ilişkin ileti kuyruğuna yerleştirmeyi dener. Bu olanaklı değilse, teslim edilmeyen ileti aşağıdaki durumlarda olur:

- kalıcı olarak IGQ aracısı, kural dışı durum raporunu atar, yürürlükteki ileti kümesini yedekler ve yeniden deneme durumuna girer. Daha fazla bilgi için, bkz. “Grup içi kuyruğa alma özel özellikleri” sayfa 217.
- Kalıcı olmayan, IGQ aracısı, kural dışı durum raporunu atar ve SYSTEM.QSG.TRANSMIT.QUEUE.

z/OS Grup içi kuyruğa alma güvenliği

Bu konu, grup içi kuyruğa alma için güvenlik düzenlemelerini açıklar.

IGQAUT (IGQ yetkisi) ve IGQUSER (IGQUSER kullanıcı kimliği) kuyruk yöneticisi öznitelikleri, IGQ aracısı hedef kuyrukları açtığında gerçekleştirilen güvenlik denetimi düzeyini denetleyebilecek şekilde ayarlanabilir.

Grup içi kuyruğa alma yetkisi (IGQAUT)

IGQAUT özniteliği, gerçekleştirilecek güvenlik denetimlerinin tipini ve dolayısıyla, hedef kuyruğa ileti koyma yetkisi belirlendiğinde, IGQ aracısı tarafından kullanılacak kullanıcı kimliklerini belirlemek için ayarlanabilir.

IGQAUT özniteliği, kanal tanımlamalarında bulunan PUTAUT özniteliğiyle benzerlik göstermektedir.

Grup içi kuyruğa alma kullanıcı kimliği (IGQUSER)

IGQUSER özniteliği, bir hedef kuyruğa ileti koyma yetkisi belirlerken, IGQ aracısı tarafından kullanılacak bir kullanıcı kimliğini (kullanıcı kimliği) göstermek için kullanılabilir.

IGQUSER özniteliği, kanal tanımlamalarında kullanılabilir olan MCAUSER özniteliğine benzer.

z/OS Grup içi kuyruğa alma özel özellikleri

Bu kısımda, nesne tanıtıcılarının geçerlik denetimi, kendi kendine kurtarma ve grup içi kuyruğa alma aracısının yeniden deneme yeteneği ve grup içi kuyruğa alma aracısı ve diziselleştirme de dahil olmak üzere, grup içi kuyruğa alma özel özellikleri açıklanmaktadır.

Nesne tutamaçlarının geçersiz kılınması (MQRC_OBJECT_CHANGED)

Nesne açıldıktan sonra bir nesnenin öznitelikleri değiştiyse, kuyruk yöneticisi, sonraki kullanımla nesne tanıtıcısını MQRC_OBJECT_CHANGED ile geçersiz kılar.

Grup içi kuyruklama, nesne tanıtıcısı geçersiz kılma işlemi için aşağıdaki kuralları tanıtır:

- SYSTEM.QSG.TRANSMIT.QUEUE , açık işleme sırasında grup içi kuyruklama ENABLED olduğu için, açık işleme sırasında ad çözme yolunda yer aldı, ancak grup içi kuyruğa alma sırasında DISABLED değeri saptandığında kuyruk yöneticisi nesne tanıtıcısını geçersiz kılar ve MQRC_OBJECT_CHANGED ile koyma isteği başarısız olur.
- SYSTEM.QSG.TRANSMIT.QUEUE , açık işleme sırasında grup içi kuyruğa alma geçersiz kılındığı için, açık işleme sırasında ad çözme yolunda yer almadı, ancak grup içi kuyruğa alma sırasında kuyruğa alma işlemi ENABLED olarak saptandıktan sonra kuyruk yöneticisi nesne tanıtıcısını geçersiz kılar ve MQRC_OBJECT_CHANGED ile koyma isteği başarısız olur.
- SYSTEM.QSG.TRANSMIT.QUEUE , açık işleme sırasında grup içi kuyruklama etkinleştirildiği için, açık işleme sırasında ad çözme yoluna eklendi, ancak SYSTEM.QSG.TRANSMIT.QUEUE tanımlamasının zaman koyma değerine göre değişeceği saptandığında, kuyruk yöneticisi nesne tanıtıcısını geçersiz kılar ve MQRC_OBJECT_CHANGED ile koyma isteği başarısız olur.

Grup içi kuyruğa alma aracısının kendi kendine kurtarılması

If the IGQ agent terminates abnormally, message CSQM067E is issued and the IGQ agent starts again.

Grup içi kuyruğa alma aracısının yeniden deneme yeteneği

IGQ aracısı, SYSTEM.QSG.TRANSMIT.QUEUE (kuyruk) (örneğin, tanımlı olmayan ya da yanlış özniteliklerle tanımlanmış olduğu ya da alınan ya da başka bir nedenle engellendiği için), IGQ aracısı yeniden deneme durumuna girer.

IGQ aracısı kısa ve uzun yeniden deneme sayılarını ve aralıklarını sunar. Bu sayılara ve aralıklara ilişkin değerler değiştirilemeyen değerler aşağıdaki gibidir:

Çizelge 19. Kısa ve uzun yeniden deneme sayısı ve aralık değerleri

Sabit	**Değer**
Kısa yeniden deneme sayısı	10
Kısa Yeniden Deneme Aralığı	60 saniye = 1 dak
Uzun yeniden deneme sayısı	999,999,999
Uzun Yeniden Deneme Aralığı	1200 saniye = 20 dk

Grup içi kuyruğa alma aracısı ve diziselleştirme

Eşdüzey kurtarma işlemi devam ederken, IGQ aracısının paylaşılan kuyruklara erişimi diziselleştirmek için bir girişimde bulunmaması başarısız olabilir.

IGQ aracısı paylaşılan bir kuyrukta ya da kuyruklarda kesinleştirilmemiş iletiler ile çalışırken bir kuyruk paylaşım grubunda kuyruk yöneticisi hatası varsa, IGQ aracısı sona erer ve başarısız olan kuyruk yöneticisi için paylaşılan kuyruk eşdüzey kurtarma işlemi gerçekleşir. Paylaşılan kuyruk eşdüzey kurtarma işlemi zamanuyumsuz bir etkindir; bu, başarısız olan kuyruk yöneticisinin ve aynı zamanda o kuyruk yöneticisi için IGQ aracısının, paylaşılan kuyruk eşdüzey kurtarma işleminin tamamlanmadan önce

yeniden başlatılma olasılığını bırakır. Bu da, kesinleştirilen iletilerin, hala kurtarılmakta olan iletilerle birlikte ve sıraların dışında işlenmesini sağlar. İletilerin sıra dışı işlenmediğinden emin olmak için IGQ aracısı, paylaşılan kuyruklara erişimi MQCONNX API çağrısını vererek diziselleştirir.

Eşdüzey kurtarma işlemi devam ederken, IGQ aracısının paylaşılan kuyruklara erişimi diziselleştirmek için bir girişimde bulunmaması başarısız olabilir. Bir hata iletisi yayınlanır ve IGQ aracısı yeniden deneme durumuna girilir. Kuyruk yöneticisi eşdüzey kurtarma işlemi tamamlandığında (örneğin, sonraki yeniden deneme sırasında IGQ aracısı başlayabilir.)

z/OS z/OSüzerinde depolama yönetimi

IBM MQ for z/OS , kalıcı ve geçici veri yapıları gerektirir ve bu verileri saklamak için sayfa kümelerini ve bellek arabelleklerini kullanır. Bu konular, IBM MQ ' in bu sayfa kümelerini ve arabellekleri nasıl kullandığı hakkında daha fazla ayrıntı verir.

İlgili kavramlar

“IBM MQ for z/OSiçin sayfa kümeleri” sayfa 218
IBM MQ for z/OS ' un iletileri saklamak için sayfa kümelerini nasıl kullandığını anlamak için bu konuyu kullanın.

“IBM MQ for z/OSiçin depolama sınıfları” sayfa 219
Depolama sınıfı, kuyruk yöneticisinin kuyrukları sayfa kümelerine eşlemesine olanak tanıyan bir IBM MQ for z/OS kavramsıdır. Hangi veri kümelerinin hangi kuyruklar tarafından kullanıldığını denetlemek için depolama sınıflarını kullanabilirsiniz.

“IBM MQ for z/OSiçin arabellekler ve arabellek havuzları” sayfa 220
IBM MQ for z/OS , geçici olarak verileri önbelleğe almak için arabellekleri ve arabellek havuzlarını kullanır. Arabelleklerin nasıl düzenlendiğini daha iyi anlamak için bu konuyu kullanın ve kullanılır.

İlgili başvurular

“IBM MQ for z/OSdepolama yönetimine ilişkin daha fazla bilgi nereden bulabileceği” sayfa 222
Use this topic as a reference to find further information about storage management for IBM MQ for z/OS.

z/OS IBM MQ for z/OSiçin sayfa kümeleri

IBM MQ for z/OS ' un iletileri saklamak için sayfa kümelerini nasıl kullandığını anlamak için bu konuyu kullanın.

Sayfa kümesi , IBM MQtarafından kullanılmak üzere özel olarak biçimlendirilmiş bir VSAM doğrusal veri kümesidir. Sayfa kümeleri, çoğu ileti ve nesne tanımlamasını saklamak için kullanılır.

Bu kural dışı durumlar, Db2' ta paylaşılan bir havuzda saklanan genel tanımlardır ve paylaşılan kuyruklardaki iletiler. Bunlar kuyruk yöneticisi sayfa kümelerinde saklanmaz. Paylaşılan kuyruklar hakkında daha fazla bilgi için bkz. “Paylaşılan kuyruklar ve kuyruk paylaşım grupları” sayfa 160ve genel tanımlamalarla ilgili daha fazla bilgi için Özel ve genel tanımlarkonusuna bakın.

IBM MQ sayfa kümeleri 64 GB ' ye kadar boyutlarda olabilir. Her bir sayfa kümesi, 00-99 aralığında bir tamsayı olan bir sayfa kümesi tanıtıcısı (PSID) ile tanımlanır. Her kuyruk yöneticisinin kendi sayfa kümeleri olmalıdır.

IBM MQ , nesne tanımlamalarını ve kuyruk yöneticiyle ilgili diğer önemli bilgileri saklamak için sayfa kümesi sıfırını (PSID=00) kullanır. For normal operation of IBM MQ it is essential that page set zero does not become full, so do not use it to store messages.

Sisteminizin performansını artırmak için, uzun ömürlü iletilerden kısa ömürlü iletileri farklı sayfa kümelerine yerleştirerek de ayırmalısınız.

Sayfa kümelerini biçimlendirmelisiniz ve IBM MQ bunun için bir FORMAT yardımcı programı sağlar; bkz. Sayfa kümelerini biçimlendirme (FORMAT). Sayfa kümeleri de IBM MQ altsisteminde tanımlanmalıdır.

IBM MQ for z/OS , tam olarak dolduysa, bir sayfa kümesini dinamik olarak genişletebileceğiniz şekilde yapılandırılabilir. IBM MQ , yeterli disk depolama alanı varsa, 123 mantıksal kapsamın varolması durumunda sayfa kümesini genişletmeye devam eder. Bu şekilde doğrusal veri kümesi tanımlanırsa, kapsamlar birimlere yayılabilir; ancak IBM MQ , 64 GB ' nin ötesinde sayfa kümelerini genişletemez.

Farklı bir IBM MQ kuyruk yöneticisinden bir IBM MQ kuyruk yöneticisinden sayfa kümelerini kullanamaz ya da kuyruk yöneticisi adını değiştiremezsiniz. Verileri bir kuyruk yöneticisinden diğerine aktarmak istiyorsanız, ilk kuyruk yöneticisinden tüm nesneleri ve iletileri kaldırmalı ve bunları başka bir kuyruk yöneticisinden yeniden yüklemeniz gerekir.

V6'dan önceki bir yayın düzeyini çalıştıran bir kuyruk yöneticisinde 4 GB' den büyük sayfa kümeleri kullanılması olanaklı değildir. Geçiş dönemi boyunca, büyük olasılıkla önceki bir kod yayınına geri düşmeniz gerekebilir:

- Sayfa kümesi 0 'ı 4 GB ' den büyük olacak şekilde değiştirmeyin.
- Önceki yayın düzeyiyle bir kuyruk yöneticisi yeniden başlatılırken, 4 GB ' den büyük diğer sayfa kümeleri çevrimdışı bırakılacaktır.

4 GB 'nin ötesinde genişleyen var olan sayfa kümelerinin geçişi hakkında daha fazla bilgi için bkz. 4 GB' den büyük olacak şekilde bir sayfa kümesi tanımlama.

Bir denetimcinin, çalışmakta olan bir kuyruk yöneticisine dinamik olarak sayfa kümeleri eklemesi ya da çalışan bir kuyruk yöneticisinden sayfa kümeleri kaldırılması mümkündür (sayfa kümesi sıfır dışında). Kuyruk yöneticisi yeniden başlatıldıktan sonra DEFine PSID komutu, yalnızca komut DSN anahtar sözcüğünü içeriyorsa çalıştırılabilir.

z/OS IBM MQ for z/OSiçin depolama sınıfları

Depolama sınıfı, kuyruk yöneticisinin kuyrukları sayfa kümelerine eşlemesine olanak tanıyan bir IBM MQ for z/OS kavramsıdır. Hangi veri kümelerinin hangi kuyruklar tarafından kullanıldığını denetlemek için depolama sınıflarını kullanabilirsiniz.

Depolama sınıflarının tanıtılması

Depolama sınıfı , bir ya da daha çok kuyrukları bir sayfa kümesine eşler. Bu, kuyruğa ilişkin iletilerin o sayfa kümesinde depolanmış olduğu anlamına gelir.

Depolama sınıfları, denetim, veri kümesi alanı ve yük yönetimi ya da uygulama yalıtma amacıyla, paylaşılmayan ileti verilerinin depolandığı yeri denetlemenizi sağlar. You can also use storage classes to define the XCF group and member name of an IMS region if you are using the IMS bridge (described in “IBM MQ ve IMS” sayfa 275).

Paylaşılan kuyruklar, sayfadaki iletiler sayfa kümelerinde saklanmadığı için, bir sayfa kümesi eşlemesi elde etmek için depolama sınıflarını kullanmaz.

Depolama sınıfları nasıl çalışır

- Bir depolama sınıfı tanımlıyor, DEFINE STGCLASS komutunu kullanarak, bir sayfa kümesi tanıtıcısı belirtin (PSID).
- Bir kuyruk tanımladığınızda, STGCLASS öznitelideki depolama sınıfını belirtiyorsunuz.

In the following example, the local queue QE5 is mapped to page set 21 through storage class ARC2.

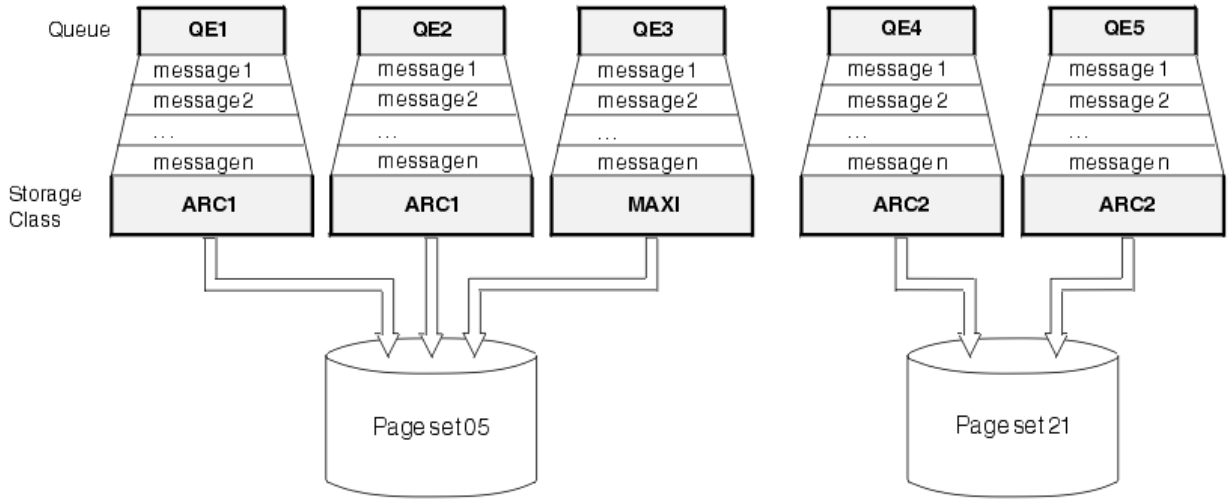
```
DEFINE STGCLASS(ARC2) PSID(21)
DEFINE QLOCAL(QE5) STGCLASS(ARC2)
```

Başka bir deyişle, QE5 kuyruğuna yerleştirilecek iletiler, (DASD ' ye yazılacak kadar kuyruğun üzerinde kalırlarsa) sayfa kümesi 21 'de saklanır.

Birden çok kuyruk aynı depolama sınıfını kullanabilir ve istediğiniz sayıda depolama sınıfı tanımlayabilirsiniz. Örneğin, aşağıda gösterildiği gibi, daha fazla depolama sınıfı ve kuyruk tanımlaması içerecek şekilde önceki örneği genişletebilirsiniz.

```
DEFINE STGCLASS(ARC1) PSID(05)
DEFINE STGCLASS(ARC2) PSID(21)
DEFINE STGCLASS(MAXI) PSID(05)
DEFINE QLOCAL(QE1) STGCLASS(ARC1) ...
DEFINE QLOCAL(QE2) STGCLASS(ARC1) ...
DEFINE QLOCAL(QE3) STGCLASS(MAXI) ...
DEFINE QLOCAL(QE4) STGCLASS(ARC2) ...
DEFINE QLOCAL(QE5) STGCLASS(ARC2) ...
```

Şekil 72 sayfa 220' ta hem ARC1 , hem de MAXI depolama sınıfları 05 sayfa kümesiyle ilişkilendirilir. Bu nedenle, QE1, QE2ve QE3 kuyrukları sayfa kümesi 05 'e eşlenir. Similarly, storage class ARC2 associates queues QE4 and QE5 with page set 21.



Şekil 72. Kuyruklar, depolama sınıfları aracılığıyla sayfa kümeleriyle eşleniyor

Bir depolama sınıfı belirtmeden bir kuyruk tanımlarsanız, IBM MQ varsayılan bir depolama sınıfı kullanır.

Var olmayan bir depolama sınıfının adı olan bir kuyruğa ileti konursa, uygulama bir hata alır. Var olan bir depolama sınıfı adı vermek için kuyruk tanımlamasını değiştirmeli ya da kuyruğun adını taşıyan bir depolama sınıfı yaratmanız gerekir.

Depolama sınıfını yalnızca aşağıdaki durumlarda değiştirebilirsiniz:

- Bu depolama sınıfını kullanan tüm kuyruklar boştur ve kesinleştirilmemiş bir etkinleşmez.
- Bu depolama sınıfını kullanan tüm kuyruklar kapatılır.

z/OS IBM MQ for z/OSiçin arabellekler ve arabellek havuzları

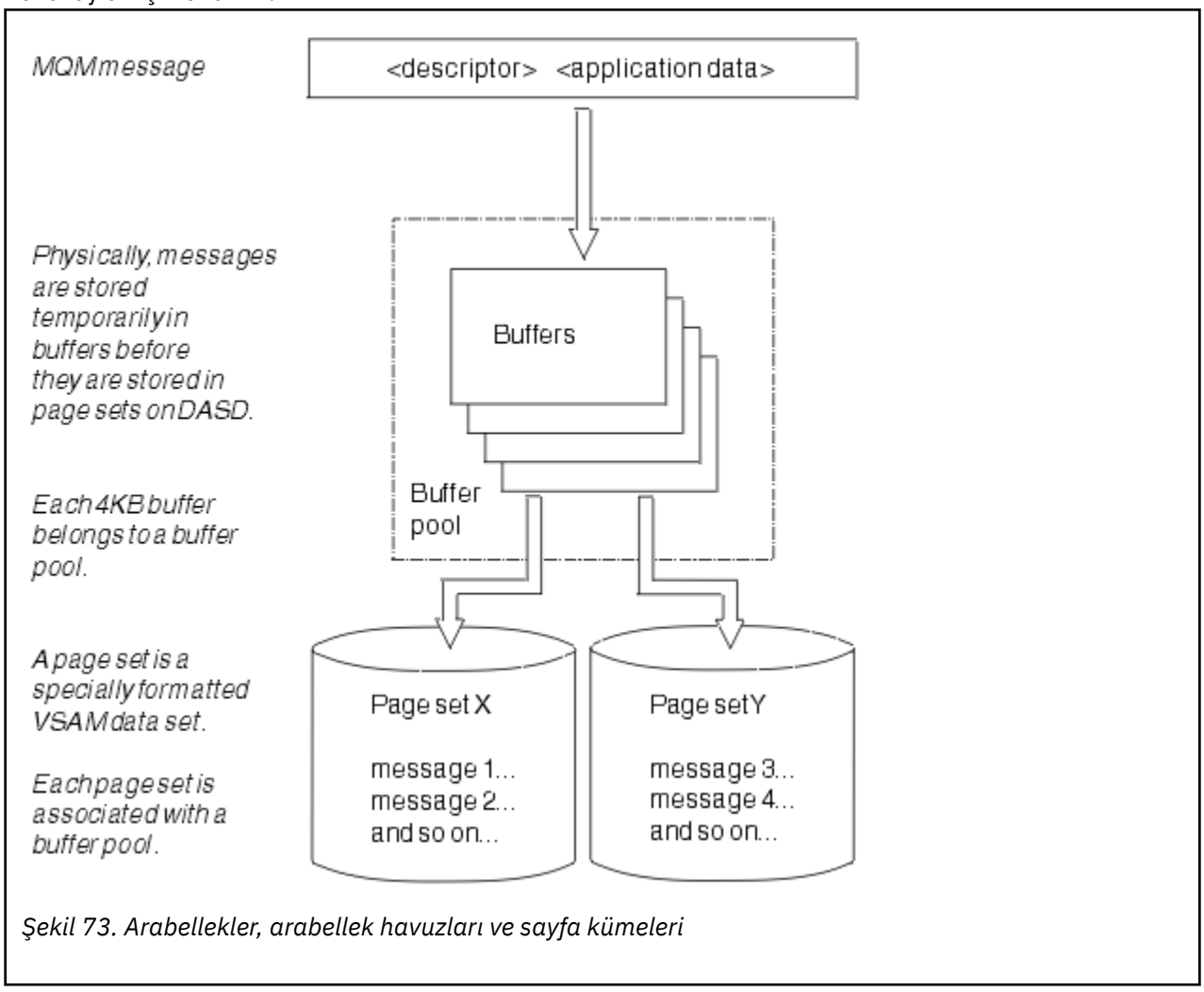
IBM MQ for z/OS , geçici olarak verileri önbelleğe almak için arabellekleri ve arabellek havuzlarını kullanır. Arabelleklerin nasıl düzenlendiğini daha iyi anlamak için bu konuyu kullanın ve kullanılır.

Verimlilik için, IBM MQ , iletiler (ve nesne tanımlamaları), DASD ' de sayfa kümelerinde saklanmadan önce geçici olarak arabellekler içinde saklanmakta olan bir önbelleğe alma biçimini kullanır. Kısa süreli iletiler, bunlar, alındıktan kısa bir süre sonra bir kuyruktan alınan iletiler, yalnızca arabelleklerde saklanabilir. Bu önbelleğe alma etkinliği, IBM MQbileşeninin bir parçası olan bir arabellek yöneticisi tarafından denetlenir.

Arabellekler *arabellek havuzları*içinde düzenlenir. Her kuyruk yöneticisi için en çok 100 arabellek havuzu (0-99) tanımlayabilirsiniz.

Şekil 73 sayfa 221' ta belirtilen nesne ve ileti tipi ayrışmasıyla tutarlı olarak en az sayıda arabellek havuzu kullanmanız ve uygulamanızın sahip olabileceği veri yalıtımı gereksinimlerini kullanmanız önerilir. Her arabellek 4 KB uzunluğlıdır. Arabellek havuzları varsayılan olarak 31 bit depolamayı kullanır; bu kipte, arabellek sayısı üst sınırı kuyruk yöneticisi adres alanında bulunan 31 bit saklama alanı miktarına göre belirlenir; arabellekler için %70 'ten fazlasını kullanmayın. Diğer bir seçenek olarak, arabellek havuzu depolama ayırma işlemi 64 bit depolamadan yapılabilir (**DEFINE BUFFPOOL** komutunun LOCATION özniteliğini kullanın). 64 bit depolamanın kullanılması için LOCATION (YUKARıN) kullanılması iki avantajdan yararlanır. Öncelikle, arabellek havuzlarının çok daha büyük ve ikinci olarak, 31 bit depolama alanı diğer işlevler tarafından kullanılmak üzere kullanılabilir hale getirilebileceği çok daha fazla 64 bit depolama alanı vardır. Genellikle, daha fazla arabelleğe sahip olduğunuz için, arabelleğe alma ve IBM MQ' ın performansı o kadar iyi olur.

Şekil 73 sayfa 221 , iletiler, arabellekler, arabellek havuzları ve sayfa kümeleri arasındaki ilişkiyi gösterir. Arabellek havuzu, bir ya da daha çok sayfa kümesiyle ilişkilendirilir; her sayfa kümesi tek bir arabellek havuzuyla ilişkilendirilir.



Şekil 73. Arabellekler, arabellek havuzları ve sayfa kümeleri

ALTER BUFFPOOL komutunu kullanarak, arabellek havuzu büyüklüğünü ve yerini değiştirmek için komutları dinamik olarak yayınlayabilirsiniz. Sayfa kümeleri, **DEFINE PSID** komutu kullanılarak dinamik olarak eklenebilir ya da **DELETE PSID** komutu kullanılarak silinebilir.

Bir arabellek havuzu çok küçükse, IBM MQ iletisi CSQP020Eiletisi verir. Bundan sonra, etkilenen arabellek havuzuna devingen olarak daha fazla arabellek ekleyebilirsiniz (bu işlemi yapmak için diğer arabellek havuzlarından arabellekleri kaldırmanız gerekebilir).

You specify the number of buffers in a pool with the **DEFINE BUFFPOOL** command, and you can dynamically resize buffer pools with the **ALTER BUFFPOOL** command. **DISPLAY USAGE** komutunu

kullanarak, arabellek havuzunu kullanan bir sayfa kümesini görüntüleyerek, bir havuzdaki geçerli arabelleklerin sayısını dinamik olarak saptarsınız.

Başarım nedenleri için, ileti ve nesne tanımlamalarını aynı arabellek havuzuna koymayın. Nesne tanımlamalarının alıkonduğu sayfa kümesi sıfır için yalnızca bir arabellek havuzu kullanın (sıfır numarası). Benzer şekilde, kısa ömürlü iletileri ve uzun ömürlü iletileri farklı arabellek havuzlarında ve bu nedenle farklı sayfa kümelerinde ve farklı kuyruklarda tutun.

Yeni bir arabellek havuzu yaratmak için yeniden başlatıldıktan sonra **DEFINE BUFFPOOL** komutu kullanılamaz. Bunun yerine, bir **DEFINE PSID** komutu DSN anahtar sözcüğünü kullanıyorsa, o anda tanımlı olmayan bir arabellek havuzunu açık olarak tanımlayabilir. Daha sonra, yeni arabellek havuzu yaratılacak.

z/OS IBM MQ for z/OSdepolama yönetimine ilişkin daha fazla bilgi nereden bulabileceği

Use this topic as a reference to find further information about storage management for IBM MQ for z/OS.

Bu bölümdeki konularla ilgili daha fazla bilgiyi aşağıdaki kaynaklardan bulabilirsiniz:

Çizelge 20. Depolama yönetimi hakkında daha fazla bilgi nereden bulalacağı	
Konu	**Nereye bakılacağı**
Ne kadar depolama alanına gereksinim duyarsınız?	Depolama ve performans gereksinimlerinizi z/OS üzerinde planlama
Sayfa kümelerinizi yapmak için ne kadar büyük ve arabellek havuzları	Sayfa kümelerinizi ve arabellek havuzlarınızı planlayın
Sayfa kümelerinin yönetilmesi	Sayfa kümelerinin yönetilmesi
MQSC komutları	MQSC komutları

z/OS oturum açmaIBM MQ for z/OS

IBM MQ maintains *günlükler* of data changes and significant events as they occur. Bu günlükler, gerekiyorsa önceki bir duruma veri kurtarmak için kullanılabilir.

önyükleme veri kümesi (BSDS), günlükleri içeren veri kümeleriyle ilgili bilgileri depolar.

Günlük, istatistik, izleme ya da performans değerlendirmesine ilişkin bilgileri içermez. IBM MQ ' in topladığı istatistik ve izleme bilgileriyle ilgili daha fazla ayrıntı için bkz. Monitoring and statistics.

Günlüğe kaydetme hakkında daha fazla bilgi için aşağıdaki konulara bakın:

- “IBM MQ for z/OSiçindeki günlük dosyaları” sayfa 223
- “Günlüğün nasıl yapılandırıldığı” sayfa 227
- “IBM MQ for z/OS günlüklerinin nasıl yazıldığı” sayfa 227
- “Daha büyük günlük Göreli Bayt Adresi” sayfa 230
- “Önyükleme verileri kümesi” sayfa 231

İlgili görevler

Günlüğe kaydetme ortamınızın planlanması

Sistem parametre modülü kullanılarak günlüklerin ayarlanması

z/OS Yönetmez/OS

İlgili başvurular

z/OS IBM MQ for z/OSiçin iletiler

z/OSüzerinde MQSC komutları yayınlayabileceğiniz kaynaklar

z/OS IBM MQ for z/OSiçindeki günlük dosyaları

Günlük dosyaları, hareket kurtarma için gereken bilgileri içerir. Etkin günlük dosyaları, günlük verilerini uzun bir süre tutabilmeniz için arşivleyebilirler.

Günlük dosyası nedir

IBM MQ , önemli olayları bir *etkin günlük*' de ortaya çıkar. Günlük, kurtarılması için gereken bilgileri içerir:

- Kalıcı iletiler
- IBM MQ nesneleri (kuyruklar gibi)
- IBM MQ kuyruk yöneticisi

Etkin günlük, döngüsel olarak kullanılan veri kümelerinin (310 'a kadar) bir derlemini oluşturur.

Etkin bir günlük bir kopyayı doldururken bir arşiv veri kümesinde yer alan günlük arşivlemeyi etkinleştirebilir. Arşivlemenin kullanılması, günlük verilerini uzatılmış bir süre için saklamanızı sağlar. Arşivleme kullanmayacaksa, günlüklerin aşağı kaydırılıp daha önceki verilerin üzerine yazılır. Bir sayfa kümesini kurtarmak ya da bir CF yapısındaki verileri kurtarmak için, sayfa kümesinin ya da yapının yedeğinin alındığı zaman veri günlüğü verilerine gereksinim duyarsınız. Diskte ya da manyetik bantta bir arşiv günlüğü yaratılabilir.

arşivleme

Etkin günlüğün sabit bir boyutu olduğu için, IBM MQ her bir günlük verilerinin içeriğini düzenli olarak bir *arşiv günlüğüne*kopyalar; bu da normalde doğrudan erişimli bir depolama aygıtı (DASD) ya da manyetik bant üzerine ayarlanan bir veri kümesidir. Bir altsistem ya da işlem hatası varsa, IBM MQ etkin günlüğü kullanır ve gerekirse, kurtarma işlemi için arşiv günlüğünü kullanır.

Arşiv günlüğü en çok 1000 adet sıralı veri kümesi içerebilir. You can catalog each data set using the z/OS integrated catalog facility (ICF).

Arşivleme, IBM MQ kurtarma için önemli bir bileşendir. Kurtarma birimi uzun süredir çalışıyorsa, bu kurtarma birimi içindeki günlük kayıtları arşiv günlüğünde bulunabilir. Bu durumda, kurtarma işlemi arşiv günlüğünden veri gerektirir. Ancak, arşivleme devre dışı bırakılırsa, yeni günlük kayıtları içeren etkin günlük kaydı, daha önceki günlük kayıtlarının üzerine yazılıyor. Bu, IBM MQ ' in kurtarma birimini geri veremeyebileceği ve iletilerin kaybedilebileceği anlamına gelir. Kuyruk yöneticisi olağandışı bir şekilde sonlanır.

Bu nedenle, bir üretim ortamında **hiçbir zaman arşivlemeyi kapatmayın**. Bunu yapmazsanız, bir sistem ya da işlem hatasından sonra veri kaybetme riskini göze alınız. Yalnızca bir test ortamında çalıştırıyorsanız, arşivlemeyi kapatmayı düşünebilir misiniz? If you need to do this, use the CSQ6LOGP macro, which is described in CSQ6LOGPkomutunu kullanma.

Planlanmamış uzun çalışma birimleriyle ilgili sorunların önlenmesine yardımcı olmak için IBM MQ , bir ileti (CSQJ160I ya da CSQJ161I) yayınlar. Etkin günlük boşaltma işlemi sırasında uzun süre çalışan bir iş birimi saptanırsa.

İkili günlüğe kaydetme

İkili günlüğe kaydetme işleminde, her bir günlük kaydı, yeniden başlatma sırasında veri kaybı sorunları olasılığını en aza indirmek için iki farklı etkin günlük veri kümesine yazılır.

IBM MQ konfigürasyonunu, *tek günlüğe kaydetme* ya da *ikili günlük kaydı*ile çalışacak şekilde yapılandırabilirsiniz. Tek günlük kaydı ile, günlük kayıtları etkin bir günlük veri kümesine bir kez yazılır. Her etkin günlük veri kümesi, tek kapsamlı bir VSAM doğrusal veri kümesidir (LDS). Çift günlük kaydı ile her bir günlük kaydı iki farklı etkin günlük veri kümesine yazılır. İkili günlük kaydı, yeniden başlatma sırasında veri kaybı sorunlarının ortaya çıkma olasılığını en aza indirir.

Günlüğe kaydetme işlemi

Günlüğe kaydetme işlemi, bazı iş birimlerinin günlük kayıtlarına ilişkin günlük kayıtlarının günlüğe daha sonra yazılmasına neden olur. Bu, uzun süre çalışan ya da uzun süreli iş birimleri için kuyruk yöneticisi yeniden başlatmada okunması gereken günlük verileri miktarını ya da geri alma işlemini geri alma işlemini azaltır.

Bir iş birimi uzun olduğu düşünüldüğünde, her bir günlük kaydının bir temsili daha sonra günlüğe yazılır. Bu teknik, *yıldırıcı*olarak bilinir. İş biriminin tamamı işlendiğinde, iş birimi *kapatılmış* durumda olur. Kapatılan iş birimi ile ilgili herhangi bir geri alma ya da yeniden başlatma etkinliği, özgün iş günlüğü kaydı birimini kullanmak yerine, shunted günlük kayıtlarını kullanabilir.

Uzun süredir çalışan bir iş biriminin saptanması denetim noktası işleminin bir işlecidir. Denetim noktası zamanında, her etkin iş birimi, kapatılıp açılması gerekip gerekmediğini belirlemek üzere işaretlendi. İş birimi oluşturulduğundan bu yana iki ön denetim noktasından geçtiyse ya da son kapatıldığı için, iş birimi parçalanmış olmaya uygundur. Bu, tek bir iş biriminin birden çok kez parçalanabileceği anlamına gelir. Bu, *multi-shunted* iş birimi olarak bilinir.

Bir iş birimi her üç kontrol noktasında bir parçalanmış durumda. Ancak, denetim noktası, günlük anahtarına (ya da LOGLOAD ' un aşılmasına neden olan günlük kaydının yazılması) zamanuyumsuz olarak gerçekleştirilir.

Bir kerede yalnızca tek bir denetim noktası vardır; bu nedenle, denetim noktası tamamlanmadan önce birden çok günlük anahtarı olabilir.

Başka bir deyişle, yeterli sayıda etkin günlük yoksa ya da çok küçüklerse, tüm günlükler doldurulmadan önce büyük bir iş birimi ile dalga geçilebilir.

Yıldırıcı tamamlamazsa, CSQR027I iletisi sonuçları elde eder.

If log archiving is turned off, ABEND 5C6 with reason 00D1032A occurs if there is an attempt to back out the unit of work for which shunting failed. Bu sorunu önlemek için, OFFLOAD=YES kullanmalısınız.

Oturum açma işlemi her zaman etkindir ve günlük arşivlemenin etkinleştirilip etkinleştirilmediğini çalıştırır.

Not: Bir iş birimine ilişkin tüm günlük kayıtları kaldırılsa da, her kaydın içeriğinin tamamı parçalanmaz, yalnızca geriletme için gerekli olan parça kapatılır. Bu, yazılan günlük verisi miktarının bir alt sınır olarak alıkonması ve bir sayfa kümesi hatası oluşması durumunda bu kayıtsız kayıtların kullanılamaması anlamına gelir. Uzun süre çalışan bir iş birimi, üç kuyruk yöneticisi denetim noktası için çalışmakta olan bir iş birimidir.

Günlüğe kaydetme işlemi hakkında daha fazla bilgi için Günlerin yönetilmesibaşlıklı konuya bakın.

Günlük sıkıştırması

IBM MQ for z/OS ' u, günlük veri kümesinden yazılırken ve okundukları için, günlük kayıtlarını sıkıştırmak ve sıkıştırılmış olarak açmak için yapılandırabilirsiniz.

Günlük sıkıştırması, özel kuyruklardaki kalıcı iletiler için günlüğe yazılan veri miktarını azaltmak için kullanılabilir. Elde edilen sıkıştırma miktarı, iletilerde bulunan veri tipine bağlıdır. Örneğin, yapılandırılmış ya da kayıt odaklı veriler için iyi sonuç verebilen yinelenen bayt eşgörünümlerini sıkıştırarak, Run Length Encoding (RLE) çalışır.

Uyarı: Paylaşılan bir kuyruğa konmakta olan kalıcı iletiler, günlük sıkıştırma işlemi için söz konusu değildir.

Sistem Yönetimi Olanağı 115 (SMF) kayıtlarının Log Manager (Günlük yöneticisi) bölümündeki alanları, veri sıkıştırmasının ne kadar elde edileceğini izlemek için kullanabilirsiniz. SMF ile ilgili daha fazla bilgi için bkz. Sistem Yönetimi Olanağının Kullanılması ve Muhasebe ve istatistik iletileri.

Günlük sıkıştırması, sistemin işlemci kullanımını artırır. Yalnızca kuyruk yöneticinizin verimi, günlük veri kümelerine yazma GÇ bant genişliği tarafından kısıtlandıysa ya da günlük veri kümelerini tutmak için gerekli disk depolama alanı tarafından kısıtlandıysa, sıkıştırma özelliğini kullanmayı düşünmelisiniz. Paylaşılan kuyruklar kullanıyorsanız, kuyruk paylaşım grubuna ek kuyruk yöneticileri eklenerek ve iş yükünü daha fazla kuyruk yöneticisi arasında dağıtmak için GÇ bant genişliği kısıtlamalarının giderilmesi giderilebilir.

Günlük sıkıştırma seçeneği, kuyruk yöneticisini durdurma ve yeniden başlatma gerekmeden gerektiği şekilde etkinleştirilebilir ve devre dışı bırakılabilir. Kuyruk yöneticisi, yürürlükteki günlük sıkıştırma ayarından bağımsız olarak, sıkıştırılmış günlük kayıtlarını okuyabilir.

Kuyruk yöneticisi, günlük sıkıştırma işlemi için 3 ayarı destekler.

YOK
Günlük veri sıkıştırma kullanılmıyor. Bu varsayılan değerdir.

RLE
Günlük veri sıkıştırma işlemi, çalıştırma uzunluğu kodlaması (RLE) kullanılarak gerçekleştirilir.

HERHANGİ BİRİ
Kuyruk yöneticisini geçerli kılmak için en yüksek düzeyde günlük kaydı sıkıştırması veren sıkıştırma algoritmasını seçin. Bu seçenek RLE sıkıştırması ile sonuçlanır.

Aşağıdakilerden birini kullanarak günlük kayıtlarının sıkıştırmasını denetleyebilirsiniz:

- MQSC ' de SET ve DISPLAY LOG komutları; bkz. SET LOG ve DISPLAY LOG
- PCF arabirimindeki Günlük ve Sorgulama Günlüğü Işlevleri; bkz. Günlük tanımlanması ve Günlük sorgulama günlüğü
- Sistem parametre modülündeki CSQ6LOGP makrosu; bkz. CSQ6LOGPkomutunu kullanma

Ayrıca, Günlük Yazdırma yardımcı programı CSQ1LOGP , sıkıştırılmış günlük kayıtlarının genişletilmesi için destek içerir.

Günlük verileri

Günlük en fazla 18 milyon milyon milyon ($1.8* 10^{19}$) byte. Her bir bayt, günlüğün başlangıcından göreli konularak adreslenilebilir ve bu göreli konum *göreli bayt adresi* (RBA) olarak bilinir.

RBA, 6 ya da 8 baytlık günlük RBA ' ların kullanımda olup olmadığına bağlı olarak toplam adreslenebilir 2^{48} baytı ya da 2^{64} baytı veren 6 baytlık ya da 8 baytlık bir alan tarafından başvurulmaktadır.

However, when IBM MQ detects that the used range is beyond F00000000000 (if 6-byte RBAs are in use) or FFFF800000000000 (if 8-byte log RBAs are in use), messages CSQI045, CSQI046, CSQI047, and CSQJ032 are issued, warning you to reset the log RBA.

RBA değeri FFF800000000 'e ulaşırsa (6 baytlık günlük RBA' ları kullanımdaysa) ya da FFFFFFC000000000 (8 baytlık günlük RBA ' ları kullanımdaysa) kuyruk yöneticisi, 00D10257neden koduyla sonlanır.

Kullanılan günlük aralığına ilişkin uyarı iletileri yayınlandıktan sonra, kuyruk yöneticisinin 8 baytlık günlük RBA ' ları kullanacak şekilde dönüştürülebileceği bir kuyruk yöneticisi kesintisi planlamanız ya da günlük ilk duruma getirilebilmesi için bir kuyruk yöneticisi kesintisi planlamanız gerekir. Günlüğü ilk durumuna getirme yordamı Kuyruk yöneticisinin günlüğünün sıfırlanmasıkonusunda belgelenir.

Kuyruk yöneticiniz 6 baytlık günlük RBA 'ları kullanıyorsa, Implementing the daha büyük log relative Byte Address(Göreli Byte Adresinin uygulanması) başlıklı bölümdeki yordamı izleyerek, kuyruk yöneticisinin günlük RBA' yı sıfırlamak yerine 8 baytlık günlük RBU ' ları kullanacak şekilde dönüştürmeyi düşünebilirsiniz.

Günlük, her biri tek bir birim olarak işlem gören bir günlük verileri kümesi olan *günlük kayıtları*' dan oluşur. Günlük kaydı, üstbilgisinin ilk baytının RBA 'si ya da günlük kaydı sıra numarası (LRSN) tarafından tanımlanır. RBA ya da LRSN, günlükteki belirli bir noktada başlayan bir kaydı benzersiz olarak tanımlar.

Günlük noktalarını tanımlamak için RBA ya da LRSN ' yi kullanıp kullanmayacağınızı, kuyruk paylaşım gruplarını kullanmanıza bağlıdır. Kuyruk paylaşımı ortamında, bir günlük noktasını benzersiz olarak tanımlamak için göreli byte adresini kullanamazsınız; çünkü, birden çok kuyruk yöneticisi aynı kuyruğu aynı anda güncelleyebilirler ve her birinin kendi günlüğü vardır. Bunu çözmek için, günlük kaydı sıra numarası bir zaman damgası değerinden türetilir ve günlük kaydının günlük kaydının fiziksel olarak yer değiştirmesini göstermez.

Her bir günlük kaydının, tipini veren bir üstbilgisi, kaydı yapan IBM MQ alt bileşeni ve kurtarma kayıtları birimi için bir kurtarma tanıtıcısı birimi bulunur.

Aşağıdaki başlıklar altında açıklanan dört günlük kaydı tipi vardır:

- Kurtarma günlüğü kayıtları birimi
- Denetim noktası kayıtları
- Sayfa kümesi denetim kayıtları
- CF yapısı yedekleme kayıtları

Kurtarma günlüğü kayıtları birimi

Günlük kayıtlarının çoğu, IBM MQ kuyruklarındaki değişiklikleri açıklar. Bu tür tüm değişiklikler, kurtarma birimleri içinde yapılır.

IBM MQ , yeniden başlatma sürelerini azaltmak ve sistem kullanılabilirliğini artırmak için *geri alma/yineleme* ve *günlük kayıtlarını dengelemek* ile ilgili özel günlüğe kaydetme tekniklerini kullanır.

Bunun bir etkisi, yeniden başlatma sürelerinin sınırlı olması. Yeniden başlatma sırasında bir hata oluşursa, kuyruk yöneticisinin ikinci kez yeniden başlatılması gerekir; ilk yeniden başlatma sırasında hata noktasına kadar tamamlanan tüm kurtarma etkinliğinin ikinci yeniden başlatma sırasında yeniden uygulanması gerekmez. Bu, art arda yeniden başlatma işlemlerinin aşamalı olarak daha uzun sürme süreceği anlamına gelir.

Denetim noktası kayıtları

Yeniden başlatma süresini kısalmak için, IBM MQ olağan işletim sırasında periyodik denetim noktaları alır. Bunlar aşağıdaki gibi oluşur:

- Önceden tanımlanmış bir sayıda günlük kaydı yazıldığında. This number is defined by the checkpoint frequency operand called LOGLOAD of the system parameter macro CSQ6SYSP, described in CSQ6SYSPkomutunu kullanma.
- Başarılı bir yeniden başlatma sonunda.
- Normal sonlandırma.
- IBM MQ , döngüde bir sonraki etkin günlük veri kümesine geçtiğinde.

At the time a checkpoint is taken, IBM MQ issues the DISPLAY CONN command (described in GöRüNEN EKRAN) internally so that a list of connections currently in doubt is written to the z/OS console log.

Sayfa kümesi denetim kayıtları

Bu kayıtlar, her bir denetim noktasında IBM MQ kuyruk yöneticisi tarafından bilinen sayfa kümelerini ve arabellek havuzlarını kaydeder ve denetim noktası sırasında sayfa kümesinin ortam kurtarmasını gerçekleştirmek için gerekli olan günlük aralıklarıyla ilgili bilgileri kaydeder.

Sayfa kümeleri ve arabellek havuzlarına ilişkin belirli dinamik değişiklikler, değişikliklerin kurtarılması ve bir sonraki kuyruk yöneticisinin yeniden başlatılması sırasında otomatik olarak yeniden yürürlüğe konması için sayfa kümesi denetim kayıtları olarak da yazılır.

CF yapısı yedekleme kayıtları

Bu kayıtlar, BACKUP CFSTRUCT komutuna yanıt olarak bir bağlaşım olanağı listesi yapısından okunan verileri tutar. Bir bağlaşım tesisi yapısı arızası olasılığının düşük olması durumunda, bu kayıtlar, kurtarma kayıtları birimi ile birlikte, bağlaşım tesisi yapısının ortam kurtarma işlemini hata noktasına kadar gerçekleştirmesi için RECOVER CFNO komutu tarafından kullanılır.

İlgili görevler

Daha büyük günlük Göreli Bayt Adresini uygulama

z/OS Günlüğün nasıl yapılandırıldığı

Günlük kayıtlarını tanımlamak için kullanılan terminolojiyi anlamak için bu konuyu kullanın.

Her etkin günlük veri kümesi bir VSAM doğrusal veri kümesi (LDS) olmalıdır. Etkin günlük veri kümesine yazılan fiziksel çıkış birimi 4 KB denetim aralığıyla (CI). Her bir yapılandırma öğesi bir VSAM kaydı içerir.

Fiziksel ve mantıksal günlük kayıtları

Bir VSAM yapılandırma öğesi, *fiziksel* bir kayıttır. Belirli bir zamanda günlüğe kaydedilen bilgiler, Yapılandırma Öğesindeki kullanılabilir alandan bağımsız olarak değişen bir uzunluğa sahip bir *mantıksal* kayıt oluşturur. Bu nedenle, bir fiziksel kayıt şunlar olabilir:

- Çok sayıda mantıksal kayıt
- Bir ya da daha çok mantıksal kayıt ve başka bir mantıksal kaydın bir parçası
- Yalnızca bir mantıksal kaydın parçası

Günlük kaydı terimi, depolamak için kaç tane *fiziksel* kaydın gerektiği dikkate alınmaksızın, *mantıksal* kayıt anlamına gelir.

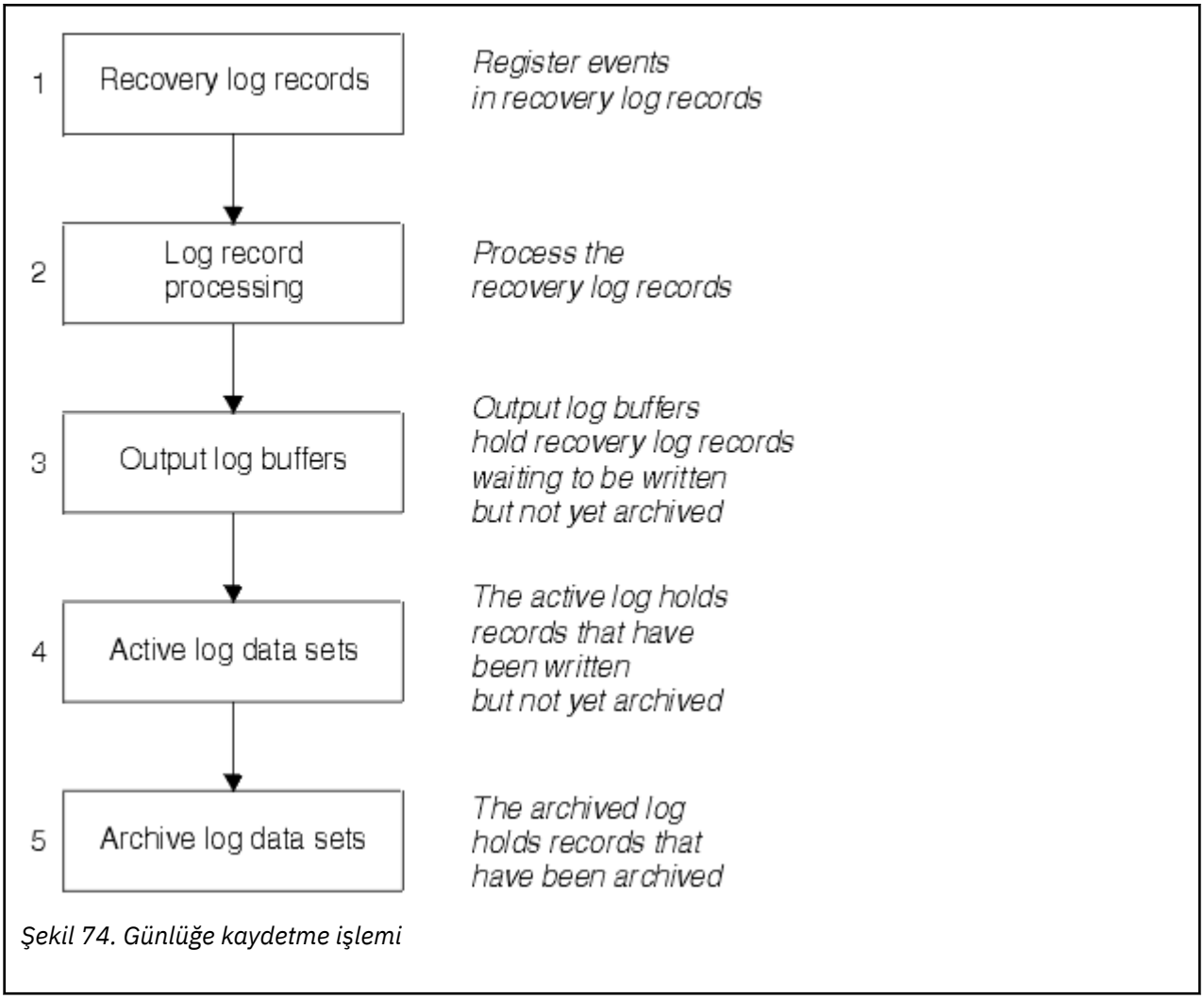
z/OS IBM MQ for z/OS günlüklerinin nasıl yazıldığı

IBM MQ ' in günlük dosyası kayıtlarını nasıl işlediğini anlamak için bu konuyu kullanın.

IBM MQ , her bir günlük kaydını *etkin günlük*olarak adlandırılan bir DASD veri kümesine yazar. Etkin günlük dolunca, IBM MQ , içeriğini *arşiv günlüğü*olarak adlandırılan bir DASD ya da manyetik bant veri kümesine kopyalar. Bu işleme *boşaltma*adı verilir.

Şekil 74 sayfa 228 , günlüğe kaydetme işlemini gösterir. Günlük kayıtları genellikle aşağıdaki döngüden geçer:

1. IBM MQ notları, veri değişiklikleri ve kurtarma günlüğü kayıtlarında önemli olayları değiştirir.
2. IBM MQ , kurtarma günlüğü kayıtlarını işler ve gerekirse bunları kesimler halinde ayırır.
3. Günlük kayıtları, VSAM Controls Intervals (Yapılandırma Öğesi) olarak biçimlendirilen *çıkış günlüğü arabelleklerine*sırayla yerleştirilir. Her bir günlük kaydı, 0-2 ^{64} --1 aralığındaki göreli bayt adresi ile tanımlanır.
4. Yapılandırma öğeleri, sırayla ve geri dönüştürülmüş olarak kullanılan, önceden tanımlanmış bir DASD etkin günlük veri kümeleri kümesine yazılır.
5. Her etkin günlük veri kümesi dolu olduğu için arşivleme etkinse, içeriği otomatik olarak yeni bir arşiv günlüğü veri kümesine yüklenir.



Şekil 74. Günlüğe kaydetme işlemi

Etkin günlük yazıldığında

Depolama içi günlük arabellekleri, aşağıdaki herhangi bir durumda etkin bir günlük veri kümesine yazılır:

- Günlük arabellekleri dolu olur.
- Yazma eşiğine ulaşılır (CSQ6LOGP makroda belirtildiği gibi).
- Kesinleştirme noktası gibi belirli önemli olaylar oluşur ya da bir IBM MQ BACKUP CFSTRUCT komutu verildiğinde.

Kuyruk yöneticisi kullanıma hazırlandığında, BSDS 'de adı geçen etkin günlük veri kümeleri, kuyruk yöneticisi tarafından dışlayıcı kullanım için dinamik olarak ayrılır ve kuyruk yöneticisi sona erinceye kadar yalnızca IBM MQ ' ye ayrılmış olarak kalır.

Günlük veri kümeleri dinamik olarak ekleniyor

Kuyruk yöneticisi çalışırken yeni etkin günlük veri kümelerinin dinamik olarak tanımlanması mümkündür. Bu özellik, geçici bir sorun nedeniyle, arşivleme etkin günlükleri boşaltamıyorsa, kuyruk yöneticisi askıda kalma sorununu hafifletir. Ek bilgi için DEFINE LOG komutuna bakın.

Not: Etkin günlükleri yeniden tanımlamak ya da kaldırmak için, kuyruk yöneticisini sonlandırmanız ve yeniden başlatmanız gerekir.

IBM MQ ve Depolama Yönetimi Altsistemi

IBM MQ parametreleri, IBM MQ arşiv günlüğü veri kümelerini dinamik olarak ayırdığında, Storage Management Subsystem (MVS/DFP SMS) depolama sınıflarını belirtmenizi sağlar. IBM MQ initiates the archiving of log data sets, but you can use SMS to perform allocation of the archive data set.

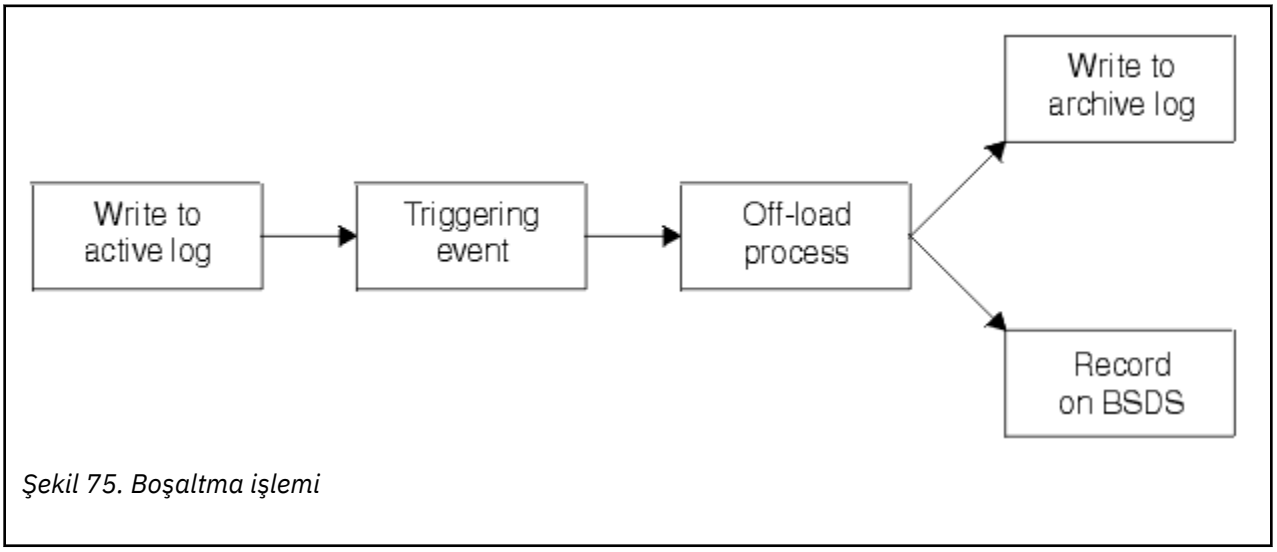
İlgili başvurular

“IBM MQ for z/OS arşiv günlüğü yazıldığında” sayfa 229
Etkin günlüklerin arşiv günlüklerine kopyalanmasını ve işlemin ne zaman gerçekleşeceğini anlamak için bu konuyu kullanın.

z/OS *IBM MQ for z/OS arşiv günlüğü yazıldığında*

Etkin günlüklerin arşiv günlüklerine kopyalanmasını ve işlemin ne zaman gerçekleşeceğini anlamak için bu konuyu kullanın.

Etkin günlüklerin arşiv günlüklerine kopyalanması işlemi *boşaltma*olarak adlandırılır. Diğer günlüğe kaydetme olaylarına ilişkin göreli olarak ilişki, Şekil 75 sayfa 229içinde şema olarak gösterilir.



Şekil 75. Boşaltma işlemi

Boşaltma işlemi tetikleniyor

Etkin bir günlüğün arşiv günlüğüne boşaltma işlemi birkaç olay tarafından tetiklenebilir. Örneğin:

- Etkin bir günlük veri kümesinin doldurulması.
- MQSC ARCHIVE LOG komutu kullanılıyor.
- Etkin bir günlük veri kümesine yazılırken oluşan bir hata oluştu.

Veri kümesi, hata noktasından önce kesilir ve yazılmamış olan kayıt, yeni veri kümesinin ilk kaydı olur. Normal bir tam günlük veri kümesi için olduğu gibi kesilen veri kümesi için offloading tetiklenir. İkili etkin günlükler varsa, iki kopya da eşzamanlanır şekilde, her iki kopya da kesilir.

Message CSQJ110E is issued when the last available active log is 5% full and at 5% increments thereafter, stating the percentage of the log's capacity that is in use. If all the active logs become full, IBM MQ stops processing, until offloading occurs, and issues this message:

```
CSQJ111A +CSQ1 OUT OF SPACE IN ACTIVE LOG DATA SETS
```

Boşaltma işlemi

Tüm etkin günlükler dolduğunda, IBM MQ boşaltma işlemi çalıştırılır ve boşaltma işlemi tamamlanıncaya kadar halleler. Etkin günlükler dolu olduğunda yük boşaltma işlemi başarısız olursa, IBM MQ olağandışı sona erer.

Etkin bir günlük boşaltılmaya hazır olduğunda, bir manyetik bandı monte etmek ya da bir DASD birimi hazırlamak için z/OS konsolu işletmenine bir istek gönderilir. ARCWTOR günlük kaydı seçeneğinin değeri (ek bilgi için bkz. CSQ6ARVPkomutunu kullanma) isteğin alınıp alınmayacağını belirler. Boşaltma işlemi için manyetik bant kullanıyorsanız, ARTTOR=YES değerini belirtin. Değer YES ise, isteğin başında bir WTOR (ileti numarası CSQJ008E) gelir. İşletmenin, tahsis edilecek bir arşiv günlüğü veri kümesi hazırlaması gerektiğini söylüyor.

İşlecin bu iletiyi hemen yanıtlamaması gerekir. Ancak, yanıtın ertelenmesi, yük boşaltma işleminin geciktirilmesine neden olur. operator, IBM MQ ' in etkin günlüklerden çalıştırılmasına neden olan yanıtı geciktirmediği sürece, bu, IBM MQ performansını etkilemez.

İşleç, boşaltma işlemini iptal ederek yanıt verebilir. Bu durumda, ayırma ikili arşiv veri kümelerinin ilk kopyasına ilişkin ise, boşaltma işlemi yalnızca bir sonraki etkin günlük veri kümesi dolu oluncaya kadar geciktirilir. Ayırma ikinci kopya için yer aldıysa, arşiv işlemi yalnızca bu veri kümesi için tek kopyalama kipine geçer.

Boşaltma sırasında kesintiler ve hatalar

Kuyruk yöneticisini durdurma isteği, boşaltma işlemi tamamlanıncaya kadar yürürlüğe girmez. Boşaltma işlemi devam ederken IBM MQ başarısız olursa, boşaltma işlemi kuyruk yöneticisi yeniden başlatıldığında yeniden başlar.

Boşaltma işlemi sırasında iletiler

Yüklenen iletiler z/OS konsoluna IBM MQ ve boşaltma işlemi tarafından gönderilir. Çeşitli günlük veri kümelerindeki RBA aralıklarını bulmak için bu iletileri kullanabilirsiniz.

z/OS Daha büyük günlük Göreli Bayt Adresi

Bu işlev, günlüğü sıfırlamak zorunda bulunmadan önce zaman aralığını artırarak kuyruk yöneticisinin kullanılabilirliğini artırır.

Kurtarma verileri, kuyruk yöneticisi yeniden başlatıldığında kalıcı iletilerin kullanılabilir olması için günlüğe yazılır. Günlük Göreli Bayt Adresi (log RBA), günlüğün başlangıcından göreli konum olarak veri yerini belirtmek için kullanılır.

IBM MQ 8.0öncesinde, 6 baytlık günlük RBA en çok 256 terabaytlık veri ele verebilirdir. Bu günlük kaydı miktarı yazılmadan önce, Kuyruk yöneticisinin günlüğünün sıfırlanmasıbaşlıklı konuda belgelenen yordamı izleyerek kuyruk yöneticisinin günlüğünü ilk durumuna getirmeniz gerekir.

Kuyruk yöneticilerinin günlüklerinin ilk durumuna getirilmesi hızlı bir işlem değildir ve işlemin bir parçası olarak sayfa kümelerini sıfırlaması gerekmesi nedeniyle genişletilmiş bir kesinti gerektirebilir. Yüksek bir kullanım kuyruk yöneticisi için bu işlem genellikle yılda bir kez yapılabilir.

IBM MQ 8.0'tan, günlük RBA' nın 8 bayt uzunluğunda olabilir ve günlük RBA ' nın ilk durumuna getirilmesi gerekmeden önce kuyruk yöneticisi artık 64 bin kez fazla veri (16 exabayt) olarak ele alabiliyor. Daha büyük günlük RBA ' yı kullanmanın etkisi, yazılan günlük verilerinin büyüklüğünün birkaç bayta kadar artmasını sağlar.

Bu işlev ne zaman etkinleştirilmiştir?

LTS Bu işlevi herhangi bir zamanda etkinleştirebilirsiniz, ancak ideal olarak değişiklik için plan yapmak ve mevcut günlük RBA ' nın 6 baytlık günlük RBA aralığının sonuna yaklaşılmadan önce, bu değişikliği planlamak ve işletmenize uyum sağlamak için açık bir şekilde etkinleştirmeniz gerekir. 8 baytlık

günlük RBA ' ya geçişin planlanması için yönergeler için Adreslenebilir günlük aralığı üst sınırını artırmak için planlama başlıklı konuya bakın.

V 9.2.5 IBM MQ 9.2.5 ya da daha sonraki bir zamanda yaratılan kuyruk yöneticileri bu işlevi etkinleştirmiş durumda.

Önemli: Bu yeteneği bir kuyruk paylaşım grubunda kullanabilmeniz için, kuyruk paylaşım grubundaki tüm kuyruk yöneticilerinin aşağıdaki düzeylerden birinde olması gerekir:

- At IBM MQ 9.0.n CD, IBM MQ 9.1.0 LTS, or later
- IBM MQ 9.0.0 ve **OPMODE**= (*NEWFUNC,800*) ya da **OPMODE**= (*NEWFUNC,900*) ile başlanmıştır.

Yürürlükteki günlük RBA, günlük RBA aralığının sonuna yaklaşıyorsa, kuyruk yöneticisini, kuyruk yöneticisinin günlüğünü sıfırlamak yerine 8 baytlık bir günlük RBA kullanacak şekilde dönüştürmeyi düşünün. Kuyruk yöneticisinin 8 baytlık günlük RBA ' yı kullanmak üzere dönüştürülmesi, günlüğün ilk durumuna getirilmesinden daha kısa bir kesinti gerektirmesi ve günlüğü ilk durumuna getirmeniz için zaman aralığını önemli ölçüde artırması gerekir.

Message CSQJ034I, issued during queue manager initialization, indicates the end of the log RBA range for the queue manager as configured, and can be used to determine whether 6 byte or 8 byte log RBAs are in use.

Bu işlev nasıl etkinleştirilir?

8 baytlık günlük RBA, kuyruk yöneticisi sürüm 2 biçimi BSDS ile başlatılarak etkinleştirilir. Özetle, aşağıdaki bilgiler elde edilir:

1. Kuyruk paylaşım grubundaki tüm kuyruk yöneticilerinin 8 baytlık günlük RBA ' yı etkinleştirmeye ilişkin gereksinimleri karşılamasını sağlama
2. Kuyruk yöneticisi temizleme işlemi sona erdiriliyor
3. BSDS ' nin sürüm 2 biçiminde bir kopyasını yaratmak için BSDS dönüştürme yardımcı programı dosyasını çalıştırma.
4. Dönüştürülen BSDS ile kuyruk yöneticisi yeniden başlatılıyor.

Kuyruk yöneticisi 8 baytlık günlük RBA 'yı kullanmak üzere dönüştürüldükten sonra, 6 baytlık günlük RBA' yı kullanmaya geri dönemez.

8 baytlık günlük RBU 'ların nasıl etkinleştirileceği konusunda ayrıntılı yordam için Daha büyük günlük Göreli Bayt Adresini uygulama ' e bakın.

İlgili görevler

Adreslenebilir günlük aralığı üst sınırını artırmayı planlama

İlgili başvurular

BSDS dönüştürme yardımcı programı (CSQJUCNV)

z/OS Önyükleme verileri kümesi

Önyükleme veri kümesi, günlük veri kümelerine ve günlük kayıtlarına gönderme yapmak için IBM MQ tarafından bir mekanizma olarak gereklidir. Bu bilgiler olağan işleme sırasında gereklidir ve kurtarma işlemini yeniden başlatın.

Önyükleme veri kümesinin neden olduğu

önyükleme veri kümesi (BSDS), IBM MQtarafından gerekli bilgileri içeren bir VSAM anahtar sıralı veri kümesidir (KSDS). Bu öğe aşağıdakileri içerir:

- IBM MQile bilinen tüm etkin ve arşivlenmiş günlük verileri kümelerinin dökümü. IBM MQ , bu dökümü aşağıdakileri yapmak için kullanır:
  - Etkin ve arşivlenmiş günlük verileri kümelerini izle
  - Normal işleme sırasında günlük okuma isteklerini yerine getirebilmesi için günlük kayıtlarını bulun

– Yeniden başlatma işlemini işleyebilmesi için günlük kayıtlarını bulun.

  Bir arşiv günlüğü veri kümesi tanımlansa ya da etkin bir günlük veri kümesi yeniden kullanıldığında,IBM MQ dökümdeki bilgileri saklar. Etkin günlükler için, tam ve hangilerinin yeniden kullanım için kullanılabilir olduğunu gösteren envanter gösterilir. Döküm, o veri kümesinde tutulan günlüğün her bir bölümünün göreli bayt adresini (RBA) tutar.

- En son IBM MQ etkinliğinin bir *aşağı kayması* dökümü. Kuyruk yöneticisini yeniden başlatmanız gerekirse bu gereklidir.

Kuyruk yöneticisinin bir hatası varsa ve yeniden başlatmanız gerekirse, BSDS gereklidir. IBM MQ **gerekir** 'un bir BSDS' leri vardır. Yeniden başlatma işlemi sırasında sorun olasılığını en aza indirmek için, IBM MQ ' u çift BSDSs ile yapılandırabilir ve her biri aynı bilgileri kaydetmektedir. Çift BSDSs kullanıldığında, *ikili kipte*çalışır durumda olarak bilinir. Olabilirse, kopyaları ayrı birimlere yerleştirin. Bu, birim bozuk ya da yok edildiyse, bunların her ikisinin de kaybolma riskini azaltır. DASD ' ye çift yazma yerine çift BSDSS kullanın.

BSDS, IBM MQ uyarlandığında ayarlanır ve değişiklik günlüğü döküm yardımcı programını (CSQJU003) kullanarak döküm alma işlemi yapabilirsiniz. Bu yardımcı programla ilgili daha fazla bilgi için bkz. Yönetme IBM MQ for z/OS. Bu, kuyruk yöneticisi başlatma yordamında bir DD deyimi tarafından başvuruluyor.

Olağan durumda, IBM MQ , BSDS ' nin kopyalarını çoğalir. Bir G/Ç hatası ortaya çıkarsa, hata oluştuğu kopyayı serbest bırakır ve tek bir BSDS ile devam eder. Çift kipli işlemi geri yükleyebilirsiniz. Bu işlem Yönetme IBM MQ for z/OSiçinde açıklanmaktadır.

The active logs are first registered in the BSDS when IBM MQ is installed. Etkin günlükleri sonlandırmadan ve kuyruk yöneticisini yeniden başlatamadan değiştiremezsiniz.

Arşiv günlüğü veri kümeleri dinamik olarak ayrılır. Bir değer ayrıldığında, veri kümesi adı BSDS ' de kaydedilir. Arşiv günlüğü veri kümelerinin listesi, arşivler eklendikçe genişler ve kullanıcı tarafından belirlenen sayıda girdi sayısına ulaşıldığında sona erir. Çift günlük kaydı için tek arşiv günlüğü ve 2000 için giriş sayısı üst sınırı 1000 'dir.

You can use a tape management system to delete the archive log data sets (IBM MQ does not have an automated method). Bu nedenle, arşiv günlüğü veri kümesiyle ilgili bilgiler, arşiv günlüğü veri kümesi sistem yöneticisi tarafından silindikten uzun süre sonra BSDS ' de olabilir.

Bunun tersine, arşiv günlüğü veri kümesi sayısı üst sınırı aşılmış olabilir ve veri kümesi süre bitimi tarihine ulaşmadan uzun süre önce BSDS ' deki veriler bırakılmıştır.

Günlüğün kapsamını belirlemek için aşağıdaki MQSC komutunu ve çeşitli ortam tipleri ya da kuyruk yöneticisi kurtarma işlemi için gerekli olan en erken günlük RBA ' yı bulunduran etkin ya da arşiv günlüğü veri kümesinin adını kullanabilirsiniz:

```
DISPLAY USAGE TYPE(DATASET)
```

Sistem parametre modülü, arşiv günlüğü veri kümelerinin ayrıldığında kataloğa alındığını belirtiyorsa, BSDS sonraki ayırmalar için gereken bilgiler için Integrated Catalog Facility (ICF) kataloğunu gösterir. Tersi durumda, her birim için BSDS girdileri, daha sonraki ayırmalar için gereken birim seri numarasını ve birim bilgilerini kaydeder.

BSDS sürümü

BSDS ' nin biçimi, kendi sürümüne göre değişir. BSDS sürümünün artırılması, yeni özelliklerin kullanılmasına olanak sağlar. Aşağıdaki BSDS sürümleri IBM MQtarafından desteklenmektedir:

Sürüm 1
IBM MQ' un tüm yayın düzeyleri tarafından desteklenir. Sürüm 1 BSDS, 6 baytlık günlük RBA değerlerini destekler.

Sürüm 2
IBM MQ 8.0 ve üstü tarafından desteklenir. Sürüm 2 BSDS, 8 baytlık günlük RBA değerlerini ve her etkin günlük kopyasında en çok 310 veri kümesini etkinleştirir.

V 9.2.5 IBM MQ 9.2.4 ve daha yüksek bir yerde yaratılan kuyruk yöneticileri için varsayılan olarak etkindir.

Sürüm 3
IBM MQ 8.0 ve üstü tarafından desteklenir. Etkin günlük kopyasına 31 'den fazla veri kümesi eklendiğinde BSDS, 2. sürümden sürüm 3 'e otomatik olarak dönüştürülür.

Yazdırma günlüğü eşlemi yardımcı programını (CSQJU004)çalıştırarak BSDS sürümünü belirleyebilirsiniz. BSDS ' yi Sürüm 1 'den Sürüm 2 'ye dönüştürmek için BSDS dönüştürme yardımcı programını çalıştırın (CSQJUCNV).

6 baytlık ve 8 baytlık günlük RBA ' larla ilgili daha fazla bilgi için “Daha büyük günlük Göreli Bayt Adresi” sayfa 230 başlıklı konuya bakın.

Günlük veri kümelerini ve BSDS kopyalarını arşivle

Her yeni arşiv günlüğü veri kümesi her yaratıldığında, BSDS ' nin bir kopyası da yaratılır. Arşiv günlüğü manyetik bantsa, BSDS ilk çıkış birimindeki ilk veri kümesidir. Arşiv günlüğü DASD ' de ise, BSDS ayrı bir veri kümesidir.

Arşiv günlüğünün veri kümesi adları ve BSDS kopyası aynıdır; ancak, arşiv günlüğü adının en alt düzey niteleyicisi A ile başlar ve BSDS kopyası B ile başlar, örneğin:

Arşiv günlüğü adı
CSQ.ARCHLOG1.E00186.T2336229. *A* 0000001

BSDS kopyası adı
CSQ.ARCHLOG1.E00186.T2336229. *B* 0000001

If there is a read error while copying the BSDS, the copy is not created, message CSQJ125E is issued, and the offloading to the new archive log data set continues without the BSDS copy.

z/OS Defining your system on z/OS

IBM MQ for z/OS , birçok varsayılan nesne tanımlaması kullanır ve bu varsayılan nesneleri yaratmak için örnek JCL sağlar. Bu varsayılan nesneleri ve örnek JCL ' yi anlamak için bu konuyu kullanın.

Sistem parametrelerinin ayarlanması

In IBM MQ for z/OS, a system parameter module controls the logging, archiving, tracing, and connection environments that IBM MQ uses in its operation. Sistem parametreleri üç çevirici makrosu tarafından belirtilir:

CSQ6SYSP
Bağlantı ve izleme ortamını ayarlama da içinde olmak üzere sistem parametreleri.

CSQ6LOGP
Günlüğe kaydetme parametreleri.

CSQ6ARVP
Arşiv parametrelerini günlüğe kaydet.

Varsayılan parametre modülleri IBM MQ for z/OSile birlikte sağlanır. Bu değerler kullanmak istediğiniz değerleri içermiyorsa, IBM MQile verilen örneği kullanarak kendi parametre modüllerinizi oluşturabilirsiniz. Örnek: `thlqual.SCSQPROC(CSQ4ZPRM)`.

Bir kuyruk yöneticisi çalışırken bazı sistem değiştirgelerinden değişiklik yapabilirsiniz. MQSC komutlarıiçinde SET SYSTEM, SET LOG ve SET ARCHEVE komutlarına bakın.

Tanımlama hakkında daha fazla bilgi için aşağıdaki konulara bakın:

- “IBM MQ for z/OSiçin sistem nesnelerinin tanımlanması” sayfa 234
- “IBM MQ for z/OSüzerinde kuyruk yöneticinizin ayarlanması” sayfa 238
- “Sample definitions supplied with IBM MQ for z/OS” sayfa 239

İlgili kavramlar
z/OSüzerinde MQSC komutları yayınlayabileceğiniz kaynaklar
İlgili görevler
Örnek kullanıma hazırlama giriş veri kümelerini uyarla
Yönetmez/OS
Kümeleri yapılandırma
İzlemeIBM MQ

z/OS IBM MQ for z/OSiçin sistem nesnelerinin tanımlanması

IBM MQ for z/OS , yayınlama/abone olma uygulamaları, küme ve kanal denetimi ve diğer sistem denetimi işlevleri için önceden tanımlanmış ek nesneler gerektirir.

IBM MQ for z/OS için gereken sistem nesneleri, aşağıdaki kategorilere ayrılabilir:

- Nesneleri yayınla/abone ol
- Varsayılan sistem nesneleri
- Sistem komut nesneleri
- Sistem denetimi nesneleri
- Kanallar kuyrukları
- Küme kuyrukları
- Kuyruk paylaşımı grup kuyrukları
- Depolama sınıfları
- Sistem nesnesi ölü harf kuyruğunun tanımlanması
- Varsayılan iletim kuyruğu
- İç kuyruklar
- “Kanal doğrulama kuyruğu” sayfa 238

Nesneleri yayınla/abone ol

There are several system objects that you need to define before you can use publish/subscribe applications with IBM MQ for z/OS. Bu nesneleri tanımlamanıza yardımcı olmak için IBM MQ ile birlikte örnek tanımlamalar sağlanır. Bu örnekler CSQ4INSGiçinde açıklanmıştır.

Yayınlama/abone olma olanağını kullanmak için aşağıdaki nesneleri tanımlamanız gerekir:

- SYSTEM.RETAINED.PUB.QUEUE(kuyruk yöneticisinde saklanan her bir yayının bir kopyasını tutmak için kullanılır). Her tam konu adı, bu kuyruğun üzerinde saklanan tek bir alıkonma yayınına sahip olabilir. Uygulamalarınız birçok farklı konuda alıkonan yayınlardan kullanılırsa ya da alıkonan yayın iletilerinizde büyük iletiler varsa, bu kuyruğa ilişkin depolama gereksinimleri, depolama gereksinimlerinin büyük olması durumunda kendi sayfa kümesine atanılması da dahil olmak üzere, dikkatle planlanmalıdır. Başarımı artırmak için, bu kuyruğu bir MSGID dizin tipiyle tanımlamanız gerekir (sağlanan örnek kuyruk tanımlamasında gösterildiği gibi).
- A local queue called SYSTEM.DURABLE.SUBSCRIBER.QUEUE, which is used to hold a persistent copy of the durable subscriptions in the queue manager. Başarımı artırmak için, bu kuyruğu bir CORRELID dizin tipiyle tanımlamanız gerekir (sağlanan örnek kuyruk tanımlamasında gösterildiği gibi).
- SYSTEM.DURABLE.MODEL.QUEUE; yönetilen kalıcı abonelikler için model olarak kullanılır.
- SYSTEM.NDURABLE.MODEL.QUEUE; kalıcı olmayan abonelikler için model olarak kullanılır.
- SYSTEM.QPUBSUB.QUEUE.NAMELIST), kuyruğa alınan yayınlama/abone olma arabirimiyle izlenen kuyruk adlarının listesini içerir.
- SYSTEM.QPUBSUB.SUBPOINT.NAMELIST, kuyruklanan yayınlama/abone olma arabirimi tarafından, konu nesnelerine abone olma noktalarıyla eşleşen bir konu nesnelerinin listesini içerir.

- SYSTEM.BASE.TOPIC, öznitelikleri çözümlemek için temel konu olarak kullanılır.
- SYSTEM.BROKER.DEFAULT.STREAM, kuyruğa yollanmış yayınlama/abone olma arabirimi tarafından kullanılan varsayılan akıdır.
- SYSTEM.BROKER.DEFAULT.SUBPOINT, kuyruğa yollanmış yayınlama/abone olma arabirimi tarafından kullanılan varsayılan RFH2 abonelik noktasıdır.
- SYSTEM.BROKER.ADMIN.STREAM, kuyruğa yollanmış yayınlama/abone olma arabirimi tarafından kullanılan denetim akışıdır.
- SYSTEM.DEFAULT.SUB, DEFINE ALT komutlarında varsayılan değerler sağlamak için kullanılan bir varsayılan abonelik nesnesi.

Sistem varsayılan nesneleri

Sistem varsayılan nesneleri, bir nesne tanımlarken varsayılan öznitelikleri sağlamak için kullanılır ve tanımlamayı temel olarak temel almak için başka bir nesnenin adını belirtmeyin.

Varsayılan sistem nesnesi tanımlamalarının adları "SYSTEM.DEFAULT"ya da"SYSTEM.DEF. " Örneğin, sistemin varsayılan yerel kuyruğu SYSTEM.DEFAULT.LOCAL.QUEUE.

Bu nesneler, bu IBM MQ nesnelerinin özniteliklerine ilişkin sistem varsayılanlarını tanımlar:

- Yerel kuyruklar
- Model kuyrukları
- Diğer Adlar
- Uzak kuyruklar
- Süreçler
- Ad listeleri
- Kanallar
- Depolama sınıfları
- Kimlik doğrulama bilgileri

Paylaşılan kuyruklar, yerel kuyruğun özel bir tipidir; bu nedenle, paylaşılan bir kuyruk tanımladığınızda, tanımlama SYSTEM.DEFAULT.LOCAL.QUEUE. Varsayılan tanımda belirtilmediğinden, Coupling Facility yapı adı için bir değer belirtmeyi unutmamanız gerekir. Diğer bir seçenek olarak, paylaşılan kuyruklar için temel olarak kullanmak üzere kendi varsayılan paylaşılan kuyruk tanımlamanızı tanımlayabilir ve böylece tüm bunların gerekli öznitelikleri edinmesini de kullanabilirsiniz. Yalnızca kuyruk paylaşım grubunda tek bir kuyruk yöneticisinde paylaşılan bir kuyruk tanımlamanız gerektiğini unutmayın.

Sistem komut nesneleri

Sistem komut nesnelerinin adları SYSTEM.COMMAND. Bir IBM MQ altsistemine komut vermek için IBM MQ işlemlerini ve denetim panolarını kullanabilmeniz için önce bu nesneleri tanımlamanız gerekir.

İki sistem komutu nesnesi vardır:

1. Sistem komutu giriş kuyruğu, komutların IBM MQ komut işlemcisi tarafından işlenmeden önce yerleştirdiği yerel bir kuyruktır. Adı SYSTEM.COMMAND.INPUT. SYSTEM.ADMIN.COMMAND.QUEUE , IBM MQ for Multiplatformsile uyumluluk ve MQ Console ve administrative REST APItarafından kullanılmak üzere tanımlanmalıdır.
2. SYSTEM.COMMAND.REPLY.MODEL , sistem komutu yanıtlama kuyruğunu tanımlayan bir model kuyruğıdır.

IBM MQ Explorertarafından kullanılmak üzere iki ek nesne vardır:

- SYSTEM.MQEXPLORER.REPLY.MODEL kuyruğu
- SYSTEM.ADMIN.SVRCONN kanalı

SYSTEM.REST.REPLY.QUEUE , IBM MQ administrative REST APItarafından kullanılan yanıt kuyruğıdır.

Komutlar genellikle kalıcı olmayan iletiler kullanılarak gönderilir. Böylece, sistem komut nesnelerinin DEFPSIST (NO) özniteliğine sahip olması gerekir. Böylece, bu iletiler (örneğin, yardımcı program ve işlemler ve denetim panoları gibi sağlanan uygulamalar da dahil olmak üzere), varsayılan olarak kalıcı olmayan iletiler elde eder. Komutlar için kalıcı iletiler kullanan bir uygulamanız varsa, bu tür komutlara yanıt iletileri kalıcı olduğundan, yanıtlama kuyruğu için DEFTYPE (PERMDYN) özniteliğini ayarlayın.

Sistem denetimi nesneleri

Sistem denetimi nesnelerinin adları, SYSTEM.ADMIN.

Yedi sistem denetimi nesnesi vardır:

- SYSTEM.ADMIN.CHANNEL.EVENT kuyruğu
- SYSTEM.ADMIN.COMMAND.EVENT kuyruğu
- SYSTEM.ADMIN.CONFIG.EVENT kuyruğu
- SYSTEM.ADMIN.PERFM.EVENT kuyruğu
- SYSTEM.ADMIN.QMGR.EVENT kuyruğu
- SYSTEM.ADMIN.TRACE.ROUTE.QUEUE kuyruğu
- SYSTEM.ADMIN.ACTIVITY.QUEUE kuyruğu

Kanal kuyrukları

Dağıtılmış kuyruklama kullanmak için aşağıdaki nesneleri tanımlamanız gerekir:

- SYSTEM.CHANNEL.SYNCQ, kanalların sıra numaralarını ve mantıksal iş tanıtıcılarının (LUWID) mantıksal birimlerini korumak için kullanılır. Kanal başarımını artırmak için, bu kuyruğu bir MSGID dizin tipiyle (sağlanan örnek kuyruk tanımlamasında gösterildiği gibi) tanımlamanız gerekir.
- SYSTEM.CHANNEL.INITQ(kanal komutları için kullanılır).

Bu kuyrukları paylaşılan kuyruklar olarak tanımlayamazsınız.

Küme kuyrukları

IBM MQ kümelerini kullanmak için aşağıdaki nesneleri tanımlamanız gerekir:

- SYSTEM.CLUSTER.COMMAND.QUEUE(kuyruk), kuyruk yöneticileri arasında havuz değişikliklerini iletmek için kullanılır. Bu kuyruğa yazılan iletiler, havuzdaki yerel kopyaya uygulanacak havuz verilerine ilişkin güncellemeler ya da havuz verilerine ilişkin istekler içerir.
- SYSTEM.CLUSTER.REPOSITORY.QUEUE(kuyruk), havuzun kalıcı bir kopyasını tutmak için kullanılır.
- SYSTEM.CLUSTER.TRANSMIT.QUEUE(Kuyruk), kümedeki tüm hedeflere ilişkin iletim kuyruğidir. Başarım nedenleriyle, bu kuyruğu bir CORRELID dizin tipiyle tanımlamanız gerekir (örnek kuyruk tanımlamasında gösterildiği gibi).

Bu kuyruklar genellikle çok sayıda ileti içerir.

Bu kuyrukları paylaşılan kuyruklar olarak tanımlayamazsınız.

Kuyruk paylaşım grubu kuyrukları

Paylaşılan kanalları ve grup içi kuyruğa alma olanağını kullanmak için aşağıdaki nesneleri tanımlamanız gerekir:

- SYSTEM.QSG.CHANNEL.SYNCQ, paylaşılan kanallara ilişkin eşitleme bilgilerini tutmak için kullanılır.

- SYSTEM.QSG.TRANSMIT.QUEUE, grup içi kuyruğa alma için iletim kuyruğu olarak kullanılır. Bir kuyruk paylaşım grubunda çalıştırıyorsanız, grup içi kuyruklama kullanmasanız da, bu kuyruğu tanımlamanız gerekir.

Depolama sınıfları

Aşağıdaki altı depolama sınıfını tanımlamanız önerilir. You must define four of them because they are required by IBM MQ. Diğer depolama sınıfı tanımlamaları, örnek kuyruk tanımlamalarında kullanıldığından önerilir.

DEFAULT (gerekli)
Bu depolama sınıfı, başarım açısından kritik önem veren ve diğer depolama sınıflarına uymayan tüm ileti kuyrukları için kullanılır. Ayrıca, kuyruk tanımlarken birini belirtmezseniz, bu değer de sağlanan varsayılan depolama sınıfıdır.

NODEFINE (gerekli)
Bu depolama sınıfı, bir kuyruk tanımladığınızda belirtilen depolama sınıfı tanımlı değilse kullanılır.

UZAK (gerekli)
Bu depolama sınıfı öncelikle iletim kuyrukları için, yani kısa ömürlü performans açısından kritik iletilere sahip sistem kuyrukları için kullanılır.

SYSLNGLV
Bu depolama sınıfı uzun ömürlü, performans açısından kritik öneme sahip iletiler için kullanılır.

SYSTEM (gerekli)
This storage class is used for performance critical, system related message queues, for example the SYSTEM.CHANNEL.SYNQ and the SYSTEM.CLUSTER.* kuyruklar.

SYSVOLAT
Bu depolama sınıfı, kısa ömürlü, performans açısından kritik öneme sahip iletiler için kullanılır.

Özniteliklerini değiştirebilir ve gereken diğer depolama sınıfı tanımlarını ekleyebilirsiniz.

Sistem nesnesi dead-letter kuyruğunun tanımlanması

İleti hedefi geçerli değilse, ölü-mektup kuyruğu kullanılır. IBM MQ , bu tür iletileri ölü-mektup kuyruğu adı verilen bir kuyruğa koyar. Ölü bir kuyruk olması zorunlu olmasa da, özellikle dağıtılmış kuyruklama ya da IBM MQ köprülerinden biri kullanıyorsanız, bu kuyruğun zorunlu olduğu kabul edilmemelidir.

Ölü-mektup kuyruğunu paylaşılan bir kuyruk olarak **tanımlamayın** . Bir kuyruk yöneticilikteki yerel bir kuyruğa koyma işlemi, çıkış kuyruğu kuyruğuna konabilir. İleti kuyruğu, paylaşılan bir kuyruksa, farklı bir sistemde bulunan bir ileti kuyruğu işleyicisi iletiyi işleyebilir ve aynı adı taşıyan bir kuyruğa yerleştirebilir; ancak bu, farklı bir kuyruk yöneticisininde olduğu için, kuyruğun yanlış olması ya da farklı bir güvenlik profilinin olması olabilir. Kuyruk yoksa, yeniden işleme işlemi başarısız olur.

Bir ölü-mektup kuyruğu tanımlamaya karar verirseniz, kuyruk yöneticisine adını da söylemeniz gerekir. Bu işlemi yapmak için ALTER QMGR DEADQ (*kuyruk-adı*) komutunu kullanın. Ek bilgi için Kuyruk yöneticisi özniteliklerinin görüntülenmesi ve değiştirilmesibaşlıklı konuya bakın.

Varsayılan iletim kuyruğu

Varsayılan iletim kuyruğu, başka bir kuyruk yöneticisine ileti göndermek için uygun başka bir uygun iletim kuyruğu olmadığında kullanılır. Varsayılan bir iletim kuyruğu tanımlarsaniz, kuyruğa hizmet edecek bir kanal da tanımlamanız gerekir. Bunu yapmazsanız, varsayılan iletim kuyruğuna yerleştirilecek iletiler uzak kuyruk yöneticisine iletilmez ve kuyruğun üzerinde kalır.

Varsayılan bir iletim kuyruğu tanımlamaya karar verirseniz, kuyruk yöneticisine adını da söylemeniz gerekir. Bunu yapmak için ALTER QMGR komutunu kullanın.

İç kuyruklar

- **Bekleyen veri kuyruğu**
  - İç kullanım için tanımlanmış bir kuyruk, SYSTEM.PENDING.DATA.QUEUE, JMS yayınlama/abone olma ortamında dayanıklı aboneliklerin kullanılmasını destekler.
- **JMS 2.0 teslim gecikmesi hazırlama kuyruğu**
  - JMS 2.0 tarafından sağlanan teslim gecikmesi işlevselliği kullanılırsa, bir iç konaklama kuyruğu SYSTEM.DDELAY.LOCAL.QUEUEolarak tanımlanmalıdır. Bu kuyruk, kuyruk yöneticisi tarafından, teslim gecikmesi tamamlanıncaya kadar sıfır olmayan bir teslim gecikmesiyle gönderilen iletileri geçici olarak saklamak için kullanılır ve ileti hedef hedefine konadır. CSQ4INSGiçinde, bu kuyruk için örnek bir tanımlama sağlanır, bu tanım geçersiz kılındı.
  - SYSTEM.DDELAY.LOCAL.QUEUE kuyruğunda, teslim gecikmesiyle gönderilecek beklenen ileti sayısı için STGCLASS, MAXMSGL ve MAXDEPTH özniteliklerini ayarlamalısınız. Ek olarak, SYSTEM.DDELAY.LOCAL.QUEUE kuyruğu, yalnızca kuyruk yöneticisinin bu kuyruğa ileti yerleştirebildiğinden emin olun. Hiçbir kullanıcı kimliğinin bu kuyruğa ileti koyma yetkisi olmadığından emin olmak için dikkatli olmanız gerekir.

Kanal doğrulama kuyruğu

Kanal kimlik doğrulamasının iç kullanımı için SYSTEM.CHLAUTH.DATA.QUEUE kuyruğu gerekli. Bu nesneleri tanımlamanıza yardımcı olmak için IBM MQ ile birlikte örnek tanımlamalar sağlanır. Bu örnek, bazı varsayılan kuralları da tanımlayan CSQ4INSA' da açıklanmaktadır.

z/OS IBM MQ for z/OSüzerinde kuyruk yöneticinizin ayarlanması

Temel performans sorunlarını önlemek için kuyruk yöneticinizin ayarlandığından emin olmak için alabileceğiniz birkaç basit adım vardır.

ALTER QMGR komutu tarafından ayarlanan kuyruk yöneticisi özniteliklerinin denetlediği kuyruk yöneticinizin başarımını artırabileceğiniz çeşitli yollar vardır. Bu kısımda, kuyruk yöneticisine izin verilen ileti sayısı üst sınırını ayarlayarak ya da kuyruk yöneticisininde 'houseleasy' (houselearn ') işlemini belirleyerek bunu nasıl yapabileceğiyle ilgili bilgiler yer alır. IBM MQ SupportPac MP16 - WebSphere MQ for z/OS Kapasite planlaması ve ayarlama , performans ve ayarlama hakkında daha fazla bilgi sağlar.

Eşitleme Noktaları

Kuyruk yöneticisinin rollerinden biri, bir uygulama içindeki syncpoint denetimidir. Uygulama, MQCMIT çağrısıyla sonlandırılan MQPUT ya da MQGET çağrılarını içeren bir iş birimini oluşturur.

Bir MQCMIT kapsamı içinde MQPUT ya da MQGET çağrılarının sayısı arttıkça, kesinleştirmenin performans maliyeti de önemli ölçüde artar. Genel olarak, uygulamalar MQPUT/MQGET değil, tek bir sözdiziminde çok sayıda iletiyi almak için tasarlanmalıdır.

Yönetimsel olarak, MAXUMSGS kuyruk yöneticisi özniteliğini kullanarak herhangi bir tek eşitleme noktası içindeki ileti sayısını daraltabilirsiniz. Bir uygulama bu sınırı aşarsa, sınırı aşan MQPUT,MQPUT1ya da MQGET çağrısına MQRC_SYNCPOINT_LIMIT_LIPRISTAD değerini alır. Uygulama bundan sonra, MQCMIT ya da MQBACK komutunu uygun şekilde yayınlamalıdır.

MAXUMSGS varsayılan değeri 10000 'dir. Bu değer, döngülü uygulamalara karşı korunmaya yardımcı olabilecek bir alt sınır uygulamak istiyorsanız, bu değeri indirebilir. MAXUMSGS ' yi azaltmadan önce, var olan uygulamalarınızın sınırı aşmamasını sağladığından emin olun ya da MQRC_SYNCPOINT_LIMIT_IADED dönüş koduna tahammül edebilir.

Süresi dolmuş iletiler

Süresi dolan iletiler, sonraki uygun MQGET çağrısıyla atılır. Ancak, böyle bir çağrı gerçekleşmezse, süresi dolan iletiler atılmaz ve bazı kuyruklar için, özellikle ileti alma işleminin MessageId, CorrelIdya da GroupId tarafından yapıldığı ve kuyruk performans için dizinlenmişse, süresi dolan birçok ileti birikebilir. Kuyruk yöneticisi, süresi dolmuş iletiler için düzenli aralıklarla herhangi bir kuyruğu tarayabilir ve daha sonra silinir. Bu taramanın ne sıklıkta gerçekleşeceğini, her zaman ne kadar sıklıkla gerçekleşeceğini seçebilirsiniz. Bunu yapmanın iki yolu vardır:

Belirtik istek
Hangi kuyrukların tarandığını ve ne zaman tarandığını denetleyebilirsiniz. Taranmasını istediğiniz kuyruğu ya da kuyrukları belirterek REFRESH QMGR TYPE (EXPIRY) komutunu verin.

Periyodik tarama
EXPRYINT özniteliğini kullanarak kuyruk yöneticisi nesnesinde bir süre bitimi aralığı belirtebilirsiniz. Kuyruk yöneticisi, her bir kuyrukta süresi dolan iletiler hakkında bilgi sağlar ve süresi dolan iletilerin taramanın ne zaman değdiğini bilir. EXPRYINT aralığına her ulaşıldığında, kuyruk yöneticisi süresi dolan iletiler için tarama yapmaya değer aday kuyruklarını arar ve yalnızca değerli olması için emdiği kuyrukları tarar. Tüm kuyrukları taramaz. Bu, herhangi bir işlemci süresinin gereksiz taramalar üzerinde boşa harcanmasını önler.

Paylaşılan kuyruklar yalnızca, kuyruk paylaşım grubunda bir kuyruk yöneticisi tarafından taranır. Genellikle, ilk kez yeniden başlatılacak ilk kuyruk yöneticisi ya da ilk olarak EXPYRINT kümesine sahip ilk kuyruk yöneticisi tarama gerçekleştirir.

Not: Bir kuyruk paylaşım grubu içindeki tüm kuyruk yöneticileri için aynı EXPRYINT değerini ayarlamalısınız.

z/OS Sample definitions supplied with IBM MQ for z/OS

Use this topic as a reference for the sample JCL, and code supplied with IBM MQ for z/OS.

Aşağıdaki örnek tanımlamalar, thlqual.SCSQPROC kitaplığındaki IBM MQ ile sağlanır. Bunları, sistem nesnelerini tanımlamak ve kendi nesnenizi uyarlamak için kullanabilirsiniz. Kullanıma hazırlama giriş veri kümelerine (Başlatma komutları içinde açıklanmıştır) bazılarını ekleyebilirsiniz.

Çizelge 21. Sistem nesneleri içinIBM MQ örnek tanımlamaları

Giriş veri kümesi kullanıma hazırlanıyor	**Örnek adı**
CSQINP1	CSQ4INP1 CSQ4INPR
CSQINP2	CSQ4INSA CSQ4INYS [1] CSQ4INSX CSQ4INSG CSQ4INSR CSQ4INSS CSQ4INSJ CSQ4INSM CSQ4INYG CSQ4INYR CSQ4INYC CSQ4INYD CSQ4INSC
CSQINPT	CSQ4INST CSQ4INYT

Çizelge 21. Sistem nesneleri içinIBM MQ örnek tanımlamaları (devamı var)	
Giriş veri kümesi kullanıma hazırlanıyor	**Örnek adı**
Diğer	CSQ4DISP CSQ4INPX CSQ4IVPQ CSQ4IVPG CSQ4MSTR CSQ4MSRR V 9.2.0 CSQ4QMIN

Not:

1. Bu örnek tanımların sırası önemlidir: INYS, INSX ve INSG yanlış sipariş edildiyse bir hata oluşur.

CSQINP1 örnekleri

Use the sample CSQINP1 data set thlqual.SCSQPROC(CSQ4INP1) when you are using one page set for each class of message, or thlqual.SCSQPROC(CSQ4INPR) when using multiple page sets for the major classes of message. Arabellek havuzlarının tanımlarını, sayfa kümesini arabellek havuzu ilişkilendirmelerini ve bir ALTER SECURITY komutunu içerir. Numuneyi, kuyruk yöneticinizin CSQINP1 bitiştirme işlemine dahil edin.

CSQINP2 örnekleri

CSQ4INSG sistem nesnesi örneği

Örnek CSQINP2 veri kümesi thlqual.SCSQPROC(CSQ4INSG), genel kullanım için aşağıdaki sistem nesnelerine ilişkin tanımlamalar içerir:

- Sistem varsayılan nesneleri
- Sistem komut nesneleri
- Sistem denetimi nesneleri
- Sistem kullanımına ilişkin diğer nesneler

Bu örnekteki nesneleri tanımlamanız gerekir, ancak bunu yalnızca altsistem ilk kez başlatıldığında bir kez yapmanız gerekir. CSQINP2 veri kümesinde tanımlamalar da dahil olmak üzere, bunu yapmak için en iyi yoldur. Bunlar kuyruk yöneticisi kapatma ve yeniden başlatma işlemi boyunca sürdürülenir. Nesne adlarını değiştirmemeniz gerekir, ancak gerekirse özniteliklerini değiştirebilirsiniz.

Aşağıdaki koşullar karşılandığında, SYSTEM.DURABLE.SUBSCRIBER.QUEUE kuyruğu (abone olma abone olma etkin değilse bile):

- QMGR özniteliği PSMODE özniteliği DISABLE olarak ayarlandı
- Örnek nesne CSQ4INST deyimi `DEFINE SUB('SYSTEM.DEFAULT.SUB')` var.

Bunu önlemek için `DEFINE SUB('SYSTEM.DEFAULT.SUB')` deyimini silin ya da yorum yapın.

JMS 2.0 teslim gecikmesi hazırlama kuyruğu, SYSTEM.DDELAY.LOCAL.QUEUE yalnızca JMS 2.0 teslim gecikmesi kullanılıyorsa tanımlanması gerekir. Varsayılan olarak, kuyruk tanımlaması açıklama satırı olarak tanımlıdır; bu açıklama gerekliyse açıklama satırı olarak kaldırılabilir.

CSQ4INSA sistem nesnesi ve kimlik doğrulama örneği

Örnek CSQINP2 veri kümesi thlqual.SCSQPROC(CSQ4INSA), kanal kimlik doğrulama sistem kuyruğu tanımını içerir. Bu kuyruk, kanal kimlik doğrulama kayıtlarını içerir. Ayrıca, varsayılan kanal doğrulama kurallarını da içerir.

Bu örnekte CHLAUTH, kuyruk yöneticisinde CHLAUTH etkinse ve kanal çalıştırmak istiyorsanız ya da CHLAUTH kaydının SET ya da DISPLAY CHLAUTH kaydı olmasını istiyorsanız, bu örnekteki nesneleri tanımlamanız gerekir. Bunları yalnızca altsistem ilk kez başlatıldığında tanımlamanız gerekir. CSQINP2 veri kümesinde tanımlamalar da dahil olmak üzere, bunu yapmak için en iyi yoldur. Bunlar, kuyruk yöneticisi kapatma ve yeniden başlatma işlemi boyunca sürdürür, kuyruk adını değiştirmemeniz gerekir.

CSQ4INSS sistem nesnesi örneği

Kuyruk paylaşım gruplarını kullanıyorsanız, ek sistem nesneleri tanımlayabilirsiniz.

Örnek veri kümesi thlqual.SCSQPROC(CSQ4INSS), CF yapılarıyla birlikte kullanılmak üzere örnek komutlar ve paylaşılan kanallar ve grup içi kuyruğa alma için gereken sistem nesnelerine ilişkin bir dizi tanımlama içerir.

Bu örneği olduğu gibi kullanamazsınız; kullanmadan önce özelleştirmeniz gerekir. Daha sonra, bu üyeyi kuyruk yöneticisi başlatma yordamında CSQINP2 DD birleştirmeye dahil edebilir ya da gerekli komutları vermek için CSQUTIL yardımcı programının COMMAND işlevine giriş olarak kullanabilirsiniz.

Grup ya da paylaşılan nesneler tanımlarken bunları, kuyruk paylaşım grubundaki yalnızca bir kuyruk yöneticisi için CSQINP2 DD birleştirmesi içine eklemeniz gerekir.

CSQ4INSX sistem nesnesi örneği

Dağıtılmış kuyruklama ve kümeleme kullanıyorsanız, ek sistem nesneleri tanımlamanız gerekir.

Örnek veri kümesi thlqual.SCSQPROC(CSQ4INSX), gereken kuyruk tanımlarını içerir. Bu üyeyi, kuyruk yöneticisi başlatma yordamında CSQINP2 DD birleştirmesine ekleyebilir ya da gereken DEFINE komutlarını vermek için CSQUTIL yardımcı programında COMMAND işlevinde giriş olarak kullanabilirsiniz.

İki tip nesne tanımlaması vardır:

- SYSTEM.CHANNEL.xx, dağıtılmış kuyruklama için gerekli
- SYSTEM.CLUSTER.xx, kümeleme için gereklidir

CSQ4INSJ sistem JMS nesnesi örneği

JMS yayınlama/abone olma etki alanında kullanılan kuyrukları tanımlar.

CSQ4INSM sistem nesnesi örneği

İleri düzey ileti güvenliği kullanıyorsanız, ek sistem nesneleri tanımlamanız gerekir. Örnek veri kümesi thlqual.SCSQPROC(CSQ4INSM), gereken kuyruk tanımlarını içerir.

CSQ4INSR nesne örneği

WebSphere Application Server ve aracıların kullandığı kuyrukları tanımlar.

CSQ4INYD nesne örneği

Dağıtılmış kuyruklama kullanıyorsanız ve kuyruklarınızı, işlemlerinizi ve kanallarınızı ayarlamanız gerekir.

Örnek veri kümesi thlqual.SCSQPROC(CSQ4INYD), dağıtılmış kuyruklama nesnenizi uyarlamak için kullanabileceğiniz örnek tanımlamaları içerir. Şu şekilde oluşur:

- Gönderme bitişi için bir tanımlama kümesi
- Alma sona erme için bir tanımlama kümesi
- İstemcileri kullanmak için bir tanımlama kümesi

Bu örneği şu şekilde kullanamazsınız; kullanmadan önce uyarlamanız gerekir. Daha sonra, bu üyeyi kuyruk yöneticisi başlatma yordamında CSQINP2 DD birleştirmeye dahil edebilir ya da gereken DEFINE komutlarını vermek için CSQUTIL yardımcı programının COMMAND işlevine giriş

olarak kullanabilirsiniz. (Bu, kuyruk yöneticisini her yeniden başlattığınızda bu nesneleri yeniden tanımlamanız gerekmediği için tercih edilir).

CSQ4INYC nesne örneği

Kümeleme kullanıyorsanız, kanal tanımlamalarına eşdeğeri olarak tanımlamalar ve dağıtılmış kuyruğa alma uzak kuyruk tanımlamalarına gerektiğinde otomatik olarak yaratılır. Ancak, bazı el ile kanal tanımları için, küme için bir küme alıcı kanalı ve en az bir küme havuzu kuyruk yöneticisine bir küme gönderici tanımlaması gerekir.

Örnek veri kümesi: thlqual.SCSQPROC(CSQ4INYC), kümeleme nesnenizi özelleştirmek için kullanabileceğiniz aşağıdaki örnek tanımlamalarını içerir:

- Kuyruk yöneticisine ilişkin tanımlar
- Alma kanalına ilişkin tanımlar
- Gönderme kanalına ilişkin tanımlar
- Küme kuyruklarına ilişkin tanımlar
- Küme listelerine ilişkin tanımlar

Bu örneği şu şekilde kullanamazsınız; kullanmadan önce uyarlamanız gerekir. Daha sonra, bu üyeyi kuyruk yöneticisi başlatma yordamında CSQINP2 DD birleştirmeye dahil edebilir ya da gereken DEFINE komutlarını vermek için CSQUTIL yardımcı programının COMMAND işlevine giriş olarak kullanabilirsiniz. Bu, IBM MQ' u her yeniden başlattığınız her defasında bu nesneleri yeniden tanımlamanız gerekmediği için tercih edilebilir.

CSQ4INYG nesne örneği

Örnek veri kümesi: thlqual.SCSQPROC(CSQ4INYG), genel kullanım için kendi nesnenizi özelleştirmek için kullanabileceğiniz aşağıdaki örnek tanımlamalarını içerir:

- Ölü-mektup kuyruğu
- Varsayılan iletim kuyruğu
- CICS bağdaştırıcı nesneleri

Bu örneği şu şekilde kullanamazsınız; kullanmadan önce uyarlamanız gerekir. Daha sonra, bu üyeyi kuyruk yöneticisi başlatma yordamında CSQINP2 DD birleştirmeye dahil edebilir ya da gereken DEFINE komutlarını vermek için CSQUTIL yardımcı programının COMMAND işlevine giriş olarak kullanabilirsiniz. Bu, IBM MQ' u her yeniden başlattığınız her defasında bu nesneleri yeniden tanımlamanız gerekmediği için tercih edilebilir.

Burada örnek tanımlara ek olarak, sistem nesnesi tanımlamalarını kendi kaynak tanımlarınız için temel olarak kullanabilirsiniz. For example, you can make a working copy of SYSTEM.DEFAULT.LOCAL.QUEUE and name it MY.DEFAULT.LOCAL.QUEUE. Bundan sonra, bu kopyadaki parametrelerin herhangi birini gerektiği şekilde değiştirebilirsiniz. Daha sonra, seçtiğiniz hangi yöntem tarafından bir DEFE komutu verebilirseniz, o tipteki kaynakları yaratma yetkiniz olmalıdır.

Varsayılan iletim kuyruğu

Varsayılan bir iletim kuyruğu tanımlayıp tanımlamayacağınıza karar vermeden önce Varsayılan iletim kuyruğu tanımlamasını okuyun.

- Varsayılan bir iletim kuyruğu tanımlamak istediğinize karar verirseniz, bunu sunmak için de bir kanal tanımlamanız gerektiğini unutmayın.
- Birini tanımlamak istemiyorsanız, örneğin, DEFXMITQ deyimini örnekteki ALTER QMGR komutundan kaldırmayı unutmayın.

CICS bağdaştırıcı nesneleri

Örnek, CICS01.INITQadlı bir başlatma kuyruğunu tanımlar. Bu kuyruk, IBM MQ tarafından sağlanan CKTI işlemi tarafından kullanılır. Bu kuyruğun adını değiştirebilirsiniz; ancak, INITPARM deyimindeki CICS sistemi kullanıma hazırlama çizelgesinde (SIT) ya da SYSIN geçersiz kılma değerinde belirtilen adla eşleşmelidir.

CSQ4INYS/CSQ4INYR nesne örnekleri

Kullanım için depolama sınıfı tanımlamaları:

- her ileti sınıfı için bir sayfa kümesi
- ana ileti sınıfları için birden çok sayfa kümesi

Örneğin, SYSTEM.COMMAND.INPUT , STGCLASS ('SYSVOLAT') ve SYSTEM.CLUSTER.TRANSMIT.QUEUE , STGCLASS ('REMOTE') olarak kullanır. CSQ4INYS' de, bu depolama sınıflarının her ikisi de aynı sayfa kümesini kullanır. CSQ4INYR' de, bu depolama sınıfları iletim kuyruğu dolduran etkisini azaltmak için farklı sayfa kümeleri kullanır.

CSQINPT örnekleri

CSQ4INST

Örnek veri kümesi: thlqual.SCSQPROC(CSQ4INST), sistem varsayılan aboneliğine ilişkin tanımlamayı içerir.

Bu örnekte nesneyi tanımlamanız gerekir, ancak bunu yalnızca yayınlama/abone olma motoru ilk kez başlatıldığında bir kez yapmanız gerekir. CSQINPT veri kümesinde tanımlama da dahil olmak üzere, bu işlem için en iyi yöntem budur. Kuyruk yöneticisi kapatma ve yeniden başlatma sırasında sürdürür. Nesne adını değiştirmemeniz gerekir, ancak gerektiğinde özniteliklerini değiştirebilirsiniz.

CSQ4INYT

Örnek veri kümesi: thlqual.SCSQPROC(CSQ4INYT), yayınlama/abone olma motoru başlatıldığında çalıştırmak isteyebileceğiniz bir komut kümesi içerir. Bu örnek, Konu ve Abonelik bilgilerini görüntüler.

Diğer

CSQ4DISP görüntü birimi örneği

Örnek veri kümesi: thlqual.SCSQPROC(CSQ4DISP), kuyruk yöneticinizde tanımlı tüm kaynakları görüntüleyen soysal bir DISPLAY komutları kümesi içerir. Bu, depolama sınıfları ve izleme gibi tüm IBM MQ nesnelerine ve tanımlamalarına ilişkin tanımlamaları içerir. Bu komutlar büyük miktarda çıkış üretebilir. Bu örneği, CSQINP2 veri kümesinde ya da CSQUTIL yardımcı programının COMMAND işlevini kullanarak giriş olarak kullanabilirsiniz.

CSQ4INPX örneği

Örnek veri kümesi: thlqual.SCSQPROC(CSQ4INPX), kanal başlatıcısında her başlatıldığında yürütmek isteyebileceğiniz bir dizi komut içerir. Bu örneği kullanmadan önce özelleştirmeniz gerekir; daha sonra kanal başlatıcı için CSQINPX veri kümesine ekleyebilirsiniz.

CSQ4IVPQ ve CSQ4IVPG örnekleri

Örnek veri kümeleri: thlqual.SCSQPROC(CSQ4IVPQ) ve thlqual.SCSQPROC(CSQ4IVPG), kuruluş doğrulama programlarını (IVP ' ler) çalıştırmak için gereken DEFINE komutları kümeleridir.

Bu örnekleri CSQINP2 veri kümesine dahil edebilirsiniz. IVP ' leri başarıyla çalıştırdığınızda, kuyruk yöneticisi yeniden başlatıldığında bunları yeniden çalıştırmanız gerekmez. Bu nedenle, bu örnekleri CSQINP2 bitiştirilmesinde kalıcı olarak tutmanıza gerek yoktur.

CSQ4MSTR ve CSQ4MSRR örnekleri

Bu örnek, kuyruk yöneticisi için başlatılan görev yordamlarıdır: thlqual.SCSQPROC(CSQ4MSTR) ve thlqual.SCSQPROC(CSQ4MSRR).

CSQ4MSRR uses CSQ4INYR in the CSQINP2 concatenation so that important queues are spread across different page sets.

Gerekiyorsa, yeni yaratılan kuyruk yöneticileri için CSQMINI kartını kullanabilmeniz için açıklamaları kaldırabilirsiniz.

V 9.2.0 **CSQ4QMIN örneği**

Örnek bir QMINI veri kümesi (thlqual.SCSQPROC) (CSQ4QMIN).

QMINI veri kümesinin ve **TransportSecurity** stanza 'nın ayrıntıları için QMINI veri kümesi başlıklı konuya bakın.

z/OS Recovery and restart on z/OS

Yeniden başlatma ve kurtarma işlemi için IBM MQ for z/OS özelliklerini öğrenmek üzere bu konudaki bağlantıları kullanın.

IBM MQ for z/OS , yeniden başlatma ve kurtarma için güçlü özelliklere sahiptir. Bir kuyruk yöneticisinin durdurulduktan sonra nasıl kurtarıldığı ve yeniden başlatıldıktan sonra ne olacağı hakkında bilgi için aşağıdaki alt konulara bakın:

- “How changes are made to data in IBM MQ for z/OS” sayfa 244
- “How consistency is maintained in IBM MQ for z/OS” sayfa 246
- “What happens during termination in IBM MQ for z/OS” sayfa 248
- “What happens during restart and recovery in IBM MQ for z/OS” sayfa 249
- “Belirsiz kurtarma birimlerinin çözümlenmesi” sayfa 251
- “Paylaşılan kuyruk kurtarma” sayfa 254

İlgili kavramlar

z/OS IBM MQ for z/OS kurtarma işlemleri

z/OSüzerinde MQSC komutları yayınlayabileceğiniz kaynaklar

İlgili görevler

Yedekleme ve kurtarma planlanması

z/OS Yönetmez/OS

İlgili başvurular

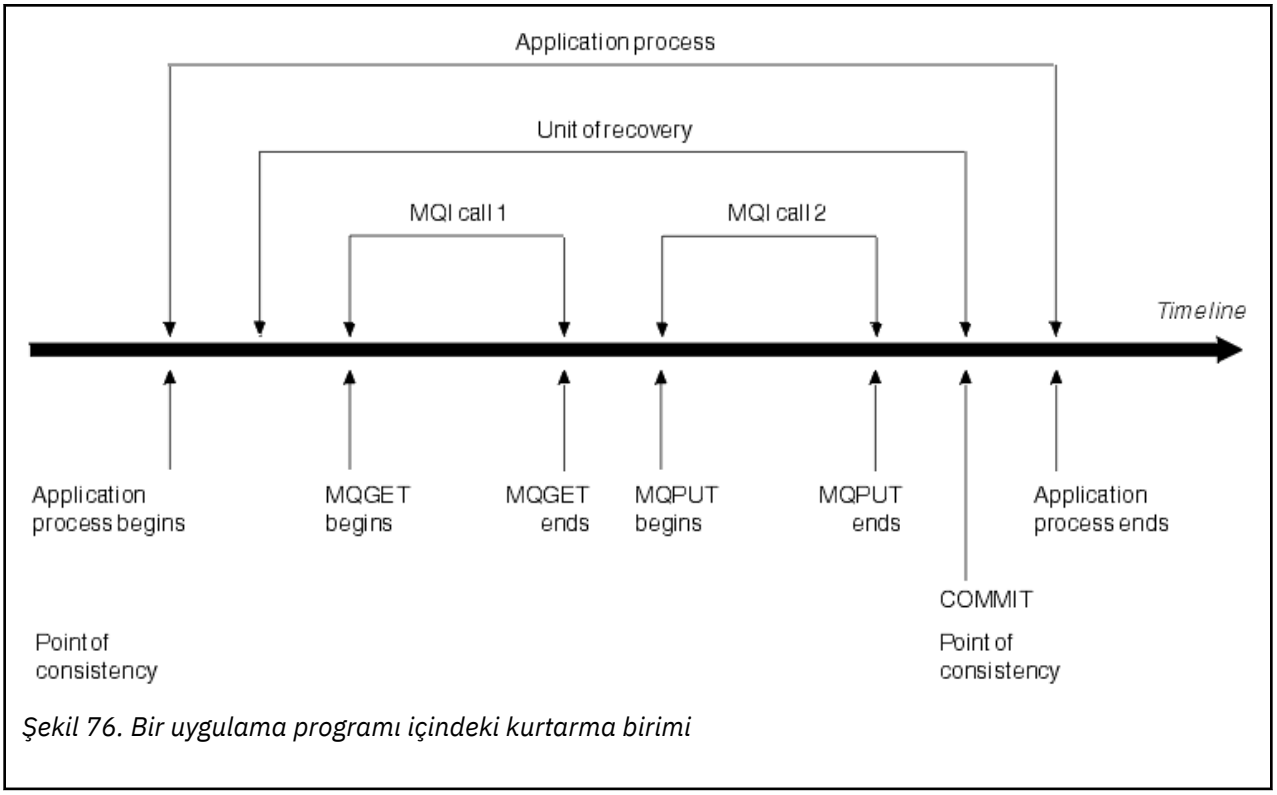
z/OS IBM MQ for z/OSiçin iletiler

z/OS How changes are made to data in IBM MQ for z/OS

IBM MQ for z/OS , tüm verileri tutarlı tutmak için diğer altsistemlerle etkileşim kurmalıdır. Bu konuda, *kurtarma birimleri*, ne oldukları ve bunların *yedeklemeleri*çinde nasıl kullanılırlarsa hakkında bilgiler yer alır.

Kurtarma birimleri

Kurtarma birimi , bir uygulama programı için tek bir kuyruk yöneticisi tarafından gerçekleştirilen işlemdir ve bu, IBM MQ verilerini bir tutarlılık noktasından diğerine dönüştürür. A *tutarlılık noktası* - also called a *syncpoint* or *kesinleştirme noktası* - is a point in time when all the recoverable data that an application program accesses is consistent.



Şekil 76. Bir uygulama programı içindeki kurtarma birimi

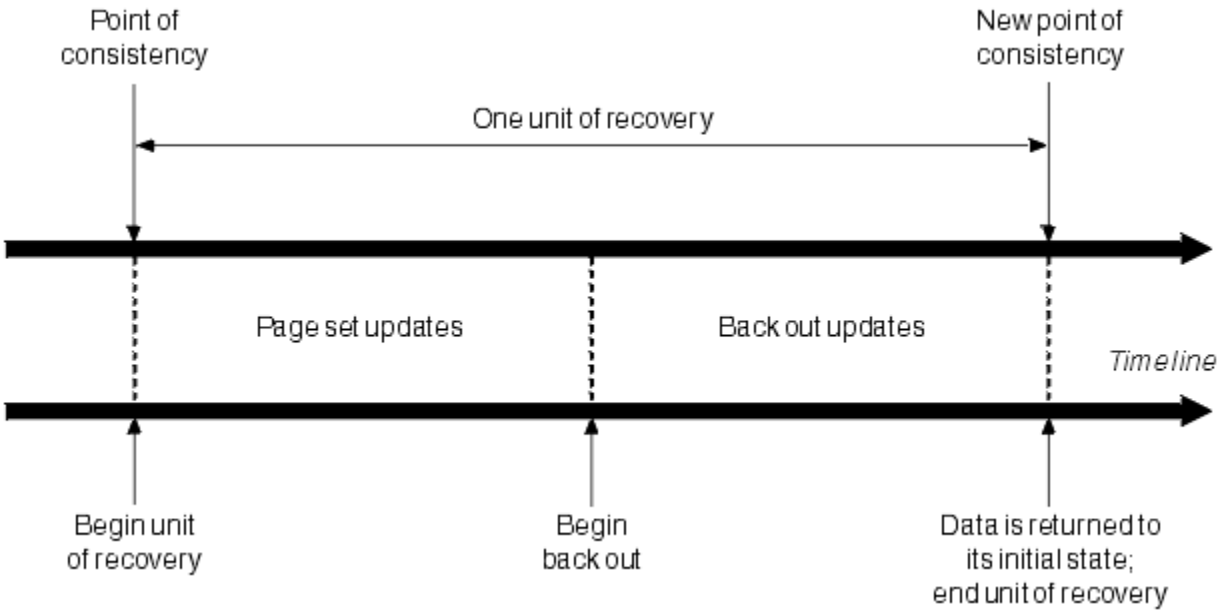
Kurtarma birimi, programın başlangıcından sonra ya da önceki tutarlılık noktasından sonra verilerde yapılan ilk değişiklikle başlar; daha sonra tutarlılık noktası ile sona erer. Şekil 76 sayfa 245 , kurtarma birimleri, tutarlılık noktası ve uygulama programı arasındaki ilişkiyi gösterir. Bu örnekte, uygulama programı 1 ve 2 numaralı MQI çağrılarında kuyruklar üzerinde değişiklik yapar. Uygulama programı birden fazla kurtarma birimi içerebilir ya da yalnızca bir iş birimi içerebilir. Ancak, tüm kurtarma birimi kesinleştirme noktalarında sona erer.

Örneğin, bir banka işlemi bir hesaptan başka bir hesaba para aktarır. İlk olarak, program ilk hesaptan tutarı çıkarır, A hesabı. Sonra, bu ikinci hesaba miktarı ekler, B. Miktar A ' dan çıkarıldıktan sonra, iki hesap tutarsız ve IBM MQ kesinleştirilemez. Miktar, B hesabına eklendiğinde tutarlar. Her iki adım da tamamlandığında, program bir kesinleştirme aracılığıyla bir tutarlılık noktası açıklayabilir ve değişiklikleri diğer uygulama programları için görünür hale getirmektedir.

Bir uygulama programının olağan biçimde sona erdirilmesi, bir tutarlılık noktasına otomatik olarak neden olur. CICS ve IMS programlarında bazı program istekleri de bir tutarlılık noktasına neden olur; örneğin, EXEC CICS SYNCPOINT.

İş yedekleniyor

If an error occurs within a unit of recovery, IBM MQ removes any changes to data, returning the data to its state at the start of the unit of recovery; that is, IBM MQ backs out the work. Olaylar Şekil 77 sayfa 246içinde gösterilir.



Şekil 77. Geri alma işlemi sırasında ortaya çıkan kurtarma birimi

z/OS How consistency is maintained in IBM MQ for z/OS

IBM MQ for z/OS içindeki verilerin toplu iş, CICS, IMSya da TSO ile tutarlı olması gerekir. Bir tanesinde değiştirilen veriler, diğerinde bir değişiklik ile eşleştirilmeli.

Bir sistem değiştirilen verileri bildirmeden önce, diğer sistemin karşılık gelen değişikliği yapabildiğini bilmesi gerekir. Yani, sistemler iletişim kurmalı.

Bir *iki aşamalı kesinleştirme* (örneğin, CICS altında) sırasında, bir altsistem işlemi koordine eder. Bu altsisteme *eşgüdümcü*adı verilir; diğeri *katılımcısıdır*. CICS ya da IMS her zaman IBM MQile etkileşimlerdeki eşgüdümcdür ve IBM MQ her zaman katılımcısıdır. In the batch or TSO environment, IBM MQ can participate in two-phase commit protocols coordinated by z/OS RRS.

Bir *tek aşamalı kesinleştirme* sırasında (örneğin, TSO ya da toplu iş altında), IBM MQ her zaman etkileşimlerdeki eşgüdümcdür ve kesinleştirme işlemini tamamen denetler.

WebSphere Application Server ortamında, JMS oturum nesnesinin anlambilimi tek fazlı ya da iki aşamalı kesinleştirme eşgüdümü kullanılıp kullanılmadığını belirler.

CICS ya da IMSile tutarlılık

IBM MQ ile CICS ya da IMS arasındaki bağlantı, aşağıdaki eşitleme noktası protokollerini destekler:

- Birden çok kaynak yöneticisi tarafından sahip olunan kaynakları güncelleyen işlemler için iki aşamalı kesinleştirme.

  Bu, standart dağıtılmış eşitleme noktası iletişim kuralıdır. Tek aşamalı kesinleştirmeye göre daha fazla günlüğe kaydetme ve ileti akışı içerir.
- Tek bir kaynak yöneticisine ait kaynakları güncelleyen işlemler için tek aşamalı kesinleştirme (IBM MQ).

  Bu protokol, günlüğe kaydetme ve ileti akışları için eniyilenir.
- IBM MQ ' u içeren ancak kuyruk yöneticisinde bir eşitleme noktası gerektiren (örneğin, bir kuyruğa göz atma) hiçbir şey yapmayan tutarlılık noktası (syncpoint) atlama noktası.

Her durumda, CICS ya da IMS , syncpoint yöneticisi olarak işlev görür.

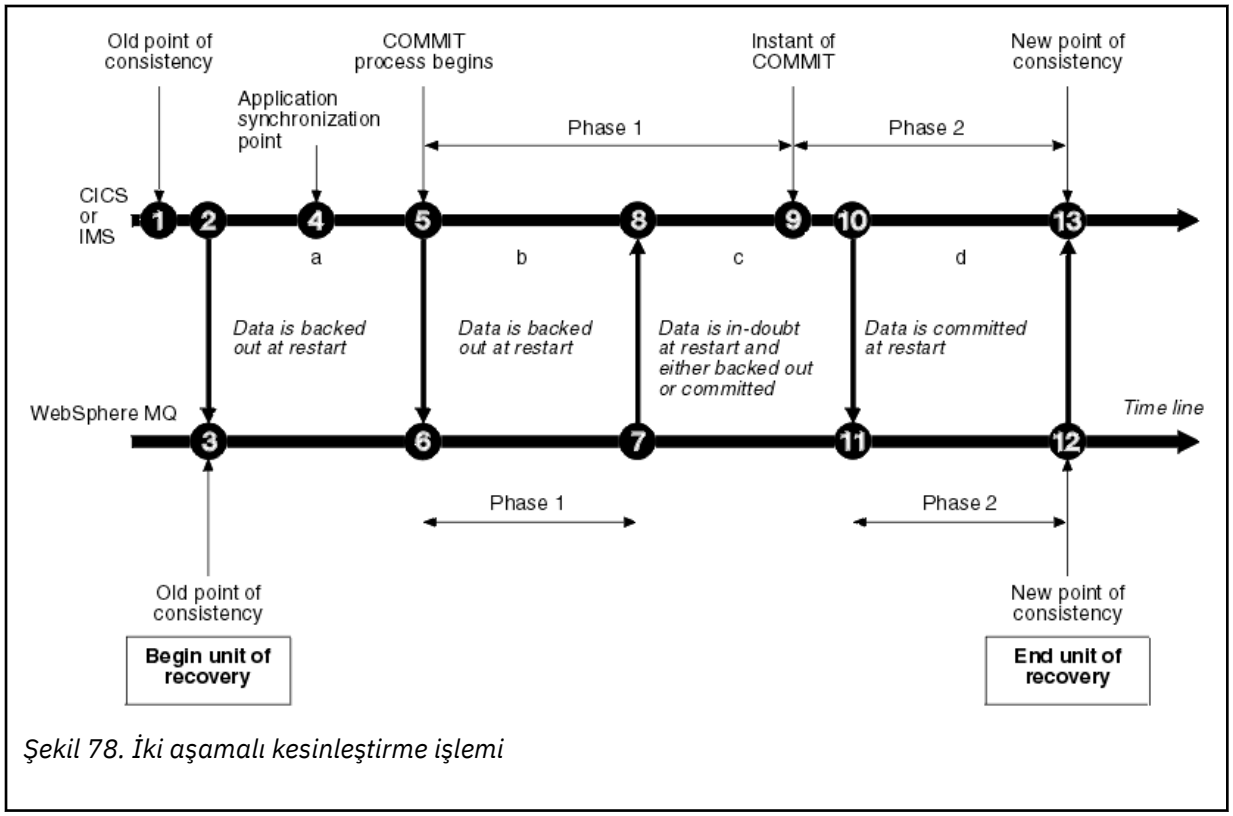
IBM MQ ' in CICS ya da IMS ile iletişim kurmak için kullandığı iki aşamalı kesinleştirmenin aşamaları aşağıdaki gibidir:

1. 1. evre 'de, her sistem, kendi günlüğünde yeterli kurtarma bilgisi kaydetmiş olup olmadığını bağımsız olarak belirler ve bu bilgileri kesinleştirebilir.

   Aşamanın sonunda, sistemler iletişim kurar. Eğer kabul etirlerse, her biri bir sonraki safhaya başlar.
2. 2. evrede, değişiklikler kalıcı kılınabiliyor. Sistemlerden biri 2. aşama sırasında olağandışı biterse, işlem yeniden başlatma işlemi sırasında kurtarma işlemi tarafından tamamlanır.

İki aşamalı kesinleştirme işleminin şekli

Şekil 78 sayfa 247 , iki aşamalı kesinleştirme işlemini gösterir. Events in the CICS or IMS coordinator are shown on the upper line, events in IBM MQ on the lower line.



Şekil 78. İki aşamalı kesinleştirme işlemi

Aşağıdaki kısımlardaki sayılar, şekildeki gösterilerle bağlantılıdır.

1. Koordinatördeki veriler tutarlılığın bir noktasında yer alıyor.
2. Eşgüdümcüdeki bir uygulama programı, bir ileti ekleyerek bir kuyruğu güncellemek için IBM MQ ' i çağırır.
3. Bu işlem, IBM MQiçinde bir kurtarma birimi başlatır.
4. Bir uygulama eşitleme noktasına ulaşılıncaya kadar eşgüdümcüde işleme devam eder.
5. Daha sonra eşgüdümcünün kesinleştirme işlemi başlatılır. CICS programları, kesinleştirmeyi başlatmak için bir SYNCPOINT komutu ya da olağan bir uygulama sonlandırma kullanır. IMS programları, bir CHKP çağrısı, bir SYNC çağrısı, IOPCB ' ye bir GET UNIQUE çağrısı ya da olağan bir uygulama sonlandırma kullanarak kesinleştirmeyi başlatabilir. Kesinleştirme işlemenin 1. aşaması başlar.
6. Eşgüdümcü 1. evre işlemesini başladığından, IBM MQyapar.
7. IBM MQ aşama 1 'i başarıyla tamamlar, bu gerçeği günlüğüne yazar ve koordinatöre bilgi verir.
8. Koordinatör bildirimi alır.
9. Eşgüdümcü, 1. evre işlemlerini başarıyla tamamlar. Her iki altsistem de veri değişikliklerini kesinleştirmeyi kabul eder, çünkü her ikisi de 1. aşamayı tamamlamış ve herhangi bir

hatadan kurtulabilmekte. Eşgüdümcü kayıtları, değişiklikleri yapmak için iki altsistemin kesinleştirilebilecek kesinleştirme kararında yer alan günlük kayıtlarındaki kayıtlardan birini kaydeder.

Koordinatör şu anda işlemin 2. aşamasına başlıyor. Gerçek taahhütten.

10. Koordinatör, IBM MQ ' nin 2. aşamasını başlatmasını bildirir.
11. IBM MQ , 2. aşamanın başlangıcını günlüğe kaydeder.
12. 2. Aşama başarıyla tamamlandı ve şimdi bu, IBM MQiçin yeni bir tutarlılık noktası. IBM MQ daha sonra, eşgüdümcünün 2. aşamasını tamamladığını bildirir.
13. Koordinatör 2. aşama işlemlerini bitiriyor. Her iki altsistem tarafından denetlenen veriler artık tutarlı ve diğer uygulamalar tarafından kullanılabilir.

Olağandışı sona erdirme işleminden sonra tutarlılık sağlanır.

Bir kuyruk yöneticisi olağandışı sona erdirildikten sonra yeniden başlatıldığında, sonlandırma sırasında etkin olan kurtarma birimlerinin kesinleştirilip kesinleştirilmeyeceğini ya da geri alma işleminin yapılıp yapılmayacağını belirlemelidir. Bazı kurtarma birimlerinde IBM MQ , kararı vermek için yeterli bilgiye sahiptir. Diğerleri için ise, bağlantı yeniden kurulduğunda eşgüdümcünden bilgi almak zorunda değildir.

Şekil 78 sayfa 247 , iki aşama içinde dört dönemi gösterir: a, b, c ve d. Bir kurtarma biriminin durumu, sonlandırma işleminin gerçekleşme dönemine bağlıdır. Durum, aşağıdakilerden biri olabilir:

Uçuşta
Kuyruk yöneticisi, 1. aşama (dönem a ya da b) bitirmeden önce sona erdi; yeniden başlatma sırasında, IBM MQ güncellemeleri geri aldı.

Belirsiz
Kuyruk yöneticisi, 1. aşamadan sonra ve 2. aşamaya (dönem c) başlamadan önce sona erdi; yalnızca eşgüdümcü, hatanın kesinleştirmeden önce mi, yoksa sonra mı olduğunu (nokta 9) bilir. Daha önce de olduysa, IBM MQ değişikliklerine geri dönmelidir; daha sonra gerçekleşirse, IBM MQ bu değişiklikleri gerçekleştirmeli ve bunları kesinleştirmelidir. Yeniden başlatma sırasında, IBM MQ bu kurtarma birimini işlemeden önce eşgüdümcünden bilgi almak için bekler.

Kesinleştirmede
Kuyruk yöneticisi kendi faz 2 işleme (dönem d) işlemini başlattıktan sonra sona erdi; bu, kesinleştirilen değişiklikleri yapar.

Arka arkaya
Kuyruk yöneticisi, bir kurtarma birimi yedeklendikten sonra sona erdi, ancak işlem yeniden başlatma sırasında tamamlanmadan (şekilde gösterilmeden) sona erdi, IBM MQ değişiklikleri geri almaya devam eder.

z/OS What happens during termination in IBM MQ for z/OS

Kuyruk yöneticisi STOP QMGR komutuna yanıt olarak olağan bir şekilde sonlandırılır. Kuyruk yöneticisi başka herhangi bir nedenle durursa, sona erdirme olağandışı olur.

Kuyruk yöneticisi sonlandırması sırasında, IBM MQ komutu dahili olarak sorun olarak yayınlar

```
DISPLAY CONN(*) TYPE(CONN) ALL WHERE (APPLTYPE NE SYSTEMAL)
```

Böylece, iş parçacıklarının kuyruk yöneticisinin kapanmayı tamamlamasını engelleyebileceğinden haberdar olun.

SYSTEMAL, SYSTEM ya da CHINIT ile ilgili APPLTYPES ile eşleşiyor; bu nedenle DISPLAY CONN komutu süzgeç uygulama tipleri SYSTEMAL ile eşleşmedi, olağan kapanmayı önleyen iş parçacıklarına ilişkin iş günlüğü bilgilerine geri döner.

Olağan sonlandırma

Olağan bir sona erdirmede, IBM MQ tüm etkinliği sırayla durdurur. Quiesce, force ya da restart kipini kullanarak IBM MQ işlemini durdurabilirsiniz. Etkiler Çizelge 22 sayfa 249içinde verilmiştir.

Çizelge 22. QUIESCE, FORCE ve RESTART kullanılarak sona erdirme

İş parçacığı tipi	QUIES	FORCE	YENİDEN BAŞLATMA
Etkin iş parçacıkları	Tamamlanmak üzere çalıştır	İptal Et	İptal Et
Yeni iş parçacıkları	Başlayabilirler	İzin verilmiyor	İzin verilmiyor
Yeni bağlantılar	İzin verilmiyor	İzin verilmiyor	İzin verilmiyor

Uygulama hala bağlıyken sona erdirme gerçekleşirse, toplu iş uygulamalarına bildirim gönderilir.

CICSile, yürürlükteki iş parçacığı yalnızca kurtarma biriminin sonuna kadar çalıştırılır. CICSile, susturma kipinde bir kuyruk yöneticisinin durdurulması CICS bağdaştırıcısını durdurur ve dolayısıyla etkin bir görev birden çok kurtarma birimi içeriyorsa, görevin tamamlanması için gerekli olmadığı anlamına gelir.

Bir kuyruk yöneticisini zorlamalı ya da yeniden başlatma kipinde durdurursanız, yeni iş parçacıkları ayrılanmaz ve bağlı iş parçacıklarına ilişkin çalışma geriye işlenir. Bu kiplerin kullanılması, kesinleştirme işlemi aşamaları arasında olan iş parçacıkları için belirsiz kurtarma birimleri yaratabilir. IBM MQ , denetleyen CICS, IMSya da RRS altsistemiyle yeniden bağlantı kurduğunda bunlar çözülür.

Bir kuyruk yöneticisini durdurduğunuzda, herhangi bir kipte aşağıdaki adımlar olur:

1. Bağlantılar sona erdirildi.
2. IBM MQ , komutları kabul etmeyi sona erdirir.
3. IBM MQ , sayfa kümelerinde bekleyen tüm güncellemelerin tamamlanmasını sağlar.
4. The DISPLAY USAGE command is issued internally by IBM MQ so that the restart RBA is recorded on the z/OS console log.
5. Kapatma denetim noktası alınır ve BSDS güncellenir.

Susturma kipini belirten sonlandırılmalar, belirsiz kurtarma birimlerini etkilemez. Şüpheye yer olan herhangi bir birim kuşku içinde kalır.

Olağandışı sona erdirme

Olağandışı bir sonlandırma, verileri tutarsız bir durumda bırakabilir, örneğin:

- Bir tutarlılık noktasına ulaşmadan önce bir kurtarma birimi kesintiye uğradı.
- Kesinleştirilen veriler, sayfa kümelerine yazılmadı.
- Kesinleştirilmemiş veriler sayfa kümelerine yazıldı.
- Kesinleştirme işleminin aşama 1 ve 2. evre arasında bir uygulama programı kesintiye uğratılmıştır; bu işlem, kurtarma birimini belirsiz bir şekilde bırakmıştır.

IBM MQ , yeniden başlatma ve kurtarma sırasında olağandışı sonlandırmadan doğan veri tutarsızlıklarını çözer.

z/OS What happens during restart and recovery in IBM MQ for z/OS

IBM MQ , kurtarma günlüğünü ve önyükleme veri kümesini (BSDS) kullanarak, yeniden başlatma sırasında neyin kurtarılması gerektiğini belirler. BSDS, etkin ve arşiv günlüğü veri kümelerini ve günlükteki en son IBM MQ denetim noktasının konumunu tanımlar.

Yeniden başlatma ve kurtarma için giriş

IBM MQ kullanıma hazırlandıktan sonra, kuyruk yöneticisi yeniden başlatma işlemi aşağıdaki gibi gerçekleşir:

- Günlük kullanıma hazırlama
- Geçerli durum yeniden oluşturma
- İleriye ilişkin günlük kurtarma
- Geriye doğru günlük kurtarma
- Kuyruk dizinini yeniden oluşturma

Kurtarma işlemi tamamlandığında:

- Kesinleştirilen değişiklikler veriye yansıtılır.
- Belirsiz etkinlik veriye yansıtılır. However, the data is locked and cannot be used until IBM MQ recognizes and acts on the in-doubt decision.
- Kesintiye uğrayan-uçuş sırasında ve iptal edilen değişiklikler kuyruklardan kaldırıldı. İletiler tutarlı ve kullanılabilir.
- Yeni bir kontrol noktası alındı.
- Kalıcı iletiler içeren dizinlenmiş kuyruklar için yeni dizinler oluşturuldu (“Kuyruk dizinleri yeniden oluşturuluyor” sayfa 251 içinde açıklanmıştır).

İkili BDSS kullanılırsa, IBM MQ , BSDS ' deki zaman damgalarının tutarlılığını denetler:

- BSDS ' nin her iki kopyası da akışıysa, IBM MQ iki saat damgasının eşit olup olmadığını sınar. If they are not, IBM MQ issues message CSQJ120E and terminates. Bu durum, BSDS ' nin iki kopyası ayrı DASD birimlerinde sürdürüldüğünde ve kuyruk yöneticisi durdurulurken birimlerden biri geri yüklendiğinde ortaya gelebilir. IBM MQ , durumu yeniden başlatma sırasında saptar.
- BSDS ' nin bir kopyası ayrılırsa ve günlüğe kaydetme tek bir BSDS ile devam ederse, bir sorun ortaya çıkabilir. BSDS ' nin her iki kopyası tek bir birimde tutulur ve birim geri yüklenirse ya da her iki BSDS kopyası ayrı olarak geri yüklendiyse, IBM MQ geri yükleme işlemi algılamayabilir. Bu durumda, BSDS ' de not not not not not not not unknown to the system.

Yeniden başlatma gerçekleştiğinde toplu iş başvuruları bildirilmez *bundan sonra* , uygulama bir bağlantı isteğinde bulundu.

Kurtarma için gereken günlük aralığını anlama

Yeniden başlatma sırasında, okunması gereken günlük verileri aralığı birçok etmenlere bağımlıdır:

- Olağan dışı bir sona erdirme sırasında, sistemde genellikle çok sayıda eksik iş birimi vardır. Daha önce açıklandığı gibi, yeniden başlatma işlemi, sistemi bir tutarlılık durumuna getirir. Bu durum, hafif iş birimlerinin yedeklenmesi ya da belirsiz iş birimlerindeki kilitlerin kurtarılması olabilir. İş birimi kurtarma birimi, tüm iş birimi günlük kayıtları, giriş/çıkış, belirsiz ve belirsiz iş birimleri için kullanılabilir. IBM MQ , 'eski iş birimleri' olanağını kullanır; böylece, iş birimi kurtarma birimi çok daha küçük bir günlük verileri aralığı kullanılarak gerçekleştirilebilir.
- Olağan dışı bir sona erdirme sırasında, genellikle arabellek havuzu önbelleğinde tutulan çok sayıda kalıcı güncelleme vardır. Henüz diske yazılmadılar. Bu değişiklikler günlüğünden okunmalı ve sayfa kümelerinde tutulan verilere yeniden uygulanmalıdır. Denetim noktasındaki sayfa kümesi kurtarma RBA 'ları, sayfa kümelerini tutarlı bir duruma güncellemek için gereken en düşük günlük RBA' yı tanımlar.
- Sisteme eski sayfa kümeleri tanıtıldıysa, örneğin, bir ortam hatasından kurtarmak için bir sayfa kümesi yedeği devreye sokulduysa, yedekleme işlemi gerçekleştirildiği zamandan itibaren tüm değişiklikler günlükten okunmalıdır. Bu değişiklikler, kurtarılmakta olan sayfa kümesinde tutulan verilere yeniden uygulanır. Sayfa kümesinin 0. sayfasında tutulan sayfa kümesi kurtarma RBA 'ları, bir sayfa kümesinin ortam kurtarma işlemi için gereken en düşük günlük RBA' yı açıklasın.
- Paylaşılan kuyruklarda kalıcı iletiler kullanılıyorsa, kalıcı iletileri tutan CFSTRUCTS ' leri kurtarmak için bir günlük verisi aralığı gereklidir. Bir CFSTRUCT kurtarma işlemi gerçekleştirmek için gereken en eski günlük verileri, eski CFSTRUCT BACKUP zamanından geçmektedir.

Normal çalışma sırasında, bu etmenlerle ilişkili kurtarma günlüğü aralığını görüntülemek için DISPLAY USAGE TYPE (DATASET) komutu kullanılabilir (eski sayfa kümelerinin yeniden tanıtılması nedeniyle

bilgi sağlayamaz). Olağandışı sona erdirme durumunda kuyruk yöneticisi yeniden başlatılacak sorunları önlemek için, DISPLAY KULLANIM TIPINDEN (VERI KüMESI) değer çıkışını düzenli olarak izleyin.

Ayrıca, kuyruk yöneticisi bu etmenlerle ilgili bilgi iletilerini de içerir:

- CSQJ160I ve CSQJ161I , çalışan uzun çalışma birimleri için uyarı sağlar.
- CSQR026I ve CSQR027I , bu uzun çalışma birimlerinin başarıyla kapatılıp kapatılmadığına ilişkin bilgi sağlar.
- CSQE040I ve CSQE041E , yapı yedeklemelerinin eski hale getirildiğini ve sonuç olarak bir RECOVER CFSTRUCT işleminin uzun süreceği uyarısında bulunuyor.

Hangi uygulamanın uzun bir çalışma birimine sahip olduğu saptanıyor

Uygulamanın uzun çalışan iş birimi ile belirlenmesi mümkündür. Bunu yapmak için DISPLAY CONN komutunu kullanıyorsunuz.

DISPLAY CONN komutu, kuyruk yöneticisine bağlı tüm uygulamalar için bağlantı bilgilerini döndürür. Bu bilgiler, uzun süredir çalışan bir iş biriminin hangi uygulama (lar) olduğunu belirlemenize yardımcı olacak ek bilgilerle birlikte, ek bilgi sağlar. DISPLAY CONN komutu tarafından döndürülen bilgiler, DISPLAY QSTATUS komutunun döndürdüğü bilgilere benzer, ancak ana fark, belirli bir nesneyle ilişkilendirilecek bağlantıların ayrıntıları yerine, belirli bir bağlantıya ilişkin işlem bilgileri ve nesnelerle ilgili işlem bilgilerini görüntüler.

Bağlı her uygulama için, DISPLAY CONN komutu aşağıdaki bilgileri döndürür:

- Bağlantı tanıtıcısı ve PID de içinde olmak üzere temel bilgiler.
- İşlemin yaratıldığı tarih ve saat de içinde olmak üzere (yani, ilk MQGE/PUT ' un syncpoint altında yapıldığında) ve işlemin günlüğe yazıldığı tarih de dahil olmak üzere, bu bağlantıya ilişkin işlemsel bilgiler.
- Hangi uygulamanın uzun süredir çalışan bir iş birimi olduğunu gösteren günlük zamanı bilgileri.
- Bağlantının şu anda açık olduğu tüm nesnelerin listesi. Her nesneye ilişkin ayrıntılar ayrı bir ileti olarak döndürülür; Bağlantı Tanıtıcısı bir anahtar olarak kullanılır. Kuyruklar ve kuyruk yöneticileri gibi farklı nesne tipleri olduğu için, nesneyle görüntülenen bilgiler nesne tipine özgüdür.

Kuyruk dizinleri yeniden oluşturuluyor

İletilerin sıralı olarak alınmadığı bir kuyruktan MQGET işlemlerinin hızını artırmak için, IBM MQ ' un ileti ya da ilinti tanıtıcılarının dizinini ya da o kuyruktaki tüm iletiler için groupid dizinini korumasını istediğinizi belirtebilirsiniz.

Bir kuyruk yöneticisi yeniden başlatıldığında, bu dizinler her kuyruk için yeniden oluşturulur. Bu, yalnızca kalıcı iletiler için geçerlidir; kalıcı olmayan iletiler yeniden başlatma sırasında silinir. Dizinlenmiş kuyruklarınız çok sayıda kalıcı ileti içeriyorsa, kuyruk yöneticisini yeniden başlatmak için gereken süre bu kadar artar.

You can choose to have indexes rebuilt asynchronously to queue manager startup by using the QINDXBLD parameter of the CSQ6SYSP macro. QINDXBLD=NOWAIT seçeneğini ayarlarsanız, IBM MQ dizinlerin yeniden oluşturulmasını beklemeden yeniden başlar.

z/OS Belirsiz kurtarma birimlerinin çözümlenmesi

IBM MQ başka bir kaynak yöneticisiyle bağlantısını kaybederse, tipik olarak, yeniden başlatma sırasında tüm tutarsız nesneleri kurtarma girişiminde bulunur.

IBM MQ , CICS, IMSya da RRS bağlantısını kaybederse, olağan durumda yeniden başlatma sırasında tüm tutarsız nesneleri kurtarma girişiminde bulunur. Belirsiz kurtarma birimlerinde çözülmesi gereken bilgiler, eşgüdümleme sisteminden gelmelidir. Sonraki kısımlarda, farklı ortamlara ilişkin çözüm süreci anlatılıyor.

- Belirsiz kurtarma birimlerinin CICS' den çözümlenmesi

- Belirsiz kurtarma birimlerinin IMS' den çözümlenmesi
- RRS ' den belirsiz kurtarma birimlerinin nasıl çözümlendiğini
- Bir GROUP birimiyle kurtarma yok etme birimi ile ilgili olarak belirsiz kurtarma birimleri çözümlendi

Belirsiz kurtarma birimlerinin CICS' den çözümlenmesi

Bazı durumlarda, CICS , belirsiz kurtarma birimlerinde çözümlemek için IBM MQ işlemini çalıştıramaz. Bu gerçekleştiğinde, IBM MQ aşağıdaki iletilerden birini gönderir:

- CSQC404E
- CSQC405E
- CSQC406E
- CSQC407E

Bu iletiyi CSQC408Iiletisi izliyor.

Bu iletilerin ne anlama gelimesine ilişkin ayrıntılar için IBM MQ for z/OS iletileri, tamamlama, ve neden kodları elkitabına bakın.

Belirsiz birimlerin çözümünde CICS kaynakları geçerli değildir. CICS , kurtarma eşgüdümü denetiminde ve yeniden başlatıldığında, kesinleştirmenin başlangıcındaki bir günlük kaydı olup olmadığına bağlı olarak, her birimi otomatik olarak kesinleştirir ya da yedekler arasında geri gelir. The existence of in-doubt objects does not lock CICS resources while IBM MQ is being reconnected.

One of the functions of the CICS adapter is to keep data synchronized between CICS and IBM MQ. If a queue manager abends while connected to CICS, it is possible for CICS to commit or back out work without IBM MQ being aware of it. Kuyruk yöneticisi yeniden başlatıldığında, bu iş *belirsiz*olarak tertiplenmektedir.

IBM MQ , CICS ile bağlantı yeniden başlatılıncaya ya da yeniden bağlanıncaya kadar, bu belirsiz kurtarma birimleri (IBM MQ kaynakları üzerinde yapılan değişiklikleri kesinleştirmek ya da geri almak) çözemez.

CICS bağdaştırıcısının başlatılması sırasında belirsiz kurtarma işlemi için bir işlem başlatılır. Bu işlem, bağdaştırıcı, belirsiz durumdan kurtarma birimlerinin bir listesini istediğinde başlar. O zaman:

- Bu bağdaştırıcı, IBM MQ'tan bu bağlantı tanıtıcısı için belirsiz kurtarma birimlerinden oluşan bir liste alır ve bunları çözüm için CICS ' e geçirir.
- CICS , bu listedeki girdileri, girdilerle kendi günlüğündeki girdilerle karşılaştırır. CICS , her belirsiz kurtarma birimi için hangi işlemi aldığını kendi listesinden belirler.

Çözümlenen tüm birimler için, IBM MQ kuyrukları gerektiği şekilde günceller ve ilgili kilitleri serbest bırakır. Çözümlenmemiş birimler yeniden başlatıldıktan sonra kalabilir. Bunları Yönetme IBM MQ for z/ OSiçinde açıklanan yöntemler ile çözer.

Belirsiz kurtarma birimlerinin IMS' den çözümlenmesi

IMSiçindeki belirsiz kurtarma birimleri çözülemediği DL/I kaynaklarını etkilemez. IMS , kurtarma eşgüdümü denetiminde ve yeniden başlatıldığında, otomatik olarak kesinlediğinde ya da yedekler tamamlanmamış DL/I ' yi yedeklemektedir. Çevrimiçi bölgeler için kesinleştirme ya da geri gönderme kararı (hızlı olmayan yol), X'3730 've X'3801' IMS günlük kaydı tiplerinin varlığı ya da yokluğu üzerinedir. The existence of in-doubt units of recovery does not imply that DL/I records are locked until IBM MQ connects.

Kuyruk yöneticisi yeniden başlatılırken, IBM MQ , belirsiz kurtarma birimlerinden oluşan bir liste yapar. IMS , kendi yeniden kurtarma girişlerinin (RREs) kendi listesini oluşturur. RREs 'ler, tüm girdiler çözülünceye kadar IMS denetim noktalarında günlüğe kaydedilir.

During reconnection of an IMS region to IBM MQ, IMS indicates to IBM MQ whether to commit or back out units of work marked in IBM MQ as in doubt.

Belirsiz birimler çözüldüğünde:

1. IBM MQ , kesinleştirme için bir girdi işaretlediğine ve IMS ' un geriletileceğini tanırsa, IBM MQ iletisi CSQQ010Eiletisini verir. IBM MQ , bu iletinin IBM MQ ile IMSarasında tüm tutarsızlıkları için bu iletiyi yayınlar.
2. IBM MQ ' de kalan belirsiz birimler varsa, bağdaştırıcı ileti CSQQ008Iiletisi verir.

Çözümlenen tüm birimler için, IBM MQ kuyrukları gerektiği şekilde günceller ve ilgili kilitleri serbest bırakır.

IBM MQ , çözülmemiş olan belirsiz iş üzerinde kilitleri korur. Bu, önemli kilitlerin tutulması durumunda sistemde birikim oluşmasına neden olabilir. The connection remains active so you can resolve the IMS RREs. Yönetme IBM MQ for z/OS' ta açıklanan yöntemlere göre belirsiz durumda olan iş parçacıklarını kurtarın.

IMS soğuk başlatma gibi bir yazılım ya da işletim sorunu olmadıkça, tüm belirsiz işler çözümlenmesi gerekir. IMS denetim bölgesi tarafından belirsiz bir şekilde çözülmesi iki durumda gerçekleşir:

1. At the start of the connection to IBM MQ, during which resolution is done synchronously.
2. Bir program sona erdiğinde, çözme işlemi zamanuyumsuz olarak gerçekleştirilir.

Belirsiz kurtarma birimlerinin RRS ' den çözümlenmesi

RRS bağdaştırıcısının işlevlerinden biri, verileri IBM MQ ile diğer RRS katılımcısı kaynak yöneticileri arasında eşitlenmiş tutmadır. IBM MQ , kesinleştirmenin birinci aşaması tamamlanınca bir hata oluşursa ve RRS ' den (kesinleştirme koordinatörü) karar bekliyorsa, kurtarma birimi belirsiz duruma girer.

RRS ve IBM MQarasında iletişim yeniden kurulduğunda, RRS, kesinleştirmenin başlangıcındaki bir günlük kaydı olup olmadığına bağlı olarak, her kurtarma birimini otomatik olarak kesinleştirir ya da yedekler. IBM MQ , RRS bağlantısı yeniden kuruluncaya kadar, bu belirsiz kurtarma birimleri (IBM MQ kaynakları için yapılan değişiklikleri kesinleştirmek, geri almak ya da yedeklemek) çözemez.

Bazı durumlarda, RRS, belirsiz kurtarma birimlerinde çözümlenemez. Bu gerçekleştiğinde, IBM MQ , z/OS konsoluna aşağıdaki iletilerden birini gönderir:

- CSQ3011I
- CSQ3013I
- CSQ3014I
- CSQ3016I

Bu iletilerin ne anlama gelimesine ilişkin ayrıntılar için IBM MQ for z/OS iletileri, tamamlama, ve neden kodları elkitabına bakın.

Çözümlenen tüm kurtarma birimleri için IBM MQ , kuyrukları gerektiği şekilde günceller ve ilgili kilitleri serbest bırakır. Çözümlenmemiş kurtarma birimleri yeniden başlatıldıktan sonra kalabilir. Bunları Yönetme IBM MQ for z/OSiçinde açıklanan yöntemle çözer.

Bir GROUP birimiyle kurtarma yok etme birimi ile ilgili olarak belirsiz kurtarma birimleri çözümlendi

GROUP birimi kurtarma yok etme birimi olan belirsiz hareketler, hareket eşgüdümcüsü tarafından GROUPUR kuyruk yöneticisi özniteliğinin etkin olduğu kuyruk paylaşım grubunda (QSG) herhangi bir kuyruk yöneticisi tarafından çözülebilir. Bir hareket eşgüdücünün yeniden bağlandığı zaman, genellikle belirsiz hareketlerin bir listesini ister ve daha sonra, bilgileri günlüklerinden bilgileri kullanarak çözer.

Bir hareket eşgüdümcüsü, bir grup kurtarma atma birimiyle bağlantılı olduğunda, belirsiz hareketlerin listesini ister; döndürülen liste, kuyruk paylaşım grubu boyunca var olan bir grup kurtarma yok etme birimine sahip tüm belirsiz hareketlerden oluşur. Bu liste, hangi kuyruk yöneticisinin başlatıldığı hangi kuyruk yöneticisinde başlatıldığına bağlı değildir. Böyle bir isteği işleyen bir kuyruk yöneticisi, SYSTEM.QSG.UR.RESOLUTION.QUEUE. Daha sonra kuyruk yöneticisi, etkin olmayan tüm belirsiz

hareketleri tanımlamak için, etkin olmayan kuyruk yöneticilerinin günlüklerini son denetim noktasından okur.

Bir hareket eşgüdümcüsü belirsiz bir hareketin çözülmesini istediğinde, bağlı olduğu kuyruk yöneticisi hareketin kendisi için kaynaklanıp kaynaklanmadığını belirtir. Bu durumda, işlemin bir QMGR birimi kurtarma yok etme birimine sahip işlemlerle aynı şekilde çözülüp giderilmeyeceğini belirtir. İşlem QSG ' de başka bir etkin kuyruk yöneticisinde oluşturulduysa, çözümü tamamlama isteği, SYSTEM.QSG.UR.RESOLUTION.QUEUE. İşlemin, QSG ' de etkin olmayan bir kuyruk yöneticisinde oluşturulduğu durumda, herhangi bir paylaşılan kuyruk çalışması hemen çözülür ve kalan özel kuyruk çalışmasını çözme isteği SYSTEM.QSG.UR.RESOLUTION.QUEUE. Etkin olmayan kuyruk yöneticisi, yeni iş kabul etmeden önce bu isteği başlatma sırasında işleme sokardı. Bu senaryoda, özgün kuyruk yöneticisinin günlükleri, kurtarma biriminin yeniden başlatılıncaya ve isteği işleme konuncaya kadar kuşku içinde olduğunu gösterir.

z/OS Paylaşılan kuyruk kurtarma

Kuyruk paylaşım grubu ortamındaki çeşitli bileşenlerin IBM MQ kurtarmasını ve dayanıklılığını anlamak için bu konuyu kullanın.

İşlemsel kurtarma

Bir uygulama MQBACK çağrısı yaptığında ya da olağandışı sonlandığında (örneğin, bir EXEC CICS ROLLBACK ya da bir IMS abend nedeniyle), kuyruk yöneticisinde saklanan iş parçacığı düzeyinde bilgiler, hareket halindeki iş biriminin geriye işlenmesini sağlar. Paylaşılan kuyruklardaki eşitleme noktası içindeki MQPUT ve MQGET işlemleri, paylaşılmayan kuyruklardaki güncellemelerle aynı şekilde geriye işlenir.

Eşdüzey kurtarma

Bir kuyruk yöneticisi başarısız olursa, bağlı olduğu bağlaşım olanağı yapılarıyla olağandışı bir şekilde bağlantısı kesilir. z/OS yönetim ortamı ile bağlaşım olanağı arasındaki bağlantı başarısız olursa (örneğin, fiziksel bağlantı arızası ya da bir bağlaşım olanağının ya da bölümün kapanması), kuyruk yöneticisi ile ilgili bağlaşım olanağı yapıları arasındaki bağlantının olağandışı bir şekilde sonlandırılması olarak da algılanır. Aynı yapıya bağlı kalan aynı kuyruk paylaşım grubundaki diğer kuyruk yöneticileri, olağandışı bağlantıyı saptar ve bu yapıda başarısız olan kuyruk yöneticisi için *eşdüzey kurtarma* işlemini başlatma girişiminde bulunurlar. Bu kuyruk yöneticilerinden yalnızca biri eşdüzey kurtarma işlemini başarıyla başlatır, ancak diğer tüm kuyruk yöneticileri, başarısız olan kuyruk yöneticisinin sahip olduğu iş birimlerinin kurtarılmasında işbirliği yapar.

Bir yapıya bağlı eşler olmadığında bir kuyruk yöneticisi başarısız olursa, başka bir kuyruk yöneticisi o yapıya bağlandığında ya da başarısız olan kuyruk yöneticisi yeniden başlatıldığında kurtarma gerçekleştirilir.

Eşdüzey Kurtarma (Peer Level Recovery; PLR) olarak adlandırılan eşdüzey kurtarma, yapı temelinde bir yapıda gerçekleştirilir ve tek bir kuyruk yöneticisinin aynı anda birden fazla yapının kurtarılmasına

katılması mümkündür. Ancak, farklı yapıların kurtarılmasında işbirliği yapan eşler kümesi, hata sırasında hangi kuyruk yöneticilerinin farklı yapılara bağlandığına bağlı olarak değişebilir.

Başarısız olan kuyruk yöneticisi yeniden başlatıldığında, hata sırasında bağlı olduğu yapılara yeniden bağlanır ve eşdüzey kurtarma tarafından kurtarılmayan çözümlenmemiş iş birimlerini kurtarır.

Eş kurtarma, çok aşamalı bir işlemdir. İlk aşamada, devam eden aşamanın ötesine geçen iş birimleri kurtarılabilir; bu, kesinleştirilmekte olan iş birimlerine ilişkin iletilerin kesinleştirilmesini ve belirsiz iş birimlerine ilişkin iletilerin kilitlenmesini içerebilir. İkinci aşamada, başarısız olan kuyruk yöneticisinde kendilerine karşı etkin iş parçacıkları olan kuyruklar denetlenir, devam eden iş birimleriyle ilgili kesinleştirilmemiş iletiler geriye işlenir ve başarısız olan kuyruk yöneticisindeki paylaşılan kuyruklardaki etkin tanıtıcılara ilişkin bilgiler ilk durumuna getirilir. Bu, IBM MQ ' in, başarısız olan kuyruk yöneticisinin giriş için paylaşılan bir kuyruğu açık olduğuna ilişkin göstergeleri ilk durumuna getirmesi ve diğer etkin kuyruk yöneticilerinin kuyruğu giriş için açmalarına izin vermesi anlamına gelir.

Paylaşılan kuyruk tanımlamaları

Paylaşılan bir kuyruğun özniteliklerini gösteren kuyruk nesneleri, kuyruk paylaşım grubu tarafından kullanılan paylaşılan Db2 havuzunda tutulur. IBM MQ nesnelerini tutmak için kullanılan Db2 çizelgelerinin yedeklenmesi ve kurtarılması için yeterli yordamın bulunduğundan emin olun. IBM MQ CSQUTIL yardımcı programını, Db2içinde saklanan paylaşılan kuyruk ve grup tanımlamaları da içinde olmak üzere IBM MQ nesnelerini yeniden tanımlamak üzere bir kuyruk yöneticisinde yeniden yürütmek üzere MQSC komutları yaratmak için de kullanabilirsiniz.

Günlük Kaydı

Kuyruk paylaşım grupları, paylaşılan kuyruklardaki iletiler kuyruk yöneticisi günlüklerine kaydedilebildiği için kalıcı iletileri destekleyebilir.

Bağlaşım olanağı ve yapı arızaları

Bir bağlaşım olanağı (CF) yapısı için bildirilebilecek iki tip hata vardır: yapı arızası ve bağlantı kaybı. Veri paylaşımı için Sysplex hizmetleri (XES), IBM MQ ' e bir CF yapısı arızası ya da bir yapı hatası olayı ile ilgili bilgi verir. XES bir bağlantı kaybı olayı oluşturursa bu, yapıyla ilgili bir sorun olduğunu göstermez, yapıyla iletişim kurmak için kullanılabilir bir bağlantı olmayabilir. Tüm kuyruk yöneticilerinin yapı için bağlantı kaybı almaması olasıdır; bu, CF ' ye yönelik bağlantıların yapılandırmasına bağlıdır. Bağlanırlık kaybı olayı, işletmen komutları nedeniyle de alınabilir; örneğin, VARY PATH OFFLINE ya da CONFIG CHP OFFLINE.

IBM MQ tarafından kullanılan CF yapıları, sistem tarafından yönetilen çift yönlülük kullanacak şekilde yapılandırılabilir. Bu, tek bir hata oluşursa, sistem tarafından yönetilen hata durumunda yedek sisteme geçiş işleminin bir yapının arızalanmasını ya da bağlantı kaybını gizlediği ve kuyruk yöneticisine arızanın bildirilmediği anlamına gelir. Çift yönlü bir yapının ya da bağlantının her iki örneğinin de başarısız olması durumunda, kuyruk yöneticisi uygun olayı alır ve tek yönlü bir yapının başarısızlık olayıyla aynı şekilde işler. Kuyruk yöneticisinin olayları nasıl işleyeceğine ilişkin ayrıntılar için Senaryolarkonusuna bakın.

Olası olmayan bir CF ya da yapı arızası durumunda, etkilenen uygulama yapılarında saklanan kalıcı olmayan iletiler kaybolur. RECOVER CFSTRUCT komutunu kullanarak kalıcı iletileri kurtarabilirsiniz. Kurtarılabilir bir uygulama yapısı başarısız olduysa, yapı kurtarılıncaya kadar bu yapıya ilişkin diğer uygulama etkinlikleri önlenir.

Bir CF yapısını makul bir süre içinde kurtarabildiğinizden emin olmak için, BACKUP CFSTRUCT komutunu kullanarak sık sık yedekleme yapın. Yedeklemeleri, kuyruk paylaşım grubundaki herhangi bir kuyruk yöneticisinde gerçekleştirmeyi seçebilir ya da tüm yedeklemeleri gerçekleştirmek için bir kuyruk yöneticisini ayırmayı seçebilirsiniz. Düzenli olarak alındığından emin olmak için yedeklerin alınması sürecini otomatikleştirin.

Her yedek, yedeklemeyi alan kuyruk yöneticisinin etkin günlük veri kümesine yazılır. Paylaşılan kuyruk Db2 havuzu, yedeklenmekte olan CF yapısının adını, yedeklemeyi yapan kuyruk yöneticisinin adını, o kuyruk yöneticisinin günlüğündeki bu yedekleme için RBA aralığını ve yedekleme süresini kaydeder.

Denetim yapısı, herhangi bir uygulama yapısı arızası sırasında paylaşılan kuyruklardaki tamamlanmamış iş birimlerine ilişkin bilgileri içerir; bu nedenle, denetim yapısı RECOVER CFSTRUCT işlemi sırasında kullanılabilir olmalıdır. Denetim yapısı başarısız olduysa, RECOVER CFSTRUCT komutunu vermeden önce, kuyruk paylaşım grubundaki tüm kuyruk yöneticilerinin denetim yapısı girişlerini yeniden oluşturmaları gerekir.

Kuyruk yöneticileri, yönetim yapısı girişlerini otomatik olarak ve sonlandırmadan yeniden oluşturur. Hata sırasında bir kuyruk yöneticisi çalışmıyorsa, denetim yapısı girişleri, aynı ya da daha yüksek düzeyde çalışan kuyruk paylaşım grubundaki başka bir kuyruk yöneticisi tarafından yeniden oluşturulabilir.

Bir uygulama yapısını kurtarmak için, kurtarma işlemini gerçekleştirmek istediğiniz kuyruk yöneticisine RECOVER CFSTRUCT komutunu verin. Tek bir CF yapısını kurtarabilir ya da aynı anda birden çok CF yapısını kurtarabilirsiniz. Kuyruk paylaşım grubundaki herhangi bir kuyruk yöneticisini kullanarak kurtarabilirsiniz; bu, yedeklemeyi gerçekleştiren ya da daha önce başarısız olan yapıya bağlı olan bir kuyruk yöneticisi olmak zorunda değildir.

RECOVER CFSTRUCT komutu, Db2 havuz bilgileriyle bulunan yedeği kullanır (bu nedenle Db2 , kurtarma işleminin gerçekleştirildiği kuyruk yöneticisinde bulunmalıdır) ve bunu hata noktasına kadar kurtarır.

RECOVER CFSTRUCT komutu, bu işlemi, yedekleme başlangıcı ile başarısızlık zamanı arasında bir MQPUT ya da MQGET işlemi gerçekleştiren kuyruk paylaşım grubundaki her kuyruk yöneticisindeki günlük kayıtlarını CF yapısıyla eşlenen herhangi bir paylaşılan kuyruğa uygulayarak yapar. Sonuçta ortaya çıkan günlüklerin birleştirilmesi, yedeklemeden bu yana katılan kuyruk yöneticileri tarafından yazılan tüm günlük verileri okunduğundan, önemli miktarda günlük verilerinin okunmasını gerektirebilir. Özellikle yedekleme içinde büyük iletiler varsa, sık sık (örneğin, saatlik) yedekleme yapmanız önemle önerilir.

Yapı hatası senaryoları

Senaryolar

CF yapısına ilişkin bir hata bildirilirse, bağlı kuyruk yöneticileri tarafından yapılan işlem aşağıdakilere bağlıdır:

- z/OS XES bileşeni tarafından IBM MQ' e bildirilen arıza tipi.
- Yapı tipi (uygulama ya da denetim)
- Kuyruk yöneticisi düzeyi
- IBM MQ CFSTRUCT nesnesinin CFLEVEL (2, 3, 4 ya da 5). Bu, CFCC mikrokodunun CFLEVEL değil)
- CFLEVEL (5) düzeyindeki bir IBM MQ CFSTRUCT nesnesinin RECAUTO özniteliği

Aşağıdaki senaryolarda, yönetim yapısı için bir hata bildirildiğinde ne olduğu açıklanmaktadır:

- Yönetim yapısı için bir yapı hatası olayı alınırsa, kuyruk yöneticisi sonlandırılmadan yapı yeniden yerleştirilir ve otomatik olarak yeniden oluşturulur. Bir kuyruk yöneticisi yapıya bağlanmayı denediğinde, yapıya ilişkin yeni bir eşgörünüm XES tarafından ayrılır.

  Kuyruk yöneticisi yapının yeni yönetim ortamına bağlandığında, kuyruk yöneticisi kendi girişlerini yapıya yazar. Bu işlem kuyruk yöneticisi tarafından gerçekleştirilir ve XES yeniden oluşturma işleminin bir parçası değildir.

  Hata sırasında bir kuyruk yöneticisi çalışmıyorsa ya da yönetim yapısının bir kısmının kurtarılması tamamlanmadan önce kuyruk yöneticisi sona ererse, denetim yapısı girişleri kuyruk paylaşım grubundaki başka bir kuyruk yöneticisi tarafından yeniden oluşturulabilir.

  Bir kuyruk yöneticisinin denetim yapısı girişleri yalnızca aynı düzeyde ya da daha yüksek düzeylerde çalışan başka bir kuyruk yöneticisi tarafından yeniden oluşturulabilir. Bir kuyruk yöneticisinin denetim yapısı girişleri kuyruk paylaşım grubundaki başka bir kuyruk yöneticisi tarafından yeniden oluşturulamazsa, kuyruk yöneticisini yeniden başlatarak yapının bir kısmının yeniden oluşturulmasını tamamlaması için yeniden başlatın.

  Tüm kuyruk yöneticilerine ilişkin denetim yapısı girişleri yeniden oluşturuluncaya kadar bazı işlemler askıya alınır. Askıya alınan işlemler şunlardır:

  – Paylaşılan kuyrukların açılması ve kapatılması.

– Kurtarma birimleri kesinleştiriliyor ya da geri çekiliyor.
– Kuyruk yöneticisine bağlanan ya da kuyruk yöneticisiyle bağlantıyı kesen diziselleştirilmiş uygulamalar.
– Bir uygulama yapısı yedekleniyor ya da kurtarılıyor.

Kuyruk yöneticisine önceden bağlı olan diziselleştirilmiş uygulamalar işlemeye devam edebilir. Diziselleştirilmiş uygulamalar MQCNO_SERIALIZE_CONN_TAG_QSG ya da MQCNO_RESTRICT_CONN_TAG_QSG parametreleriyle bağlanmayı denerken MQRC_CONN_TAG_NOT_USABLE dönüş kodunu alır.

Kuyruk yöneticisine ilişkin denetim yapısı girişleri yeniden oluşturulduğunda, askıya alınan işlemler sürdürülür.

Aşağıdaki senaryolarda, bir uygulama yapısına ilişkin bir hata bildirildiğinde ne olacağı açıklanmaktadır:

- Bir uygulama yapısı için bir yapı hatası olayı alınırsa ve CFLEVEL 1 ya da 2 ise, kuyruk yöneticisi sona erer. Kuyruk yöneticisini yeniden başlatın. Yapıya yeniden bağlanmayı deneyen ilk kuyruk yöneticisi, XES ' in yapının yeni bir eşgörünümünü ayırmasına neden olur.
- Bir uygulama yapısı için bir yapı hatası olayı alınırsa ve CFLEVEL 3, 4 ya da 5 ise, yapıya bağlı kuyruk yöneticileri çalışmaya devam eder. Hatalı yapıdaki kuyrukları kullanmayan uygulamalar normal işlemeye devam edebilir.

  Ancak, başarısız olan yapıdaki kuyruklarda işlem yapmayı deneyen uygulamalar, yapı başarıyla yeniden oluşturuluncaya kadar bir MQRC_CF_STRUC_FAILED hatası alır ve bu noktada uygulama kuyrukları yeniden açabilir.

  Yapı yeniden oluşturma, RECAUTO (YES) ile tanımlanan CFLEVEL (5) uygulama yapıları için otomatik olarak başlatılır. Tersi durumda, yapı RECOVER CFSTRUCT komutu verildiğinde yeniden oluşturulur.

Bağlaşım olanağı bağlanırlığı arızalarına dayanıklılık

Bağlaşım olanağı bağlanırlığı hatalarına dayanıklılık nedir?

Bağlaşım olanağı bağlanırlık hatalarına dayanıklılık, bir kuyruk paylaşım grubundaki kuyruk yöneticilerinin, bir bağlaşım olanağı yapısına bağlantı kaybını sonlandırmadan tolere edebilme yeteneğini ifade eder. Bu işlev, paylaşılan kuyruklara mümkün olan en kısa sürede yeniden erişim elde etmek için daha iyi bağlanırlıkla başka bir bağlaşım olanağındaki yapıyı yeniden oluşturmayı da dener.

Kısmi bağlantı kaybı nedir?

IBM MQ , kısmi bağlanırlık kaybını, sistem tarafından erişilen yapının ayrıldığı, ancak sysplex içindeki en az bir sistemin aynı bağlaşım olanağına bağlanırlığı sürdürdüğü bağlaşım olanağına bağlanırlığını kaybettiği bir durum olarak tanımlar.

Toplam bağlantı kaybı nedir?

IBM MQ , toplam bağlanırlık kaybını, sistem şebekesindeki hiçbir sistemin bağlaşım olanağına ve bu olanak içinde ayrılmış yapıya bağlanamadığı bir durum olarak tanımlar.

Bu işlevi neden etkinleştirebilirsiniz?

Bağlaşım olanağı bağlanırlık hatalarına dayanıklılık, IBM MQkullanılabilirliğini artırarak, bir kuyruk yöneticisi bir ya da daha çok bağlaşım olanağı yapısına bağlanırlığını kaybettikten sonra paylaşılmayan kuyrukların kullanılabilir kalmasını sağlar. Buna ek olarak, bir bağlaşım olanağı yapısına bağlanırlığını kaybeden kuyruk yöneticileri, otomatik olarak başka bir kullanılabilir bağlaşım olanağında yapıyı yeniden oluşturmayı dener ve kuyruk paylaşım grubu içindeki paylaşılan kuyrukların kullanılabilirliğini artırır.

Bu işlev etkinleştirilirken dikkat edilecek noktalar

Sonlandırılmadan bağlaşım olanağı yapılarına bağlantı kaybını tolere eden bir kuyruk yöneticisi, kullanılabilir alternatif bir bağlaşım olanağı yoksa, bir süre için bir bağlaşım olanağı yapısına

yeniden bağlanamayabilir. Bağlantı kaybına uğrayan bir yapıda tanımlanan paylaşılan kuyruklar, yapıyla bağlantı yeniden kuruluncaya kadar kullanılamaz. Bu durumda, paylaşılan kuyruk işini gerçekleştirmek için kuyruk paylaşım grubu üyelerine bağlanan uygulamalar, erişmeleri gereken paylaşılan kuyrukların kullanılamadığını görebilir. Bu durumu önlemek için, bir bağlaşım olanağı yapısına bağlantı kaybedildiğinde kuyruk yöneticilerinin sonlanacak şekilde yapılandırılması önerilir. Bu sonlandırma, uygulamaları, uygulamanın gerektirdiği paylaşılan kuyrukların tanımlandığı bağlaşım olanağı yapılarına bağlanırlığı olan kuyruk paylaşım grubunun başka bir üyesine bağlanmaya zorlar.

Bağlaşım olanağı bağlanırlığı hatalarına Dayanıklılığın Yönetilmesi

Bu işlevi nasıl etkinleştirebilirim?

Bağlaşım olanağı bağlanırlığına dayanıklılık sağlamak için aşağıdaki adımlar gerçekleştirilmelidir

1. CFRM çift veri kümesinin, sistem tarafından yönetilen yeniden oluşturmayı destekleyecek şekilde biçimlendirildiğinden emin olun. Bu, kuyruk yöneticilerinin kullanılabilir bir bağlaşım olanağında bir yapıyı yeniden oluşturmak için sistem tarafından yönetilen yeniden oluşturma başlatmalarını sağlar. CFRM çift veri kümesinin biçimini belirlemek için **DISPLAY XCF,COUPLE,TYPE=CFRM** komutunu kullanın. Sistem tarafından yönetilen yeniden oluşturmayı desteklemek için CFRM çift veri kümesi aşağıdaki belirtilerek biçimlendirilmelidir:

   ```
   "ITEM NAME(SMREBLD) NUMBER(1)"
   ```

   CFRM çift veri kümesini biçimlendirmeye ilişkin daha fazla bilgi için z/OS MVS Setting Up a Sysplex (Sysplex Ayarı) belgelerine bakın.
2. Alternatif bir bağlaşım olanağı bulunduğundan ve tüm IBM MQ bağlaşım olanağı yapıları için CFRM tercih listesinde bulunduğundan emin olun. Bu, kuyruk yöneticilerinin yapılara erişimi mümkün olan en kısa sürede geri yüklemek için yapıları alternatif bir bağlaşım olanağı olarak yeniden oluşturmaya çalışmalarını sağlar.

   IBM MQ yapıları, CFRM ilkesinde ENFORCEORDER (NO) ile tanımlanmalıdır; böylece, IBM MQ yapıyı yeniden ayırması gerekiyorsa, XCF yapılandırmada en uygun CF ' yi seçebilir.

   Yapı tercihi listelerine ilişkin daha fazla bilgi için z/OS MVS Setting Up a Sysplex (Sistem Birleşimi Ayarlama) belgelerine bakın.
3. CFLEVEL (5) ile bağlantı kaybını tolere etmesi gereken tüm uygulama bağlaşım olanağı yapılarını değiştirin. Bu, bağlantı kaybını tolere edebilen en düşük düzeydir.
4. **QMGR CFCONLOS** ve **CFSTRUCT CFCONLOS** öznitelikleri için gereken değerleri belirleyin ve bunları uygun şekilde değiştirin. **QMGR CFCONLOS** özniteliği, yönetim yapısına bağlanırlık kaybının tolere edilip edilmeyeceğini denetler ve **CFSTRUCT CFCONLOS** özniteliği, her uygulama bağlaşım olanağı yapısının bağlanırlık kaybını tolere edip etmeyeceğini denetler. Bu özniteliklere ilişkin varsayılan değerler korunursa, kuyruk yöneticisi herhangi bir bağlaşım olanağı yapısına bağlanırlık kaybından sonra sona erer.
5. Her uygulama bağlaşım olanağı yapısı için **CFSTRUCT RECAUTO** özniteliği için gereken değerleri belirleyin ve bunları uygun şekilde değiştirin. Bu öznitelik, toplam bağlanırlık kaybından sonra günlüğe kaydedilen veriler kullanılarak bağlaşım olanağı yapılarının otomatik olarak kurtarılıp kurtarılmayacağını denetler. Bu özniteliğin varsayılan değeri korunursa, toplam bağlanırlık kaybından sonra uygulama yapıları için otomatik kurtarma gerçekleştirilmez.

Senaryo 1-Yönetim yapısına bağlanırlık kaybı

Kuyruk yöneticileri, kuyruk paylaşım grubundaki tüm kuyruk yöneticileri IBM WebSphere MQ 7.1 ya da daha sonraki bir düzeydeyse, sonlandırılmadan yönetim yapısına bağlanırlık kaybını tolere edebilir. Kuyruk paylaşım grubunda IBM WebSphere MQ 7.1 düzeyinden daha düşük bir düzeyde kuyruk yöneticileri varsa, kuyruk paylaşım grubundaki tüm kuyruk yöneticileri, yönetim yapısına bağlanırlık kaybolduğunda 00C510AB neden koduyla olağandışı biter.

Yönetim yapısına bağlanırlık, yönetim yapısına bağlanırlık kaybını tolere edecek şekilde yapılandırılmış herhangi bir kuyruk yöneticisi tarafından kaybedildiğinde, kuyruk paylaşım grubunun tüm üyeleri yönetim yapısıyla bağlantıyı keser. Kuyruk paylaşım grubundaki tüm etkin kuyruk yöneticileri daha

sonra yönetim yapısına yeniden bağlanmayı dener ve bu, sistem şebekesi içindeki tüm sistemlere en iyi bağlanırlıkla bağlaşım olanağı içinde yeniden tahsis edilmesine ve yönetim yapısı verilerini yeniden oluşturmasına neden olur.

Not: Bu, etkin kuyruk yöneticilerine sahip tüm sistemlere en iyi bağlanırlığı sağlayan bağlaşım olanağı olmayabilir.

Bir kuyruk yöneticisi yönetim yapısına yeniden bağlanamıyorsa (örneğin, yönetim yapısına ilişkin CFRM tercih listesindeki bağlaşım olanaklarının hiçbiri kullanılamıyorsa), kuyruk yöneticisi yönetim yapısına başarıyla yeniden bağlanıncaya ve yönetim yapısı verilerini yeniden oluşturuncaya kadar bazı paylaşılan kuyruk işlemleri kullanılamaz. Sistemde uygun bir bağlaşım olanağı kullanılabilir olduğunda yeniden bağlantı otomatik olarak gerçekleşir.

Bağlaşım olanağıyla bağlantı olmaması ya da yapıyı ayırmak için uygun bir bağlaşım olanağı olmaması nedeniyle kuyruk yöneticisi başlatılırken denetim yapısına bağlanılamaması tolere edilmez. Kuyruk paylaşım grubundaki tüm etkin kuyruk yöneticileri daha sonra yönetim yapısına yeniden bağlanmayı dener ve varsa başka bir bağlaşım olanağına yeniden yerleştirilmesine neden olur ve yönetim yapısı verilerini yeniden oluşturur.

2. senaryo-Uygulama yapısına bağlanırlık kaybı

CFLEVEL(5) ya da daha yüksek düzeydeki uygulama yapılarına bağlantı kaybı, kuyruk yöneticisi sonlandırılmadan tolere edilebilir. **CFLEVEL(4)** ya da daha düşük düzeydeki uygulama yapılarına bağlı kuyruk yöneticileri ya da **CFLEVEL(5)** adresindeki, bağlanırlık kaybını tolere edecek şekilde yapılandırılmamış yapılar, yapıya bağlanırlık kaybedildiğinde 00C510AB neden koduyla olağandışı sonlanır.

Bağlantı kaybını tolere edecek şekilde yapılandırılmış bir uygulama yapısına bağlanırlık kesildiğinde, yapıya bağlanırlığı kaybeden tüm kuyruk yöneticileri bağlantıyı keser. Kuyruk yöneticisinin sonraki davranışı, bağlanırlık kaybının kısmi mi, yoksa toplam mı olduğuna bağlıdır.

Uygulama yapısına kısmi bağlantı kaybı

Bağlanırlık kaybının kısmi olduğu belirlenirse, yapıyla bağlantısını kaybeden kuyruk yöneticileri, yapıyı daha iyi bağlanırlıkla başka bir bağlaşım olanağına taşımak için sistem tarafından yönetilen bir yeniden oluşturma başlatmayı dener. Bu yeniden oluşturma başarılı olursa, yapıdaki hem kalıcı hem de kalıcı olmayan iletiler diğer bağlaşım olanağına kopyalanır ve yapıdaki kuyruklara erişim geri yüklenir. Bağlanırlığı kaybetmeyen kuyruk yöneticileri yapıya bağlı kalır, ancak yapıya erişen işlemler, sistem tarafından yönetilen yeniden oluşturma işlemi sırasında gecikir.

Bir uygulama yapısı, daha iyi bağlanırlığı olan başka bir bağlaşım olanağına yeniden oluşturulamazsa ya da bazı kuyruk yöneticilerinin başka bir bağlaşım olanağında yeniden oluşturulduktan sonra yapıyla bağlantısı yoksa, yapıda tanımlanan kuyruklar, bağlaşım olanağına bağlanırlık geri yükleninceye kadar yapıyla bağlantısı olmayan kuyruk yöneticisinde kullanılamaz. Kuyruk yöneticileri kullanılabilir olduğunda yapıya otomatik olarak yeniden bağlanır ve yapıda tanımlanan paylaşılan kuyruklara erişim geri yüklenir.

Uygulama yapısına toplam bağlanırlık kaybı

Sistem şebekesinde bulunan tüm MVS sistemleri, uygulama yapısının ayrıldığı bağlaşım olanağına bağlanırlığını kaybetmişse, z/OS yapıya yeniden bağlanma girişiminde bulunulduğunda bağlaşım olanağından yapıyı ayırır. Kuyruk yöneticisinin birkaç nedenden ötürü yapıya yeniden bağlanmayı denemesi mümkündür; örneğin, bir uygulamanın paylaşılan bir kuyruğu açma girişimi ya da sistemden gelen ve yeni bağlaşım olanağı kaynaklarının kullanılabilir hale gelebileceğine ilişkin bir bildirim. Bu nedenle, etkilenen yapıdaki kalıcı olmayan tüm iletilerin, bir uygulama yapısına toplam bağlanırlık kaybından sonra kaybedilmesi olasıdır.

Kurtarılabilir uygulama yapıları, **RECAUTO(YES)** ile tanımlanmışsa, toplam bağlanırlık kaybından sonra otomatik olarak kurtarılır. Kurtarma, yapıyı tahsis etmek için alternatif bir bağlaşım olanağı varsa ya da böyle bir bağlaşım olanağı kullanılabilir olduğunda hemen başlar. **RECAUTO(YES)** ile bir yapı tanımlanmamışsa, **RECOVER CFSTRUCT** komutu çalıştırılarak kurtarma başlatılabilir. Bu, yapıdaki tüm kalıcı iletileri kurtarır, ancak kalıcı olmayan tüm iletiler kaybolur. Bu işlem kuyruk yöneticisi günlüğünün okunmasını içerdiğinden, yapıdaki paylaşılan kuyruklara erişim geri yükleninceye kadar yapı yedeklerinin düzenli olarak alınması önerilir.

Kuyruk yöneticileri, bir uygulama yapıda tanımlanan bir paylaşılan kuyruğu açmaya çalıştığında ya da sistemden yeni bağlaşım olanağı kaynaklarının kullanılabilir hale geldiğine ilişkin bir bildirim alındığında, kurtarılamayan uygulama yapılarına yeniden bağlanmayı dener. Yapıyı ayırmak için uygun bir bağlaşım olanağı varsa, yeni bir yapı ayrılır ve yapıda tanımlanan paylaşılan kuyruklara erişim geri yüklenir. Kalıcı iletiler kurtarılamayan yapılarda tanımlanan kuyruklara konamadığından, paylaşılan kuyruklardaki tüm iletiler kaybolur.

Operasyonel davranış

Belirli bir bağlaşım olanağı yapısına bağlanabilirlik kaybını tolere edecek şekilde yapılandırılmış bir IBM WebSphere MQ 7.1ya da daha sonraki bir kuyruk yöneticisi bağlanırlığı kaybederse, kuyruk paylaşım grubunun üyeleri hatadan otomatik olarak kurtulmayı ve yapıya yeniden bağlanmayı dener. Bu etkinlik, varsa daha iyi bağlanırlığa sahip başka bir bağlaşım tesisinde yapının yeniden tahsis edilmesini içerebilir. Ancak, bağlanırlık kaybından kurtulmak için işletmen müdahalesi gerekebilir.

Genellikle gerekli işleç işlemi aşağıdaki gibidir:

1. Bağlantı kaybına neden olan hatanın nedenini ortadan kaldırın.
2. IBM MQ yapılarının ayrılabileceği bir bağlaşım olanağının sistem şebekesi içindeki tüm sistemlerde kullanılabilir olduğundan emin olun

Bağlantı olayının kaybından sonra başka bir bağlaşım olanağında otomatik olarak yeniden yerleştirilmiş olan yapılar, kuyruk paylaşım grubundaki tüm kuyruk yöneticilerine en uygun bağlanırlıkla bağlaşım olanağına taşınabilir. Gerekirse, sistem tarafından yönetilen yeniden oluşturma komutu **SETXCF START,REBUILD** z/OS MVS System Commands Referenceiçinde belgelendiği şekilde başlatılarak bu yapılabilir.

Bir uygulama yapısına kısmi bağlantı kaybı durumunda, yapıya bağlanırlığını kaybeden kuyruk yöneticileri, sistem tarafından yönetilen bir yeniden oluşturma başlatmayı dener. Bu işlem, yapıyı yalnızca, o bağlaşım olanağı yapıya bağlı olan tüm etkin kuyruk yöneticilerine bağlanırsa, başka bir bağlaşım olanağındaki yapıyı ayırır. Bu nedenle, bir kuyruk paylaşım grubundaki kuyruk yöneticilerinin çoğunun bir uygulama yapısına bağlanırlığını kaybetmesi, özgün yapıya bağlı kuyruk yöneticileri nedeniyle yapıyı başka bir bağlaşım olanağına yeniden oluşturamamaları olasıdır. Bu durumda, hala özgün yapıya bağlı olan kuyruk yöneticileri, yapının yeniden oluşturulmasına izin vermek için kapatılabilir ya da **RESET CFSTRUCT ACTION(FAIL)** komutu yapıyı başarısız yapmak için verilebilir. **RECOVER CFSTRUCT** komutu verilerek uygulanabilir yapılarda kurtarma başlatılabilir.

Not: Yapı arızalandığında ve kurtarıldığında, yapıdaki kalıcı olmayan tüm iletiler kaybolur.

z/OS IBM MQ for z/OSiçindeki güvenlik kavramları

IBM MQgüvenliğinin önemini anlamak için bu konuyu ve sisteminizde yeterli güvenlik ayarına sahip olmamanın etkilerini anlamak için bu konuyu kullanın.

IBM MQ kaynaklarını neden korumanız gerektiğini

IBM MQ , potansiyel olarak değerli olan bilgilerin aktarılabilmesini sağlar. Güvenlik uygulanması, IBM MQ kaynağının sahip olduğu ve yönettiği kaynakların yetkisiz erişimden korunmasını sağlar. Bu erişim, bilgilerin kaybolmasına ya da açıklanmasını önleyebilir.

Herhangi bir yetkisiz kullanıcı ya da işlem tarafından, aşağıdaki kaynakların hiçbirine erişilmediğinden ya da bu kaynaklardan hiçbirinin değiştirilmediğinden emin olmanız gerekir:

- IBM MQile bağlantı
- Kuyruklar, işlemler ve ad listeleri gibiIBM MQ nesneleri
- IBM MQ iletim bağlantıları, yani, IBM MQ kanalları
- IBM MQ sistem denetimi komutları
- IBM MQ ileti
- İletilerle ilişkili bağlam bilgileri

To provide the necessary security, IBM MQ uses the z/OS system authorization facility (SAF) to route authorization requests to an External Security Manager (ESM), for example Security Server (previously known as RACF). IBM MQ , kendi güvenlik doğrulamasının olmadığını kabul eder. Where distributed queuing or clients are being used, you might require additional security measures, for which IBM MQ provides channel authentication records, channel exits, the MCAUSER channel attribute, and TLS.

Bir nesneye erişime izin verme kararı ESM tarafından yapılır ve IBM MQ bu kararı izler. ESM bir karar veremiyorsa, IBM MQ nesneye erişimi önler.

IBM MQ kaynaklarını korumuyorsanız ne olur

Güvenlikle ilgili bir şey yapmazsanız, en olası etki *Tümü* kullanıcılarının *her* kaynaklarına erişip değiştirebilmesidir. Bu, yalnızca yerel kullanıcıları değil, dağıtılmış kuyruklama ya da istemcileri kullanan uzak sistemlerdeki oturum açma güvenlik denetimlerinin, normalde z/OSiçin vakanın normalde olduğundan daha az sıkılmış olabileceği durumlarda da içerir.

Güvenlik denetimini etkinleştirmek için aşağıdakileri yapmanız gerekir:

- ESM ' yi takın ve etkinleştirin (örneğin, Security Server).
- Güvenlik Sunucusu dışında bir ESM kullanıyorsanız, MQADMIN sınıfını tanımlayın.
- MQADMIN sınıfını etkinleştirin.

Büyük ve küçük harf karışık kaynak adlarının kullanılmasının işletmeniz için yararlı olacağını göz önünde bulundurmanız gerekir. ESM tanıtımlarınızda karma büyük-küçük harf kullanımı kaynak adları kullanıyorsanız, MXADMIN sınıfını tanımlamanız ve etkinleştirmeniz gerekir.

z/OS Veri Kümesi Şifrelemesi

Data Set Encryption (DSE) provides the capability to encrypt z/OS data sets, so that the data they contain can only be viewed or modified by user IDs granted the specific permission. Bu, dosya sisteminde geri kalan verilerin şifrelenmesini sağlar ve veri kümelerini yönetmek için geçerli bir iş gereksinimi ve izinleri olan kullanıcılara duyarlı bilgilerin yanlışlıkla açıklanmasını önler.

V 9.2.0 CD Prior to IBM MQ for z/OS 9.1.4, IBM MQ for z/OS does not support use of DSE with the active logs, page sets, and shared message data sets (SMDS) that provide the primary persistence mechanisms for IBM MQ messages.

Bunun yerine, Advanced Message Security , tüm IBM MQ ağını kapsayan IBM MQ ileti sistemi için uçtan uca bir şifreleme çözümü sunar. Bu çözüm, uçtaki verilerin şifrelenmesi, dinlenmede ve hatta çalıştırma zamanı IBM MQ işlemlerinin içinde olmak üzere verilerin şifrelenmesini sağlar.

Bir IBM MQ altsisteminde kullanılan diğer VSAM ve sıralı veri kümeleri, DSE kullanılarak şifrelenebilir. Örneğin:

- Önyükleme veri kümesi (BSDS)
- CSQINPx DDNAMEs kullanılarak başlatma sırasında okunan sistem yapılandırması (MQSC) komutlarını sıralı dosyalar
- IBM MQ archive logs, often used for long term archival of IBM MQ log data for audit purposes.

Veri kümesi anahtar etiketiyle tanımlanan bir veri sınıfını ayırarak DSE kullanarak şifreleyebilirsiniz. Daha fazla bilgi için Günlük arşivleme depolamanızı planlamabaşlıklı konuya bakın.

V 9.2.0 IBM MQ for z/OS 9.1.4' tan IBM MQ for z/OS , daha önceki yayınlarda sağlanan desteğe ek olarak etkin günlükler ve sayfa kümeleriyle DSE kullanımını destekler.

V 9.2.0 IBM MQ for z/OS , paylaşılan ileti veri kümeleri (SMDS) için DSE kullanımını desteklemez.

V 9.2.0 confidentiality for data at rest on IBM MQ for z/OS with data set encryption.bölümüne bakın. daha fazla bilgi için.

İlgili kavramlar
Güvenlik kavramları
Kanal doğrulama kayıtları
z/OSüzerinde IBM MQ nesneleriyle çalışma yetkisi
Şifreleme güvenlik iletişim kuralları: TLS
z/OSüzerinde MQSC komutları yayınlayabileceğiniz kaynaklar
İlgili görevler
z/OSüzerinde güvenliğin ayarlanması
Bağlantı düzeyi güvenlik ve uygulama düzeyi güvenliği karşılaştırılıyor
İlgili başvurular
IBM MQ for z/OSiçin iletiler

z/OS IBM MQ for z/OSiçindeki güvenlik denetimleri ve seçenekler

Tüm IBM MQ altsistemi için güvenliğin açık olup olmadığını ve kuyruk yöneticisinde ya da kuyruk paylaşım grubu düzeyinde güvenlik denetimlerini gerçekleştirmek isteyip istemediğinizi belirleyebilirsiniz. Ayrıca, API-kaynak güvenliği için denetlenen kullanıcı kimliklerinin sayısını da denetleyebilirsiniz.

Altsistem güvenliği

Altsistem güvenliği, tüm kuyruk yöneticisi için herhangi bir güvenlik denetimin yapılıp yapılmadığını belirten bir denetimdir. Güvenlik denetimi gerektirmiyorsa (örneğin, bir sınama sisteminde) ya da IBM MQ ile bağlantı kurabilen tüm kaynaklarda (istemciler ve kanallar da içinde olmak üzere) güvenlik düzeyinden memnunsanız, başka bir güvenlik denetiminin gerçekleştirilmemesi için, kuyruk yöneticisi ya da kuyruk paylaşım grubu için güvenlik denetimini kapatabilirsiniz.

Bu, güvenliği tamamen kapatabilecek ve diğer güvenlik denetimlerinin gerçekleştirilip gerçekleştirilmediğini belirlemek için yalnızca bu denetim öğesidir. Yani, kuyruk yöneticisi ya da kuyruk paylaşım grubunu denetleyerek, başka bir IBM MQ denetimi yapılmazsa, IBM MQ açık bırakılırsa,securitygüvenlik gereksinimlerinizi diğer IBM MQ kaynakları için denetler.

Komutlar gibi belirli kaynak kümeleri için de güvenliği açabilir ya da kapatabilirsiniz.

Kuyruk yöneticisi ya da kuyruk paylaşım grubu düzeyinde denetleme

Güvenlik uygulayabilirsiniz; kuyruk yöneticisi düzeyinde ya da kuyruk paylaşım grubu düzeyinde güvenlik uygulayabilirsiniz. Kuyruk paylaşımı grubu düzeyinde güvenliği uygularsanız, gruptaki tüm kuyruk yöneticileri aynı profilleri paylaşır. Bu, güvenlik yönetimini daha kolay tanımlamak ve sürdürmek için daha az profil olduğu anlamına gelir. Ayrıca, var olan güvenlik profillerini devraldığı için kuyruk paylaşım grubuna yeni bir kuyruk yöneticisi eklenmesini de kolaylaştırır.

Ayrıca, kuruluş sırasında, örneğin, geçiş sırasında ya da kuyruk paylaşım grubunda, gruptaki diğer kuyruk yöneticilerine farklı güvenlik düzeyleri gerektiren bir kuyruk yöneticiniz varsa, her ikisinin birleşimi de uygulamak mümkündür.

Kuyruk paylaşım grubu düzeyinde güvenlik

Kuyruk paylaşım grubu düzeyinde güvenlik denetimi, tüm kuyruk paylaşım grubu için gerçekleştirilir. Daha az sayıda güvenlik profili tanımlamanızı gerektirdiğinden, güvenlik yönetimini basitleştirmenizi sağlar. Belirli bir kaynağı kullanmak için kullanıcı kimliği yetkisi, kuyruk paylaşım grubu düzeyinde işlenir ve kullanıcı kimliğinin kaynağa erişmek için kullandığı kuyruk yöneticilerinden bağımsızdır.

Örneğin, bir sunucu uygulamasının kullanıcı kimliği SERVER altında çalıştığını ve SERVER.REQUEST, VE SYSPlex içindeki her bir z/OS görüntüsünde bir SERVER yönetim ortamı çalıştırmak istiyorsunuz. SERVER 'ın her kuyruk yöneticisinde SERVER.REQUEST ' u tek tek (kuyruk yöneticisi düzeyinde güvenlik) açmasına izin vermek yerine, yalnızca kuyruk paylaşım grubu düzeyinde erişime izin verebilirsiniz.

Yerel ya da paylaşılan tüm kaynak tiplerini korumak için kuyruk paylaşım grubu düzeyinde güvenlik profillerini kullanabilirsiniz.

Kuyruk yöneticisi düzeyinde güvenlik
Yerel ya da paylaşılan tüm kaynak tiplerini korumak için kuyruk yöneticisi düzeyi güvenlik profillerini kullanabilirsiniz.

Her iki düzeyin birleşimi
Hem kuyruk yöneticisi hem de kuyruk paylaşım grubu düzeyinde güvenlik birleşimi kullanabilirsiniz.

Belirli bir kuyruk yöneticisine ilişkin kuyruk paylaşım grubu düzeyi güvenlik ayarlarını, o grubun üyesi olan bir kuyruk yöneticisine geçersiz kılabilirsiniz. Diğer bir kuyruk yöneticisinde, gruptaki diğer kuyruk yöneticilerinde gerçekleştirilen farklı güvenlik denetimlerini gerçekleştirmenizi sağlar.

Daha fazla bilgi için bakınız: Profiles to control queue sharing group or queue manager level security.

Denetlenen kullanıcı kimlikleri sayısını denetleme

RESLEEL, IBM MQ kaynak güvenliği için denetlenen kullanıcı kimliklerinin sayısını denetleyen bir Security Server profilidir. Olağan durumda, kullanıcı bir IBM MQ kaynağına erişmeyi denediğinde, Security Server o kaynağa erişime izin verilip verilmediğini görmek için ilgili kullanıcı kimliğini ya da tanıtıcılarını denetler. RESLEFIL tanıtımı tanımlayarak, geçerli olduğunda, sıfır, bir ya da, iki kullanıcı kimliği olup olmadığını denetleyebilirsiniz.

Bu denetimler, bağlantı temelinde ve bağlantının ömrü boyunca yapılan bir bağlantıda yapılır.

Her kuyruk yöneticisi için tek bir RESLEVEL tanıtımı vardır. Denetim, bir kullanıcı kimliğinin bu tanıtıma erişmesi için erişim tarafından uygulanır.

Büyük/küçük harf ya da büyük harf IBM MQ RACF sınıfları

Artık, karışık vaka kaynağı adlarını kullanmanıza ve bunları korumak için IBM MQ RACF profillerini tanımlamanıza olanak sağlayan karma büyük/küçük harf kullanımı RACF profili desteğini kullanabilirsiniz.

Aşağıdakilerden birini seçebilirsiniz:

- Continue using uppercase only IBM MQ RACF Classes as in previous releases, or
- Yeni karma vaka IBM MQ RACF sınıflarını kullanın.

Without the use of mixed case RACF profiles, you can still use mixed case resource names in IBM MQ for z/OS ; however, these resource names can only be protected by generic RACF profiles in the uppercase IBM MQ classes. When using mixed case IBM MQ RACF profile support you can provide a more granular level of protection by defining IBM MQ RACF profiles in the mixed case IBM MQ classes.

z/OS IBM MQ for z/OS' ta koruyabilirsiniz.

Bir kuyruk yöneticisi başlatıldığında ya da bir işletmen komutu tarafından istendiğinde, IBM MQ for z/OS tarafından hangi kaynakların korunmasını istediğinizi belirler.

Her bir kuyruk yöneticisi için hangi güvenlik denetlerinin gerçekleştirildiğini denetleyebilirsiniz. Örneğin, bir üretim kuyruğu yöneticisininde bir dizi güvenlik denetimi uygulayabilir, ancak sınama kuyruğu yöneticiliklerinden birini gerçekleştirmediğinizi varsayalım.

Bağlantı güvenliği

Bağlantı güvenliği denetimi, bir uygulama programı bir kuyruk yöneticisine bağlanmayı denediğinde yürütülür. Bu işlem, bir MQCONN ya da MQCONNX isteği yayınlayarak ya da kanal başlatıcı ya da CICS ya da IMS bağdaştırıcısı bir bağlantı isteği yayınlayarak gerçekleştirilir.

Kuyruk yöneticisi düzeyinde güvenlik kullanıyorsanız, belirli bir kuyruk yöneticisi için bağlantı güvenliği denetimini kapatabilirsiniz. Ancak, bunu herhangi bir kullanıcı bu kuyruk yöneticisine bağlayabilir.

CICS bağdaştırıcısı için, bağlantı güvenliği denetimi için yalnızca CICS adres alanı kullanıcı kimliği kullanılır; tek tek CICS uçbirimi kullanıcı kimliği değildir. IMS bağdaştırıcısı için, IMS denetimi ya da bağımlı bölgeler IBM MQ' e bağlandığında, IMS adres alanı kullanıcı kimliği denetlenir. Kanal başlatıcı için, kanal başlatıcı adres alanı tarafından kullanılan kullanıcı kimliği denetlenir.

Bağlantı güvenliği denetimini, kuyruk yöneticisi ya da kuyruk paylaşım grubu düzeyinde açabilir ya da kapatabilirsiniz.

Komut güvenliği

Bir kullanıcı, Komut verme' ta açıklanan kaynaklardan herhangi birinden bir MQSC komutu verdiğinde, komut güvenliği denetimi gerçekleştirilir. You can make a separate check on the resource specified by the command as described in “Komut kaynağı güvenliği” sayfa 264.

Komut denetimini kapadığınızda, komutlar ile ilgili üreticiler, komutu verme yetkisine sahip olup olmadıklarını kontrol etmiyorlardı.

Bir konsoldan MQSC komutları girilirse, konsolun z/OS SYS konsol yetkisi özniteliğine sahip olması gerekir. CSQINP1 ya da CSQINP2 veri kümelerinden ya da kuyruk yöneticisi tarafından dahili olarak verilen komutlar, CSQINPX için olanlar kanal başlatıcı adres alanının kullanıcı kimliğini kullanırken, tüm güvenlik denetimlerinden muaftır. Bu veri kümelerini normal veri kümesi koruması aracılığıyla kimin güncellemesine izin verileceğini denetlemeniz gerekir.

Komut güvenliği denetimini, kuyruk yöneticisi ya da kuyruk paylaşım grubu düzeyinde açabilir ya da kapatabilirsiniz.

Komut kaynağı güvenliği

Bazı MQSC komutları; örneğin, yerel bir kuyruk tanımlamak için, IBM MQ kaynaklarının işlenmesini içerir. Komut kaynağı güvenliği etkin olduğunda, bir kaynağı içeren her komut yayınlandığında IBM MQ , kullanıcının o kaynağın tanımını değiştirmesine izin verilip verilmediğini denetler.

Adlandırma standartlarının uygulanmasına yardımcı olmak için komut kaynağı güvenliğini kullanabilirsiniz. Örneğin, bir bordro denetimcisinin yalnızca adları "BORDRO" olarak başlayan kuyrukları silmesine ve tanımlamaya izin verilebileceği gibi. Komut kaynağı güvenliği etkin değilse, komut tarafından işlenmekte olan kaynaklarda hiçbir güvenlik denetimi yapılmamaktadır. Komut kaynağı güvenliğini komut güvenliği ile karıştırmayın; iki bağımsız değer bağımsız olur.

Komut kaynağı güvenliği denetiminin kapatılması, özellikle komut içermeyen diğer işlem tipleri için yapılan kaynak denetimini etkilemez.

Komut kaynağı güvenliği denetimini, kuyruk yöneticisi ya da kuyruk paylaşım grubu düzeyinde açabilir ya da kapatabilirsiniz.

Kanal güvenliğine ilişkin önemli noktalar

Kanal güvenliği

Kanalları kullanırken, kullanılabilecek güvenlik özellikleri, hangi iletişim protokolünün kullanabileceğini bağlıdır. TCP 'yi kullanıyorsanız, TLS' yi kullanabilmeniz için iletişim protokolüyle birlikte sağlanan hiçbir güvenlik özelliği yoktur. APPC kullanıyorsanız, doğrulama için hedef MCA 'ya ağ üzerinden gönderen MCA' dan kullanıcı kimliği bilgilerini akıtabilirsiniz.

Her iki iletişim kuralı için, güvenlik amacıyla hangi kullanıcı kimliklerinin denetleyeceğini ve kaç kişinin denetleneceğini belirleyebilirsiniz. Yine, kullanabileceğiniz seçenekler, kanalı tanımlarken belirttiğiniz protokole ve kanal başlatıcısına ilişkin RESLEVEL ayarına bağlıdır.

Kanal güvenliği tipleriyle ilgili ek bilgi için Kanal kimlik doğrulama kayıtları ve Güvenlik çıkışa genel bakışbaşlıklı konuya bakın.

İlgili başvurular

"IBM MQ for z/OSiçinde API-kaynak güvenliği" sayfa 265
Bir uygulama MQOPEN ya da MQPUT1 çağrısına sahip bir nesneyi açtığında kaynaklar denetlenir. Bir nesneyi açmak için gereken erişim, kuyruk açıldığında hangi açık seçeneklerin belirtilmesine bağlıdır.

z/OS *IBM MQ for z/OSiçinde API-kaynak güvenliği*

Bir uygulama MQOPEN ya da MQPUT1 çağrısına sahip bir nesneyi açtığında kaynaklar denetlenir. Bir nesneyi açmak için gereken erişim, kuyruk açıldığında hangi açık seçeneklerin belirtilmesine bağlıdır.

API-kaynak güvenliği, aşağıdaki denetlere bölünmektedir:

- Kuyruk
- Süreç
- Ad Listesi
- Diğer kullanıcı
- Bağlam

Kuyruk yöneticisi nesnesi açılırken ya da depolama sınıfı nesnelerine erişilirken güvenlik denetimi gerçekleştirilmez.

Kuyruk

Hangi kuyruğun açılmasına izin verilen ve hangi seçeneklerin açılmasına izin verilen, kuyruk güvenlik denetimi denetimlerinin bulunduğu denetlenir. Örneğin, bir kullanıcının PAYROLL.INCREASE.SALARY , kuyruktaki iletilere göz atmak için (MQOO_BROWSE seçeneğini kullanarak), ancak iletileri kuyruktan kaldırmak için değil (MQOO_INPUT_ * seçeneklerinden birini kullanarak). Kuyruk denetimini kapadığınızda, herhangi bir kullanıcı herhangi bir geçerli açma seçeneğiyle herhangi bir kuyruk açabilir (bu, MQOPEN ya da MQPUT1 çağrısında geçerli bir MQOO_ * seçeneği).

Kuyruk yöneticisi ya da kuyruk paylaşım grubu düzeyinde kuyruk güvenliği denetimini açabilir ya da kapatabilirsiniz.

Süreç

Bir kullanıcı bir süreç tanımlaması nesnesini açtığında, süreç güvenliği denetimi gerçekleştirilir. İşlemleri geri alma işlemini iptal etmek için herhangi bir kullanıcı herhangi bir işlem açabilir.

Process Security denetimini, kuyruk yöneticisi ya da kuyruk paylaşım grubu düzeyinde açabilir ya da kapatabilirsiniz.

Ad Listesi

Kullanıcı bir ad listesi açtığında, ad listesi güvenlik denetimi gerçekleştirilir. Ad listelerini denetleyebilirseniz, herhangi bir kullanıcı herhangi bir ad listesi açabilir.

Ad listesi güvenlik denetimini, kuyruk yöneticisi ya da kuyruk paylaşım grubu düzeyinde açabilir ya da kapatabilirsiniz.

Diğer kullanıcı

Diğer kullanıcı güvenliği, bir kullanıcı kimliğinin bir IBM MQ nesnesini açmak için başka bir kullanıcı kimliği yetkisini kullanıp kullanamayacağını denetler.

Örneğin:

- A server program running under user ID PAYSERV retrieves a request message from a queue that was put on the queue by user ID USER1.

- Sunucu programı istek iletisini aldığında, isteği işler ve yanıtı, istek iletisiyle belirtilen yanıtlama kuyruğuna yerleştirir.
- Yanıt kuyruğun açılmasına yetki vermek için kendi kullanıcı kimliğini (PAYSERV) kullanmak yerine, sunucu başka bir kullanıcı kimliği belirtebilir (bu durumda USER1). Bu örnekte, diğer kullanıcı güvenliği, yanıtlama kuyruğu açılırken PAYSERV kullanıcı kimliğinin USER1 kullanıcı kimliğini alternatif bir kullanıcı kimliği olarak belirlemesine izin verilip verilmediğini denetler.

Alternatif kullanıcı kimliği, nesne tanımlayıcısının (MQOD) *AlternateUserId* alanında belirtilir.

Herhangi bir IBM MQ nesnesindeki diğer kullanıcı kimliklerini (örneğin, işlemler ya da ad listeleri) kullanabilirsiniz. Diğer herhangi bir kaynak yöneticisi tarafından kullanılan kullanıcı kimliğini etkilemez; örneğin, CICS güvenliği ya da z/OS veri kümesi güvenliği için.

Diğer kullanıcı güvenliği etkin değilse, herhangi bir kullanıcı başka bir kullanıcı kimliğini başka bir kullanıcı kimliği olarak kullanabilir.

Diğer kullanıcı güvenliği denetimini, kuyruk yöneticisi ya da kuyruk paylaşım grubu düzeyinde açabilir ya da kapatabilirsiniz.

Bağlam

Bağlam, belirli bir ileti için geçerli olan ve iletinin parçası olan ileti tanımlayıcısında (MQMD) yer alan bilgilerdir. Bağlam bilgileri iki bölüm halinde gelir:

Kimlik bölümü
İletiyi ilk olarak bir kuyruğa koyan uygulamanın kullanıcısı. Bu, aşağıdaki alanlardan oluşur:

- *UserIdentifier*
- *AccountingToken*
- *ApplIdentityData*

Köken bölümü
İletiyi, şu anda saklandığı kuyruğa koyan uygulama. Bu, aşağıdaki alanlardan oluşur:

- *PutApplType*
- *PutApplName*
- *PutDate*
- *PutTime*
- *ApplOriginData*

Uygulamalar, bir MQPUT ya da MQPUT1 çağrısı yapıldığında bağlam verilerini belirtebilir. Uygulama verileri üretebilir, veriler başka bir iletiden aktarılabilir ya da kuyruk yöneticisi verileri varsayılan olarak oluşturabilir. Örneğin, sunucu programları, istekte bulunanın kimliğini denetlemek için bağlam verilerini kullanabilir; yani, bu ileti doğru uygulamadan mı geldi? Genellikle, diğer bir kullanıcının kullanıcı kimliğini belirlemek için *UserIdentifier* alanı kullanılır.

Kullanıcının herhangi bir MQOPEN ya da MQPUT çağrısındaki bağlam seçeneklerinden herhangi birini belirtip belirtemeyeceğini denetlemek için bağlam güvenliğini kullanıyorsunuz. Bağlam seçenekleriyle ilgili bilgi edinmek için İleti bağlamına ilişkin MQOPEN seçenekleribaşlıklı konuya bakın. Bağlamla ilgili ileti tanımlayıcı alanlarına ilişkin açıklamalar için bakınız: MQMD-Message descriptorMQMD -İleti tanımlayıcısı.

Bağlam güvenliğini geri verme işlemi kapatılırsa, herhangi bir kullanıcı kuyruk güvenliğinin izin verdiği bağlam seçeneklerinden herhangi birini kullanabilir.

Bağlam güvenliği denetimini, kuyrukta, kuyruk yöneticisinde ya da kuyruk paylaşım grubu düzeyinde açabilir ya da kapatabilirsiniz.

z/OS z/OSüzerinde kullanılabilirlik

IBM MQ for z/OS , yüksek düzeyde kullanılabilirlik için birçok özelliğe sahiptir. Bu konuda, kullanılabilirlik açısından dikkate alınması gereken bazı noktalar açıklanmaktadır.

Kuyruk yöneticisi ya da kanal başlatıcısı başarısız olursa, IBM MQ ' un bazı özellikleri sistem kullanılabilirliğini artırabilir. Bu özelliklerle ilgili daha fazla bilgi için aşağıdaki bölümlere bakın:

- Sysplex ile ilgili önemli noktalar
- Paylaşılan kuyruklar
- Paylaşılan kanallar
- IBM MQ ağ kullanılabilirliği
- z/OS Automatic Restart Manager (ARM) olanağının kullanılması
- z/OS Extended Recovery Facility (XRF) olanağının kullanılması
- Bir kuyruk paylaşım grubunda kurtarma için z/OS GROUPUR özniteliğinin kullanılması
- Kullanılabilirlik hakkında daha fazla bilgi nereden bulunabileceği

Sysplex ile ilgili önemli

Bir *sysplex*içinde, z/OS işletim sistemi görüntüleri tek bir sistem görüntüsünde işbirliği içinde çalışır ve bir bağlaşım olanağını kullanarak iletişim kurar. IBM MQ , gelişmiş kullanılabilirlik için sistem şebekesi ortamının olanaklarını kullanabilir.

Bir kuyruk yöneticisi ile belirli bir z/OS görüntüsü arasındaki kötülükler kaldırıldığında, bir kuyruk yöneticisinin bir görüntü hatası durumunda farklı bir z/OS görüntüsünde yeniden başlatılması gerekir. Yeniden başlatma mekanizması el ile olabilir, ARM ' yi kullanabilir ya da sistem otomasyonunu kullanarak aşağıdakileri doğruladığınızda:

- Tüm sayfa kümeleri, günlükler, önyükleme veri kümeleri, kod kitaplıkları ve kuyruk yöneticisi yapılandırma veri kümeleri paylaşılan birimlerde tanımlanır.
- Altsistem tanımlamasının sistem birleşimi (sysplex) kapsamı ve sistem birleşimi (sysplex) içinde benzersiz bir ad vardır.
- IPL zamanındaki her z/OS görüntüsünde kurulu olan *erken kod* düzeyi aynı düzeyde.
- TCP sanal IP adresleri (VIPA), sistem birleşimi (sysplex) içindeki her TCP yığınında bulunur ve IBM MQ TCP dinleyicileri ve gelen bağlantıları varsayılan anasistem adları yerine VIPA ' ları kullanacak şekilde yapılandırdınız.

TCP ' nin bir sistem şebekesinde kullanılmasına ilişkin ek bilgi için bkz. *sysplex içinde TCP/IP*, SG24-5235, bir IBM Redbooks yayını.

Ayrıca, bir sistem birleşiminde farklı işletim sistemi görüntülerinde çalışan birden çok kuyruk yöneticisi yapılandırabilirsiniz; böylece, bir kuyruk paylaşım grubu olarak çalışmak için, daha yüksek kullanılabilirlik ve iş yükü dengelemesi için paylaşılan kuyruklardan ve paylaşılan kanallardan yararlanabilirsiniz.

Paylaşılan kuyruklar

Kuyruk paylaşım grubu ortamında, bir uygulama kuyruk paylaşım grubu içindeki kuyruk yöneticilerine bağlanabilir. Kuyruk paylaşım grubundaki tüm kuyruk yöneticileri aynı paylaşılan kuyruklara erişebildiği için, uygulama belirli bir kuyruk yöneticisinin kullanılabilirliğine bağlı değildir; kuyruk paylaşım grubundaki herhangi bir kuyruk yöneticisi herhangi bir kuyrukta hizmet verebilir. Kuyruk yöneticisi, kuyruk paylaşım grubundaki diğer tüm kuyruk yöneticileri kuyruğun işlenmesine devam edebildiğinden, kuyruk yöneticisi durdurursa, daha fazla kullanılabilirlik sağlar. Paylaşılan kuyrukların yüksek düzeyde kullanılabilirliğine ilişkin bilgi için bkz. “Paylaşılan kuyruklar kullanmanın avantajları” sayfa 180.

Bir kuyruk paylaşım grubundaki iletilerin kullanılabilirliğini daha da artırmak için, IBM MQ , gruptaki başka bir kuyruk yöneticisinin bağlaşım olanağını olağandışı bir şekilde bağlayıp kesmediğini saptar ve bu kuyruk yöneticisi için hala beklemekte olan iş birimlerini tamamlar (olanaklı olduğu durumlarda). Bu, *eş kurtarma* olarak bilinir ve “Eşdüzey kurtarma” sayfa 254içinde açıklanmıştır.

Eş kurtarma işlemi, hata sırasında belirsiz olan iş birimlerini kurtaramaz. Hatada yer alan tüm sistemleri (örneğin, CICS, Db2ve IBM MQ gibi) yeniden başlatmak ve bunların tümü aynı yeni işlemcide yeniden

başlatıldığından emin olmak için Otomatik Yeniden Başlatma Yöneticisi 'ni (ARM) kullanabilirsiniz. Bu, yeniden eşzamanlayabilecekleri ve belirsiz iş birimleri için hızlı bir şekilde kurtarılabilecekleri anlamına gelir. Bu, “z/OS Automatic Restart Manager (ARM) olanağını kullanma” sayfa 268içinde açıklanmaktadır.

Paylaşılan kanallar

Kuyruk paylaşım grubu ortamında, IBM MQ , ağa yüksek düzeyde kullanılabilirlik sağlayan işlevler sağlar. Kanal başlatıcı, bir hak kazanan sunucular kümesi boyunca ağ isteklerini dengeleyen ve ağ üzerindeki sunucu hatalarını gizleyen ağ ürünlerini kullanmanıza olanak sağlar (örneğin, VTAM soysal kaynakları). IBM MQ , gelen istekler için soysal bir kapı kullanır; böylece, bağlantı istekleri kuyruk paylaşım grubundaki kullanılabilir herhangi bir kanal başlatıcısına yönlendirilebilir. Bu, “Paylaşılan kanallar” sayfa 200içinde açıklanmaktadır.

Paylaşılan giden kanallar, bir paylaşılan iletim kuyruğundan gönderdikleri iletileri alır. Paylaşılan bir kanalın durumuna ilişkin bilgiler, tüm kuyruk paylaşım grubu düzeyi için tek bir yerde tutulur. Bu, kanal başlatıcı, kuyruk yöneticisi ya da iletişim altsistemi başarısız olursa, kuyruk paylaşım grubundaki farklı bir kanal başlatıcısında otomatik olarak bir kanal yeniden başlatılabildiği anlamına gelir. Buna *eş kanal kurtarma* adı verilir ve Paylaşılan giden kanallariçinde açıklanmıştır.

IBM MQ ağ kullanılabilirliği

IBM MQ messages are carried from queue manager to queue manager in an IBM MQ network using channels. Bir kuyruk yöneticisinin ağ kullanılabilirliğini artırmak ve bir IBM MQ kanalının bir ağ sorununu saptaması ve yeniden bağlanmasını sağlamak için yapılandırmayı bir düzey düzeylerde değiştirebilirsiniz.

TCP *Keepalive* , TCP/IP kanalları için kullanılabilir. Bu, TCP ' nin ağ hatalarını algılamak için oturumlar arasında düzenli aralıklarla paket göndermesine neden olur. KAINT kanalı özniteliği, bir kanal için bu paketlerin sıklığını belirler.

AdoptMCA , bir kanalın, bir ağ kesintisinin sonlandırılması ve yeni bir bağlantı isteğiyle değiştirilmesinin sonucu olarak alma işlemi sırasında engellenmiş bir kanala izin verir. AdoptMCA ' yı, MQSC yardımcı programı ile ADOPTMCA kuyruk yöneticisi özelliğini kullanarak ya da Programlanır Komut Biçimleri arabirimiyle AdoptNewMCAType özelliğini kullanarak kontrol edin.

ReceiveTimeout , bir kanalın ağ alma çağrısında kalıcı olarak engellenmesini önler. RCVTIME ve RCVTMIN kanal başlatıcı parametrelerinde, kanallara ilişkin alma zamanaşımı özelliklerini, sağlıklı işletim bildirimi aralığının bir işlevi olarak belirleyin. Daha fazla ayrıntı için bkz. Kuyruk yöneticisi parametresi .

z/OS Automatic Restart Manager (ARM) olanağını kullanma

You can use IBM MQ for z/OS in conjunction with the z/OS automatic restart manager (ARM). Bir kuyruk yöneticisi ya da kanal başlatıcısı başarısız olursa, ARM aynı z/OS görüntüsünde yeniden başlatır. z/OS başarısız olursa, ilgili altsistemlerden ve uygulamalardan oluşan bir grup da başarısız olur. ARM, sistem birleşimi içindeki başka bir z/OS görüntüsünde, başarısız olan tüm sistemleri otomatik olarak, önceden tanımlanmış bir sırada yeniden başlatabilir. Buna sistem arası yeniden başlatma adı verilir.

ARM, paylaşılan kuyruk ortamında belirsiz hareketlerin hızla kurtarılmasını sağlar. Kuyruk paylaşım grupları kullanmayasanız da, daha yüksek kullanılabilirlik sağlar.

z/OS hatası durumunda, sistem birleşimi içindeki farklı bir z/OS görüntüsünde kuyruk yöneticisini yeniden başlatmak için ARM ' yi kullanabilirsiniz.

Otomatik yeniden başlatmayı etkinleştirmek için aşağıdaki işlemi yapmanız gerekir:

1. Bir ARM bağlaşım veri kümesi ayarlayın.
2. z/OS ' in bir *ARM ilkesinde*gerçekleştirmesini istediğiniz otomatik yeniden başlatma işlemlerini tanımlayın.
3. ARM ilkesini başlat.

Kuyruk yöneticilerini farklı z/OS görüntülerinde otomatik olarak yeniden başlatmak istiyorsanız, kuyruk yöneticisinin yeniden başlatılacağı her bir z/OS görüntüündeki her kuyruk yöneticisi, bir sistem birleşimi (sysplex) benzersiz 4 karakterlik altsistem adıyla tanımlanmalıdır.

Using ARM with IBM MQ is described in IBM MQ ağında ARM ' nin kullanılması.

z/OS Extended Recovery Facility (XRF) olanağını kullanma

IBM MQ ' u genişletilmiş bir kurtarma olanağı (XRF) ortamında kullanabilirsiniz. IBM MQ'in sahip olduğu tüm veri kümeleri (yürütülebilir kod, BSDSs, günlükler ve sayfa kümeleri) etkin ve alternatif XRF işlemcileri arasında paylaşılan DASD' de olmalıdır.

Kurtarma işlemi için XRF kullanırsanız, etkin işlemcinin kuyruk yöneticisini durdurmanız ve diğer işlemciye başlatmanız gerekir. CICSiçin, CICStarafından sağlanan komut listesi çizelgesini (CLT) kullanarak bunu yapabilirsiniz ya da sistem işletmeni bunu el ile yapabilir. For IMS, this is a manual operation and you must do it after the coordinating IMS system has completed the processor switch.

Kuyruk yöneticisinin alternatif işlemciye geçebilmesi için önceIBM MQ yardımcı programları tamamlanmalıdır ya da sonlandırılmalıdır. XRF kurtarma planlarınızı planlarken bu olası kesintinin etkisini dikkatli bir şekilde düşünün.

Etkin işlemcinin kuyruk yöneticisi sona ermeden önce, kuyruk yöneticisinin alternatif işlemciyi başlatmasını önlemek için dikkatli olun. Erken bir başlangıç, verilerde, katalogda ve günlükte ciddi bütünlük sorunlarına neden olabilir. Genel kaynak diziselleştirmesini (GRS) kullanarak, iki sistemde aynı anda IBM MQ ' nin kullanımını önleyerek bütünlük sorunlarının önlenmesine yardımcı olur. BSDS ' yi korumalı bir kaynak olarak eklemeli ve etkin ve alternatif XRF işlemcilerini GRS halkasına dahil etmeniz gerekir.

Bir kuyruk paylaşım grubunda kurtarma için z/OS GROUPUR özniteliğinin kullanılması

Kuyruk paylaşım grupları (QSG), bu konuda açıklanan ek işlem olanaklarına izin verir. GROUPUR özniteliği, XA istemci uygulamalarının, QSG ' nin herhangi bir üyesi üzerinde gerçekleştirilmesi gerekebilecek herhangi bir belirsiz hareket kurtarma işlemi gerçekleştirmesine olanak sağlar.

Bir XA istemcisi uygulaması bir Sysplex aracılığıyla bir kuyruk paylaşım grubuna (QSG) bağlanırsa, hangi kuyruk yöneticisinin bağlandığı belirli bir kuyruk yöneticisini garanti edemez. Kuyruk paylaşım grubundaki kuyruk yöneticilerine göre GROUPUR özniteliğinin kullanılması, QSG ' nin herhangi bir üyesinde gerçekleştirilmesi gerekebilecek herhangi bir belirsiz hareket kurtarma işlemini geçerli kılabilir. Uygulamanın ilk olarak bağlı olduğu kuyruk yöneticisi kullanılamıyorsa bile, işlem kurtarma işlemi gerçekleşebilir.

Bu özellik XA istemcisi uygulamasını, QSG ' nin belirli üyeleri üzerindeki herhangi bir bağımlılıktan kurtarır ve kuyruk yöneticisinin kullanılabilirliğini artırır. Kuyruk paylaşım grubu, tüm IBM MQ özelliklerini sağlayan tek bir varlık olarak ve tek bir kuyruk yöneticisi hatası olmaksızın, işlemsel uygulamaya görünür.

Bu işlevsellik, işlemsel uygulama için belirgin değildir.

Kullanılabilirlik hakkında daha fazla bilgi nerede bulunacağı

Bu konularla ilgili daha fazla bilgiyi aşağıdaki kaynaklardan bulabilirsiniz:

Çizelge 23. Kullanılabilirlik hakkında daha fazla bilgi nerede bulunacağı

Konu	**Nereye bakılacağı**
Kuyruk paylaşım grupları	“Paylaşılan kuyruklar ve kuyruk paylaşım grupları” sayfa 160

Çizelge 23. Kullanılabilirlik hakkında daha fazla bilgi nerede bulunacağı (devamı var)	
Konu	**Nereye bakılacağı**
Sistem değiştirbeleri	Sistem değiştirgelerinin yapılandırılması
Otomatik Yeniden Başlatma Yöneticisi 'nin Kullanılması Yardımcı programlar	IBM MQ ağındaki ARM ' nin kullanılması
MQSC komutları	MQSC komutları

z/OS IBM MQ for z/OSile ilgili izleme ve istatistikler

IBM MQ for z/OS , kuyruk yöneticisini izlemek ve istatistikleri toplamak için bir dizi tesis içerir.

IBM MQ , sistemi izlemek ve istatistik toplamak için gereken olanakları sağlar. Bu olanaklarla ilgili daha fazla bilgi için aşağıdaki bölümlere bakın:

- “Çevrimiçi izleme” sayfa 270
- “IBM MQ İz” sayfa 270
- “Olaylar” sayfa 271

Çevrimiçi izleme

IBM MQ , IBM MQ nesnelerinin durumunu izlemek için aşağıdaki komutları içerir:

- DISPLAY CHSTATUS, belirtilen bir kanala ilişkin durumu görüntüler.
- DISPLAY QSTATUS, belirtilen bir kuyruğun durumunu görüntüler.
- DISPLAY CONN, belirtilen bir bağlantının durumunu görüntüler.

Bu komutlarla ilgili daha fazla bilgi için bkz. MQSC komutları.

IBM MQ İz

IBM MQ , kuyruk yöneticisi çalışırken aşağıdaki bilgileri toplamak için kullanabileceğiniz bir izleme olanağı sağlar:

Performans İstatistikleri
İstatistik izleme programı, başarımı izlemenize ve sisteminizin ayarını izlemenize yardımcı olmak için aşağıdaki bilgileri toplar:

- Farklı MQI isteklerine ilişkin sayılar (ileti yöneticisi istatistikleri)
- Farklı nesne isteklerine ilişkin sayıları (veri yöneticisi istatistikleri)
- Db2 kullanımı hakkında bilgi (Db2 yöneticisi istatistikleri)
- Coupling Facility kullanımı hakkında bilgi (Coupling Facility manager istatistikleri)
- SMDS kullanımıyla ilgili bilgiler (paylaşılan ileti veri kümesi istatistikleri)
- Arabellek havuzu kullanımına ilişkin bilgiler (arabellek yöneticisi istatistikleri)
- Günlüğe kaydetme hakkında bilgi (günlük yöneticisi istatistikleri)
- Depolama kullanımına ilişkin bilgiler (depolama yöneticisi istatistikleri)
- Kilit istekleriyle ilgili bilgiler (kilit yöneticisi istatistikleri)

Hesap verileri

- Hesap izleme, MQI çağrılarını işlemek için harcanan işlemci süresi ve belirli bir kullanıcı tarafından yapılan MQPUT ve MQGET isteklerinin sayısı hakkında bilgi toplar.
- IBM MQ , IBM MQile her göreve ilişkin bilgileri de toplayabilir. Bu veriler, iş parçacığı düzeyinde bir muhasebe kaydı olarak toplanır. Her iş parçacığı için, IBM MQ bu iş parçacığı tarafından kullanılan her bir kuyruk hakkında da bilgi toplar.

İzleme tarafından oluşturulan veriler, System Management Facility (SMF) olanağına ya da genelleştirilmiş izleme olanağına (GTF) gönderilir.

Olaylar

IBM MQ olayları, bir kuyruk yöneticisinde hatalara, uyarılara ve diğer önemli oluşumlara ilişkin bilgi sağlar. By incorporating these events into your own system management application, you can monitor the activities across many queue managers, for multiple IBM MQ applications. Özellikle, sisteminizdeki tüm kuyruk yöneticilerini tek bir kuyruk yöneticisinden izleyebilirsiniz.

Olaylar, bir operatöre olayların sunumunu destekleyen bir yönetim uygulamasına kullanıcı tarafından yazılan bir raporlama mekanizması aracılığıyla raporlanabilir. Olaylar, raporları izlemek ve uygun uyarıları oluşturmak için diğer yönetim ağları için (örneğin, NetView) aracı olarak çalışan uygulamaları da etkinleştirmektedir.

İlgili görevler

IBM MQ izleme olanağını kullanma

IBM MQ olaylarını kullanma

z/OS z/OSüzerinde kurtarma yok etme birimi

Bazı işlemsel uygulamalar QMGR yerine, kuyruk yöneticisi adı yerine QSG adını belirterek bir kuyruk paylaşım grubundaki (QSG) bir kuyruk yöneticisine bağlandığında kurtarma düzeni birimi olarak bir GROUP (grup) kullanabilir. Bu, QSG ' de aynı kuyruk yöneticisine yeniden bağlanma gereksinimini kaldırarak hareket kurtarmanın daha esnek ve güçlü olmasını sağlar.

Kuyruk paylaşım grubu adı kullanılarak bağlanan uygulamalar tarafından başlatılan işlemler, bir kurtarma düzeni grubuna da sahip olur.

Bir işlemsel uygulama, bir grup kurtarma yok etme birimiyle bağlantı kurduğunda, bu uygulama, mantıksal olarak kuyruk paylaşım grubuna bağlanır ve belirli bir kuyruk yöneticisine benzeşimi olmaz. Any 2-phase commit transactions that it has started that have completed phase-1 of the commit process, that is, they are in doubt, can be inquired and resolved, when connected to any queue manager within the QSG. Kurtarma senaryolarında, hareket eşgüdümcünün aynı kuyruk yöneticisine yeniden bağlanmak zorunda olmadığı anlamına gelir. Bu durumda, bu süre içinde kullanılamaz.

Bir QMGR kurtarma düzeni birimiyle bağlantı kurulan uygulamaların, bağlı oldukları kuyruk yöneticisine doğrudan benzeşimi vardır. Bir kurtarma senaryosında, kuyruk yöneticisinin bir kuyruk paylaşım grubuna ait olup olmadığına bakılmaksızın, hareket koordinatörünün belirsiz hareketleri çözmek için aynı kuyruk yöneticisine yeniden bağlanması gerekir.

Uygulamalar bir kuyruk paylaşım grubu adı belirttiğinde ve bu nedenle, bir grup kurtarma girişlerini içeren bir QSG ' de kuyruk yöneticisine bağlandığında, kuyruk paylaşım grubu mantıksal olarak ayrı bir kaynak yöneticidir. Bunun anlamı, belirsiz hareketlerin yalnızca aynı kurtarma düzeni birimiyle yeniden bağlantı kurması durumunda bir uygulama tarafından görülebilmesinin anlamına gelir. Bir QMGR kurtarma yok etme birimine sahip belirsiz hareketler, bir grup kurtarma düzeni ile bağlantılı olan uygulamalar tarafından görülemez ve tam tersi de geçerlidir.

İlgili kavramlar

“Kurtarma grubunun kurtarma birimlerinin etkinleştirilmesi” sayfa 272
Bir kuyruk paylaşım grubu, grup kurtarma birimleri için destek yapılandırabilir ve bu desteği etkinleştirebilir.

“Uygulama desteği” sayfa 272

Bir grup kurtarma yok etme birimiyle hangi uygulamaların bağlanabileceğini belirlemek için bu sayfayı kullanın.

z/OS Kurtarma grubunun kurtarma birimlerinin etkinleştirilmesi

Bir kuyruk paylaşım grubu, grup kurtarma birimleri için destek yapılandırabilir ve bu desteği etkinleştirebilir.

Bir QSG içindeki bir kuyruk yöneticisinden kurtarma işlemi için GRUP birimlerini kullanmak için GROUPUR kuyruk yöneticisi özniteliğini etkinleştirin. Bu kavramla ilgili daha fazla bilgi için, bu konunun geri kalanını okumadan önce "z/OSüzerinde kurtarma yok etme birimi" sayfa 271 başlıklı konuya bakın.

GROUPUR kuyruk yöneticisi özniteliği geçerli kılındığında, kuyruk yöneticisi yeni bağlantıları bir grup kurtarma düzeni ile kabul eder. Bu yok etme özelliğini geçersiz kısanız, bu yok etme ile yeni bağlantılar kabul edilmez; ancak, bağlantı kesilinceye kadar bağlı olan uygulamalar etkilenmez.

Bir uygulama, bir grup kurtarma yok etme birimiyle bağlantı kurduğunda ve hangi işlemlerin belirsiz olduğunu sorguladığında ya da kuyruk paylaşım grubunun (QSG) başka bir yerde başlatılan bir işlemi çözme girişiminde bulunduğunda, artık bağlı olduğu kuyruk yöneticisi, isteği işleyebilmesi için kuyruk paylaşım grubunun diğer üyeleriyle iletişim kurabilmelidir. Bunu yapmak için, SYSTEM.QSG.UR.RESOLUTION.QUEUE. Bu kuyruk, CSQSYSAPPL adı verilen kurtarılabilir bir uygulama yapısında olmalıdır. Çözme istekleri işlenirken bu kuyruğun üzerinde kalıcı iletiler saklandığından, yapı kurtarılabilir olmalıdır.

GROUP birimlerini kurtarma işlemini etkinleştirmeden önce, bağlaşım olanağı yapısının ve paylaşılan kuyruğun tanımlandığından emin olmanız gerekir. CSQ4INSS örneğindeki tanımları kullanabilirsiniz. Kuyruk tanımlandığında ya da başlatma sırasında algılandığında, kuyruk paylaşım grubundaki her kuyruk yöneticisi, gelen istekleri alabilmesi için kuyruğu açar. Yanlış tanımlanmış olduğu için kuyruğu silmek ya da taşımak istiyorsanız, kuyruk nesnesini MQGET isteklerini engellemek için kuyruk nesnesini güncelleyerek, kuyruk yöneticilerinin üzerindeki açık tutamaçlarını kapatmalarını isteyebilirsiniz. Gereken düzeltmeleri yapınca, her kuyruk yöneticisini yeniden açmak için her kuyruk yöneticisini yönlendirdikten sonra, uygulamaların kuyruktan ileti almalarına izin verilmesi. Bir kuyruğun hangi tutamaçların açık olduğunu belirlemek için DISPLAY QSTATUS komutunu kullanın.

Bu kurulumu tamamladığınızda, işlem uygulamalarının bir grup kurtarma işlemi ile bağlantı kurabilmesini istediğiniz her kuyruk yöneticisiyle ilgili grup kurtarma birimlerini etkinleştirebilir. Bu, kuyruk paylaşım grubundaki kuyruk yöneticilerinin tümü olmamalıdır; ancak, bu işlevi yalnızca kuyruk paylaşım grubunun bir alt kümesinde etkinleştirmeyi seçerseniz, uygulamalarınızın yalnızca etkinleştirdiğiniz kuyruk yöneticilerine bağlanmayı denediğinden emin olmanız gerekir. Daha fazla bilgi için, bkz. "Uygulama desteği" sayfa 272.

GROUPUR kuyruk yöneticisi özniteliğini geçerli kılmaya çalıştığınızda, bir dizi yapılandırma denetimi gerçekleştirilir. Kuyruk yöneticisi şunları denetler:

- Bir kuyruk paylaşım grubuna ait.
- CSQ4INSSiçindeki tanımlamaya göre, SYSTEM.QSG.UR.RESOLUTION.QUEUE adlı paylaşılan kuyruk tanımlanmıştır.
- SYSTEM.QSG.UR.RESOLUTION.QUEUE , CSQSYSAPPL adı verilen kurtarılabilir bir CF yapısıdır.

Yukarıdaki denetimlerden biri başarısız olursa, GROUPUR özniteliği devre dışı kalır ve bir ileti kodu döndürülür.

Kuyruk yöneticisi özniteliği etkinleştirilmişse, bu yapılanış denetimleri de kuyruk yöneticisi başlatıldığında gerçekleştirilir. Başlatma grubu kurtarma birimlerinde yapılan denetimlerden herhangi biri geçersiz kılınırsa ve kuyruk yöneticisi, hangi denetmenin başarısız olduğunu tanımlayan bir ileti yayınlarsa. Gereken düzeltme işlemini gerçekleştirdiğinizde, kuyruk yöneticisi özniteliğini yeniden geçerli kılmanız gerekir.

z/OS Uygulama desteği

Bir grup kurtarma yok etme birimiyle hangi uygulamaların bağlanabileceğini belirlemek için bu sayfayı kullanın.

Support for the GROUP unit of recovery disposition is limited to certain types of transactional applications for which IBM MQ for z/OS is a resource manager but not the transaction coordinator. Şu anda desteklenen işlemsel uygulamalar şunlardır:

- IBM MQ genişletilmiş işlemsel istemci uygulamaları
- Bir uygulama sunucusunda çalışanIBM MQ classes for JMS uygulamaları (örneğin, WebSphere Application Server).
- CICS MQCONN kaynak tanımlaması RESYNCMEMBER (GROUPPRESYNC) ile yapılandırıldığında, CICS Transaction Server 4.2 ya da sonraki bir yayın düzeyiyle çalışanCICS uygulamaları.

İlgili kavramlar

“IBM MQ genişletilmiş işlemsel istemci uygulamaları” sayfa 273
IBM MQ Extended işlemsel istemci uygulamalarının GROUP birimini kurtarma atma birimini nasıl kullanabileceğini belirlemek için bu sayfayı kullanın.

“CICS uygulamalar” sayfa 273
CICS ' in kurtarma atma birimini nasıl kullanabileceğini belirlemek için bu sayfayı kullanın.

z/OS *IBM MQ genişletilmiş işlemsel istemci uygulamaları*

IBM MQ Extended işlemsel istemci uygulamalarının GROUP birimini kurtarma atma birimini nasıl kullanabileceğini belirlemek için bu sayfayı kullanın.

An example of an IBM MQ extended transactional client application is one that uses JMS and runs in WebSphere Application Server, connecting to IBM MQ over TCP/IP, rather than local bindings. Bu istemci uygulamaları, TCP/IP yoluyla olduğu gibi ağ bağlantıları üzerinden IBM MQ for z/OS ' e bağlanır. Bu uygulamalar için, bu değer, QMGR ya da GROUP biriminin kurtarma düzeni biriminin kullanılıp kullanılmadığını belirten xa_open çağrısında geçirilen xa_info dizesinin QMNAME parametresi için belirtilen değerdir. xa_open hakkında daha fazla bilgi için bkz. xa_open dizesinin biçimi ve xa_open için ek hata işleme. JMS uygulamaları için bu işlem, belirli bir kuyruk yöneticisi adı yerine ConnectionFactory içindeki kuyruk paylaşım grubunun (QSG) adını belirtilerek gerçekleştirilir.

XA istemci uygulamaları için, grup kurtarma (GROUP birimi) kurtarma biriminin kullanılmasından yararlanılması için, istemci uygulamalarınızın, belirli bir kuyruk yöneticisi yerine GROUPUR özniteliği etkinleştirilmiş olan kuyruk paylaşım grubundaki kuyruk yöneticilerine yöneltilmesini sağlamak için TCP/IP ayarınızı yapılandırmanız gerekir. Bunu yapmak için kullanabileceğiniz dinamik sanal IP adresi teknolojilerinden biri de z/OS SysPlex Distribütör'idir. Daha fazla ayrıntı için bkz. z/OS Communications Server ve z/OS Basic Skills: Dinamik sanal adresleme . Kuyruk paylaşım grubunuzdaki kuyruk yöneticilerinin bir alt kümesinde grup kurtarma birimlerini etkinleştirmek istiyorsanız, istemci uygulamalarınızın etkinleştirilmemiş olduğu kişilere yöneltilemediğinden emin olun. Bu, IBM WebSphere MQ 7.0.1' tan önceki bir sürümdeki herhangi bir kuyruk yöneticisini içerir.

Bir JMS istemci uygulamasında kurtarma yok etme özelliğinin grup birimini kullanmak için, IBM WebSphere MQ 7.0.1 JMS istemci kitaplıklarını kullanmanız gerekir. JMS istemci kitaplıkları için daha önceki düzeylerde, kuyruk yöneticisi (QMGR) kurtarma yok etme birimi kullanılır.

İstemci uygulamalarınız, paylaşılan kanalları kullanarak kuyruk paylaşım grubuna bağlanmak zorunda değildir.

z/OS *CICS uygulamalar*

CICS ' in kurtarma atma birimini nasıl kullanabileceğini belirlemek için bu sayfayı kullanın.

CICS 4.2 ve daha sonraki bir sürümü, bir MQCONN kaynak tanımlamasında RESYNCMEMBER (GROUPPRESYNC) grup yeniden eşzamanlama seçeneğini sağlar. Bu seçenekle yapılandırılmış bir CICS , CICS bölgesiyle aynı LPAR üzerinde çalışan bir kuyruk paylaşım grubundaki uygun bir kuyruk yöneticisine bağlanabilirler. CICS GROUPPRESYNC seçeneğini desteklemek için, bir kuyruk yöneticisi MQ V7.1 ya da sonraki bir sürümünde çalışır ve GROUPUR desteği için etkinleştirilmelidir.

GROUPPRESYNC ile MQ ' ya bağlı bir CICS bölgesi içinde çalışan hareketler, GROUP birimiyle kurtarma düzeni olan iş birimleri yaratır.

Bir kuyruk yöneticisi hatasından sonra daha hızlı kurtarma işlemini etkinleştirmek için, CICS bölgesinin aynı LPAR üzerinde çalışan bir diğer uygun kuyruk yöneticisine hemen bağlanmasını ve kuyruk yöneticisinin yeniden başlatılmasını beklemeden, belirsiz hareketleri gereken şekilde çözmesini sağlayarak daha hızlı kurtarma işlemini etkinleştirebilirsiniz.

RESYNCMEMBER (GROUPPRESYNC), CICSiçin daha esnek yeniden başlatma seçenekleri de sağlar. MQ bağlantısı olan GROUPPRESYNC ve MQ paylaşılan kuyruklarını kullanmak üzere yapılandırılmış bir CICS bölgesi, aynı kuyruk paylaşım grubunun üyesi olarak çalışan bir kuyruk yöneticisinin bulunduğu herhangi bir LPAR üzerinde yeniden başlatılabilir.

z/OS IBM MQ ve diğer z/OS ürünleri

IBM MQ ' in diğer z/OS ürünleriyle nasıl çalışabileceğini anlamak için bu konuyu kullanın.

İlgili kavramlar

“IBM MQ ve CICS” sayfa 274
All the CICS versions supported by IBM MQ 9.0.0, and later, use the CICS supplied version of the adapter and bridge.

“IBM MQ for z/OS ve WebSphere Application Server” sayfa 280
IBM MQ for z/OS ' in WebSphere Application Servertarafından kullanımını anlamak için bu konuyu kullanın.

İlgili başvurular

“IBM MQ ve IMS” sayfa 275
IBM MQ ' un IMSile nasıl çalıştığını anlamak için bu konuyu kullanın. IMS bağdaştırıcısı, kuyruk yöneticinizi IMS'e bağlamanızı ve IMS uygulamalarının MQI' yi kullanmasını sağlar.

“IBM MQ ve z/OS Batch, TSO ve RRS bağdaştırıcıları” sayfa 279
IBM MQ ' in z/OS Batch, TSO ve RRS bağdaştırıcılarıyla nasıl çalıştığını anlamak için bu konuyu kullanın.

z/OS IBM MQ ve CICS

All the CICS versions supported by IBM MQ 9.0.0, and later, use the CICS supplied version of the adapter and bridge.

IBM MQ CICS bağdaştırıcısını ve IBM MQ CICS bridge bileşenlerini yapılandırma hakkında daha fazla bilgi için, CICS belgelerinin MQile bağlantı yapılandırılması bölümüne bakın.

İlgili görevler

IBM MQ ile CICSkomutunu kullanma

z/OS CICS grup bağlantısı

CICS grup bağlantısı, bir CICS bölgesinin, tek bir kuyruk yöneticisi belirtmektense, aynı LPAR üzerinde bulunan IBM MQ kuyruk paylaşım grubunun etkin bir üyesinde bağlantı kurmasını sağlar. CICS , aynı anda hala tek bir kuyruk yöneticisine bağlanır.

CICS grup bağını desteklemek için LPAR üzerinde en az iki kuyruk yöneticisi gereklidir. Grup bağının kullanılması, etkin olması için belirli bir kuyruk yöneticisine gereksiniminiz olmadığı için daha yüksek kullanılabilirlik sağlar. CICS , LPAR üzerindeki kuyruk paylaşım grubundaki kuyruk yöneticisine bağlanır.

Daha fazla bilgi için, MQCONN kaynağındaki CICS belgelerine bakın.

CICS , bir kuyruk yöneticiyken geçirilen MQNAME ' e bağlanmayı dener:

- Kuyruk yöneticisi varsa ve etkin durumda ise, bağlantı çalışır.
- Bağlantı başarısız olursa, CICS aynı LPAR üzerinde etkin olan, gruptaki kuyruk yöneticilerinin durumunu sorgular.
- Birden çok kuyruk yöneticisi etkinse, CICS , CICS ' un belirli bir üyeye bağlanmak mı, yoksa belirli bir üyeye bağlanması mı, yoksa etkin mi olması gerektiğini belirlemek için RESYNCMEMBER (YES) ve UOW durumunu denetler.

- Belirli bir üyeye bağlanmaya gerek yoksa, CICS , bir kuyruk yöneticisi seçer (rasgele bir algoritma kullanarak).
- CICS , seçilen kuyruk yöneticisine bağlanmayı dener.
- Girişim başarısız olursa, dönüş koduna bağlı olarak, CICS bir sonraki üyeyi seçer, sonra seçim döngüünden yeniden geçer.
- Etkin kuyruk yöneticisi yoksa, CICS kuyruk yöneticisi listesine birden çok bağlantı gönderir ve ilk kuyruk yöneticisi kullanılabilir duruma gelinceye kadar ECBLIST ' te bekler.

İlgili kavramlar

“CICSiçin grup kurtarma birimleri (GROUPUR)” sayfa 275
The IBM MQ GROUPUR for CICS provides peer recovery for in-doubt units of work in a queue sharing group (QSG). Bir IBM MQ kuyruk yöneticisi, kuyruk paylaşım grubunda başka bir kuyruk yöneticisi adına çalışan belirsiz iş birimlerini çözebilir. Bunun anlamı şudur: CICS , QSG ' de farklı bir kuyruk yöneticisine grup bağlantısı yoluyla yeniden bağlanıyorsa, belirsiz hareketleri önceki bir IBM MQ bağlantısından çözebilir.

İlgili bilgiler

IBM MQ kuyruk paylaşım grupları için destek

z/OS CICSiçin grup kurtarma birimleri (GROUPUR)

The IBM MQ GROUPUR for CICS provides peer recovery for in-doubt units of work in a queue sharing group (QSG). Bir IBM MQ kuyruk yöneticisi, kuyruk paylaşım grubunda başka bir kuyruk yöneticisi adına çalışan belirsiz iş birimlerini çözebilir. Bunun anlamı şudur: CICS , QSG ' de farklı bir kuyruk yöneticisine grup bağlantısı yoluyla yeniden bağlanıyorsa, belirsiz hareketleri önceki bir IBM MQ bağlantısından çözebilir.

Bir CICS bölgesi kuyruk yöneticisiyle çalışıyorsa ve kuyruk yöneticisi olağan dışı sona erdirilirse, belirsiz hareketler kurtarılır. Bu, CICS bölgesinin, yeniden başlatmak için üzerinde çalıştığı kuyruk yöneticisini beklemesi ve daha sonra herhangi bir belirsiz iş birimi çözmesi gerisini ortadan kaldırır. Bunun anlamı, LPAR üzerinde en az iki kuyruk yöneticisi olması gerektiği anlamına gelir; böylece, CICS , ilk kuyruk yöneticisinin olağandışı olarak sona erdirilmesi durumunda başka bir kuyruk yöneticisine bağlanabilirler.

CICS MQCONN tanımlamasındaki yeni RESYNCMEMBER (GROUPPRESYNC) ayarı:

- IBM MQ grup ekleme işlevini ve eş kurtarmayı kullanır.
- GROUPUR özniteliği etkinleştirilmiş bir kuyruk yöneticisi gerektirir.
- Var olan CICS MQCONN RESYNCMEMBER ayarlarını hala destekle (YES ve NO):
  - Var olan CICS grup ekleme işlevini kullanır ve eşdüzey kurtarma işlemi yoktur.
  - Changing RESYNCMEMBER settings takes effect next time CICS connects to IBM MQ.

İlgili kavramlar

“Kurtarma grubunun kurtarma birimlerinin etkinleştirilmesi” sayfa 272
Bir kuyruk paylaşım grubu, grup kurtarma birimleri için destek yapılandırabilir ve bu desteği etkinleştirebilir.

z/OS IBM MQ ve IMS

IBM MQ ' un IMSile nasıl çalıştığını anlamak için bu konuyu kullanın. IMS bağdaştırıcısı, kuyruk yöneticinizi IMS'e bağlamanızı ve IMS uygulamalarının MQI' yi kullanmasını sağlar.

İsteğe bağlı ek IBM MQ - IMS köprüsü, uygulamaların, MQI ' yı kullanmayan bir IMS uygulamasını çalıştırmasına olanak sağlar. Bu, eski uygulamalarınızı yeniden yazmanıza gerek kalmadan IBM MQile birlikte kullanabildiğinizi gösterir.

Bu bileşenlerle ilgili daha fazla bilgi için aşağıdaki alt konulara bakın:

İlgili kavramlar

IBM MQ for z/OSüzerindeIMS ve IMS köprüsü uygulamaları

İlgili görevler

IMS bağdaştırıcısının ayarlanması

IMS köprüsünü ayarlama

IMS bağdaştırıcısının çalıştırılması

İlgili başvurular

MQIIH- IMS bilgi üstbilgisi

z/OS IMS bağdaştırıcısı

IMS bağdaştırıcısı, IMS uygulama programları ile IBM MQ altsistemi arasında bir arabirimdir.

IBM MQ bağdaştırıcıları, ileti kuyruklama ağı aracılığıyla ileti göndermek ve almak için farklı uygulama ortamlarının etkinleştirilmesini sağlar. IMS bağdaştırıcısı, IMS uygulama programları ve bir IBM MQ altsistemi arasındaki arabirimdir. Bu, IMS uygulama programlarının MQI ' yi kullanmasına olanak sağlar.

IMS bağdaştırıcısı, IMStarafından sağlanan External Subsystem Attach Facility (ESAF) kullanılarak IBM MQ erişimi isteklerini alır ve yorumlar. Bu olanak, *IMS Customization Guide*adlı kılavuzda açıklanmaktadır. Genellikle, IMS , işletmen müdahalesi olmadan IBM MQ ' a otomatik olarak bağlanır.

IMS bağdaştırıcısı, aşağıdaki kiplerde ya da eyaletlerde çalışan programlar için IBM MQ kaynaklarına erişim sağlar:

- Görev (TCB) kipi
- Sorun durumu
- Çapraz olmayan bellek kipi
- Erişim olmayan kayıt kipi

Bağdaştırıcı, bir uygulama görevi denetim bloğundan (TCB) IBM MQ' e bir bağlantı iş parçacığı sağlar.

The adapter supports a two-phase commit protocol for changes made to resources owned by IBM MQ with IMS acting as the syncpoint coordinator. Conversations where IMS is not the syncpoint coordinator, for example APPC- protected (SYNCLVL=SYNCPT) conversations, are not supported by the IMS adapter.

Bağdaştırıcı, bir tetikleyici izleme işlemi (CSQQTRMN) de sağlar. Bu, “IMS tetikleme izleme programı” sayfa 277içinde açıklanmaktadır.

You can use IBM MQ with the IMS Extended Recovery Facility (XRF) to aid recovery from a IMS error. XRF ile ilgili ek bilgi için *IMS Administration Guide: System* elkitabına bakın.

Bağdaştırıcının kullanılması

Uygulama programları ve IMS bağdaştırıcısı aynı adres alanında çalışır. Kuyruk yöneticisi, kendi adres alanında ayrıdır.

Bir ya da daha çok MQI çağrısı içeren her programı, uygun bir IMS dil arabirimi modülüne bağlamanız ve devingen MQI çağrıları kullanmadığı sürece, IBM MQtarafından sağlanan API kod parçası programı, CSQQKOD (API kod parçası) programını bağlamalısınız. Uygulama bir MQI çağrısı yayınlarken, sınırlı kod öbeği, IMS dış altsistem arabirimi aracılığıyla bağdaştırıcıya denetim aktarır; bu arabirim, isteğin ileti kuyruğu yöneticisi tarafından işlenmesini yönetir.

IMSile sistem denetimi ve işlemi

Yetkili bir IMS uçbirim işletmeni, IBM MQkomutlarını denetlemek ve izlemek için IMS komutlarını yayınlayabilir. However, the IMS terminal operator has no control over the IBM MQ address space. Örneğin, işleç IBM MQ adresini bir IMS adres alanından kapatamazsınız.

Kısıtlamalar

Aşağıdaki IBM MQ API çağrıları, IMS bağdaştırıcısı kullanılarak bir uygulama içinde desteklenmez:

- MQCB
- MQCB_FUNC
- MQCTL

IMS tetikleme izleme programı

The IMS trigger monitor (**CSQQTRMN**) is an IBM MQ-supplied IMS application that starts an IMS transaction when an IBM MQ event occurs, for example, when a message is put onto a specific queue.

Nasıl çalışır

Bir ileti, bir uygulama ileti kuyruğuna konduğunda, tetikleme koşulları karşılanırsa bir tetikleyici oluşturulur. Daha sonra kuyruk yöneticisi, *tetikleyici iletio*larak bilinen bir iletiyi (bazı kullanıcı tanımlı verileri içeren) yazar. Bu ileti kuyruğu için belirtilmiş olan başlatma kuyruğuna (kullanıcı tanımlı veriler) yazar. Bir IMS ortamında, bir başlatma kuyruğunu izlemek ve tetikleme iletilerini vardıkları şekilde almak için CSQQTRMN yönetim ortamını başlatabilirsiniz. Tipik olarak, CSQQTRMN başka bir IMS işlemini bir INSERT (ISRT) tarafından IMS ileti kuyruğuna zamanlar. Başlatılan IMS uygulaması, iletiyi uygulama ileti kuyruğundan okur ve daha sonra bu iletiyi işler. CSQQTRMN, ileti dışı BMP olarak çalışmalıdır.

CSQQTRMN hizmetlerinin her kopyası tek bir başlatma kuyruğu sağlar. When it has started, the trigger monitor runs until IBM MQ or IMS ends.

CSQQTRMN için APPLCTN makrosu SCHDTYP=PARALLEL belirtmelidir.

Tetikleme izleme programı, toplu iş odaklı bir BMP olduğundan, tetikleyici izleyicisi tarafından başlatılan IMS işlemleriyle aşağıdaki işlemleri içerir:

- IOPCB ' nin LTERM alanındaki boşluklar
- Tetikleyicinin PSB adı, IOPCB 'nin Kullanıcı Kimliği (Userid) alanındaki BMP' nin adı.

Hedef IMS işlemi Security Server (önceden RACF olarak biliniyorsa) tarafından korunuyorsa, Security Server sunucusunun bir kullanıcı kimliği olarak CSQQTRMN ' yi tanımlamanız gerekebilir.

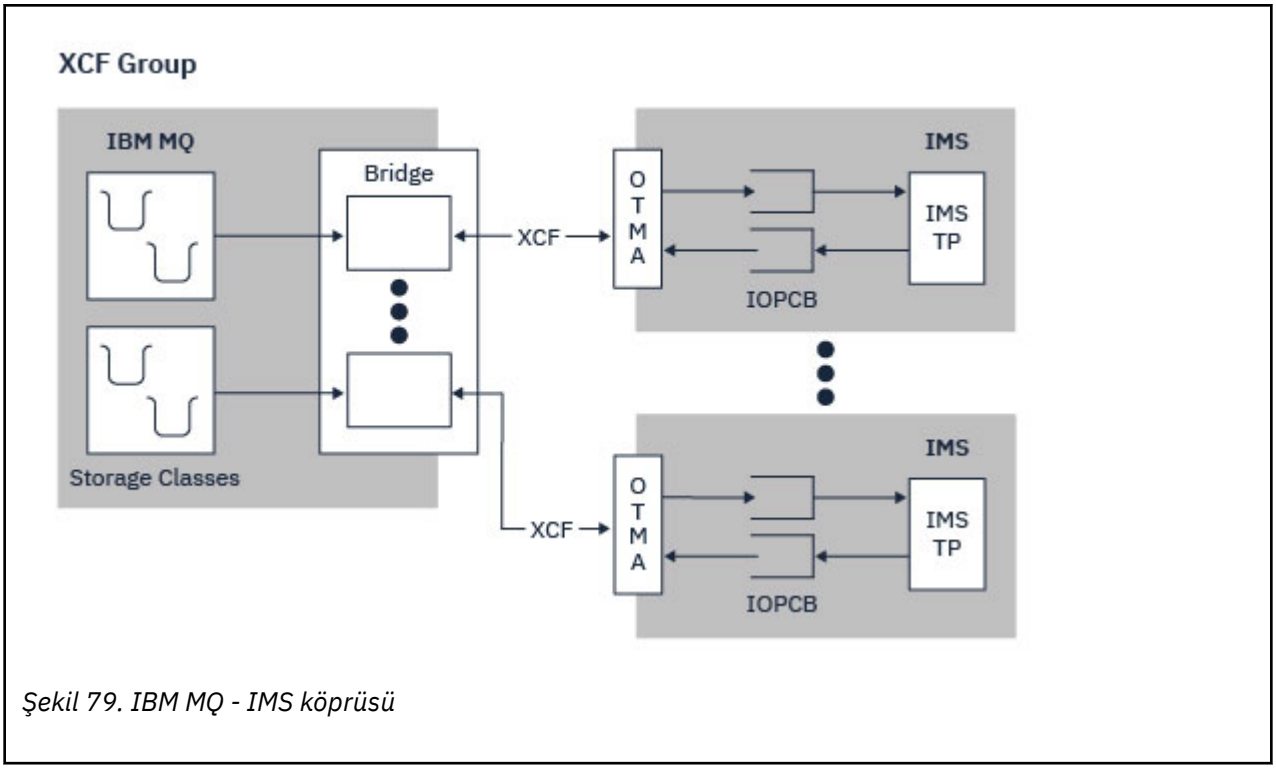
z/OS IBM MQ - IMS köprüsü

The IBM MQ - IMS bridge is the component of IBM MQ for z/OS that allows direct access from IBM MQ applications to applications on your IMS system.

IBM MQ - IMS köprüsü, *örtük MQI desteği*' i etkinleştirir. Bu, 3270 bağlantılı uçbirimlerin denetlediği eski uygulamaları IBM MQ iletileri tarafından denetlenmek, yeniden yazmak, yeniden derlemek ya da yeniden bağlamak zorunda kalmaksızın, yeniden düzenleyebileceğiniz anlamına gelir. Köprü, IMS *Open Transaction Manager Access* (OTMA) istemcisinden biri.

Köprü uygulamalarında, IMS uygulaması içinde IBM MQ çağrısı yoktur. Uygulama, IOPCB 'ye bir GET UNIQUE (GU) kullanarak girişini alır ve IOPCB' ye bir ISRT kullanarak çıkışını gönderir. IBM MQ applications use the IMS header (the MQIIH structure) in the message data to ensure that the applications can execute as they did when driven by nonprogrammable terminals. Çok bölümlü iletileri işleyen bir IMS uygulaması kullanıyorsanız, tüm bölümlerin tek bir IBM MQ iletisi içinde bulundurulması gerektiğini unutmayın.

IMS köprüsü Şekil 79 sayfa 278içinde gösterilmektedir.



Şekil 79. IBM MQ - IMS köprüsü

Bir kuyruk yöneticisi bir ya da daha çok IMS sistemine bağlanabilir ve birden çok kuyruk yöneticisi tek bir IMS sistemine bağlanabilir. Tek kısıtlama, bunların tümünün aynı XCF grubuna ait olması ve tümü aynı sysplex içinde olması gerekir.

Bir IMS köprüsü ayarına ve aynı kuyruk yöneticisine ek bir IMS bağlantısı eklemeye ilişkin bilgi için bkz. IMS köprüsünün ayarlanması .

OTMA nedir?

IMS OTMA olanağı, IMS 5.1 ya da sonraki bir yayın düzeyiyle çalışan, hareket tabanlı bir bağlantısız istemci/sunucu protokolüdür. z/OS *Cross Systems Coupling Facility* (XCF) aracılığıyla IMS TM uygulamalarına erişen anasistem tabanlı iletişim sunucuları için bir arabirim olarak işlev görür.

OTMA, müşteriler ile büyük bir ağ ya da çok sayıda oturum için IMS ileclientsarasındaki etkileşimler için yüksek performans sağlamak üzere müşterilerin IMS ' a bağlanmasını sağlar. OTMA, bir z/OS sistem şebekesi ortamında uygulanır. Bu nedenle, OTMA 'nın etki alanı XCF' nin etki alanıyla sınırlıdır.

OTMA Kaynak İzleme

Support for the x'3C' OTMA protocol messages, available in IMS v10 or higher, has been added to the IBM MQ - IMS bridge in IBM MQ for z/OS v7.1. Bu iletiler, sağlıklı işletim durumunu raporlamak için IMS tarafından OTMA istemcilerine gönderilir. Bir IMS ortağı, gönderilmekte olan işlem isteklerinin hacmini işleyemiyorsa, bir sel uyarısının ortaya çıktığını IBM MQ ' a bildirecektir. IBM MQ yanıtında, isteklerin köprü üzerinden gönderilme hızının yavaşlayacağı bir durum ortaya çıktı. IMS hala işlem isteklerini işleyemiyorsa ve tam bir sel koşulu, IMS ortağının tüm TPIPE ' lerin askıya alınmadıysa gerçekleşir. IMS ortağının, sel ya da sel uyarı koşulunun giderildiğine ilişkin bildirim üzerine, IBM MQ 'in askıya alınmış tüm TPIP' lere (uygunsa) devam etmesi ve hız üst sınırına ulaşılıncaya kadar işlem isteklerinin gönderileceği hızı kademeli olarak artırması gerekir. Console messages are issued by IBM MQ in response to a change in the status of IMS partners.

IMS v10 ortakları kullanılıyorsa, PTF UK45082 ' in uygulandığını doğrulamalısınız.

Submitting IMS transactions from IBM MQ

Köprüyü kullanan bir IMS işlemini göndermek için, uygulamalar her zamanki gibi bir IBM MQ kuyruğuna ileti yerleştirmektedir. İletiler IMS işlem verilerini içerir; IMS üstbilgisine (MQIIH yapısı) sahip olabilir ya da IBM MQ - IMS köprüsünün, iletteki verilerle ilgili varsayımları gerçekleştirmesine izin verebilir.

IBM MQ daha sonra, iletiyi bir IMS kuyruğuna koyar (veri bütünlüğünü sağlamak için eşitleme noktalarının kullanılmasını sağlamak için önce IBM MQ ' da kuyruğa alınır). The storage class of the IBM MQ queue determines whether the queue is an *OTMA kuyruğu* (that is, a queue used to transmit messages to the IBM MQ - IMS bridge) and the particular IMS partner to which the message data is sent.

Remote queue managers can also start IMS transactions by writing to these OTMA queues on IBM MQ for z/OS.

IMS sisteminden döndürülen veriler, doğrudan ileti tanımlayıcı yapısında (MQMD) belirtilen IBM MQ yanıtlama kuyruğuna yazılıdır. (Bu, MQMD ' nin *ReplyToQMgr* alanında belirtilen kuyruk yöneticisine bir iletim kuyruğu olabilir.)

İlgili kavramlar

IBM MQ for z/OSüzerindeIMS ve IMS köprüsü uygulamaları

İlgili görevler

IMS köprüsünün uyarlanması

İlgili başvurular

“IBM MQ ve IMS” sayfa 275

IBM MQ ' un IMSile nasıl çalıştığını anlamak için bu konuyu kullanın. IMS bağdaştırıcısı, kuyruk yöneticinizi IMS'e bağlamanızı ve IMS uygulamalarının MQI' yi kullanmasını sağlar.

z/OS IBM MQ ve z/OS Batch, TSO ve RRS bağdaştırıcıları

IBM MQ ' in z/OS Batch, TSO ve RRS bağdaştırıcılarıyla nasıl çalıştığını anlamak için bu konuyu kullanın.

Toplu İş bağdaştırıcılarına giriş

Batch/TSO bağdaştırıcıları, JES, TSO ya da z/OS UNIX System Servicesaltında çalışan IBM MQ ve z/OS uygulama programları arasındaki arabirimdir. Bu bağdaştırıcılar, z/OS uygulama programlarının MQI ' yi kullanmasını sağlar.

Bağdaştırıcılar, aşağıdaki kiplerde ya da durumlarda çalışan programlar için IBM MQ kaynaklarına erişim sağlar:

- Görev (TCB) kipi
- Sorun ya da gözetmen durumu
- Çapraz olmayan bellek kipi
- Erişim olmayan kayıt kipi

Uygulama programları ile IBM MQ arasındaki bağlantılar görev düzeyinde bulunur. Bağdaştırıcılar, bir uygulama görev denetim bloğundan (TCB) IBM MQ' e bir bağlantı iş parçacığı sağlar.

Batch/TSO bağdaştırıcısı, IBM MQ' e ait kaynaklarda yapılan değişiklikler için tek aşamalı kesinleştirme protokolünü destekler. Çok aşamalı kesinleştirme protokollerini desteklemez. RRS bağdaştırıcısı, IBM MQ uygulamalarının, z/OS Resource Recovery Services (RRS) tarafından koordine edilen diğer RRS etkin ürünlerle iki aşamalı kesinleştirme protokollerine katılmasına olanak sağlar.

Bağdaştırıcılar, her saniye zamanuyumsuz bir olay zamanlamak için z/OS STIMERM hizmetini kullanır. Bu olay, toplu iş uygulamasının görevi tarafından herhangi bir bekleme gerektirmeyen bir kesme isteği öbeği (IRB) çalıştırır. Bu IRB, IBM MQ sonlandırma ECB ' nin gönderilip gönderilmediğini denetler. Sonlandırma ECB 'si gönderildiyse, IRB, IBM MQ içinde bir olay üzerinde bekleyen uygulama ECB ' lerini (örneğin, bir sinyal ya da bekleme) göndermektedir.

Batch/TSO bağdaştırıcısı

IBM MQ Batch/TSO bağdaştırıcısı, z/OS Toplu İş ve TSO uygulamaları için IBM MQ desteği sağlar. z/OS Batch ya da TSO altında çalışan tüm uygulama programları, CSQBCOB API kod parçası programının bunlarla birlikte düzenlenmesine sahip olmalıdır. Sınırlı kod öbeği, uygulamaya tüm MQI çağrılarına erişimli bir uygulama sağlar. You use single-phase commit and backout for applications by issuing the MQI calls **MQCMIT** and **MQBACK**.

RRS bağdaştırıcısı

Resource Recovery Services (RRS), z/OS ürünlerinin iki aşamalı kesinleştirmesini koordine etmek için sistem çapında bir hizmet sağlayan z/OS alt bileşenidir. IBM MQ Batch/TSO RRS bağdaştırıcısı (RRS bağdaştırıcısı), bu hizmetleri kullanmak isteyen z/OS Batch ve TSO uygulamaları için IBM MQ desteği sağlar. RRS bağdaştırıcısı, IBM MQ ' in RRS koordinasyonunda tam bir katılımcı olmasını sağlar. Uygulamalar, RRS ' yi destekleyen diğer ürünlerle (örneğin, Db2) iki aşamalı kesinleştirme işlemleriyle katılabilirler.

RRS bağdaştırıcısı iki sınırlı kod öbeği sağlar; RRS ' yi bu sınırlı kod öbeklerinden biriyle kullanmak isteyen uygulama programlarını bağlamanız gerekir.

CSQBRSTB
Bu sınırlı kod öbeği, RRS çağrılabilir kaynak kurtarma hizmetlerini **MQCMIT** ve **MQBACK**adlı MQI çağrıları yerine kullanarak uygulamalar için iki aşamalı kesinleştirme ve geri alma olanağını kullanmanıza olanak sağlar.

Ayrıca, SYS1.CSSLIB ' i uygulamanızı sağlar. If you use the MQI calls **MQCMIT** and **MQBACK**, you will receive return code `MQRC_ENVIRONMENT_ERROR`.

CSQBRRSI
Bu sınırlı kod öbeği, **MQCMIT** ve **MQBACK** adlı MQI çağrılarını kullanmanızı sağlar; IBM MQ , bu çağrıları **SRRCMIT** ve **SRRBACK** RRS çağrıları olarak gerçekleştirmektedir.

RRS bağdaştırıcısını kullanan uygulama programlarının oluşturulmasıyla ilgili bilgi için bkz. RRS toplu iş bağdaştırıcısı.

z/OS Batch, TSO ve RRS bağdaştırıcılarına ilişkin daha fazla bilgi nerede bulunmanız gerekir

Bu bölümdeki konularla ilgili daha fazla bilgiyi aşağıdaki kaynaklarda bulabilirsiniz:

Çizelge 24. Where to find more information about using z/OS Batch with IBM MQ

Konu	Nereye bakılacağı
Toplu iş bağdaştırıcılarının ayarlanması	Görev 19: Toplu İş, TSO ve RRS bağdaştırıcılarını ayarlayın
RRS çağrılabilir kaynak kurtarma hizmetleri	*MVS Programlama: Yüksek Düzeyli Diller için Callable Services*

z/OS IBM MQ for z/OS ve WebSphere Application Server

IBM MQ for z/OS ' in WebSphere Application Servertarafından kullanımını anlamak için bu konuyu kullanın.

WebSphere Application Server altında çalışan Java yazılımında yazılan uygulamalar, ileti sistemini gerçekleştirmek için Java Messaging Service (JMS) belirtimini kullanabilir. Bu ortamda noktadan noktaya ileti alışverişi, bir IBM MQ for z/OS kuyruk yöneticisi tarafından sağlanabilir.

A benefit of using an IBM MQ for z/OS queue manager to provide the messaging is that connecting JMS applications can participate fully in the functionality of an IBM MQ network. Örneğin, bunlar IMS köprüsünü kullanabilir ya da diğer platformlarda çalışan kuyruk yöneticileriyle ileti alışverişi yapabilir.

WebSphere Application Server ile kuyruk yöneticisi arasında bağlantı

Kuyruk bağlantısı üreticisi nesnesi için *istemci iletimi* ya da *bağ tanımları aktarımı* seçeneğini belirleyebilirsiniz. Bağ tanımlarını taşımanızı seçerseniz, WebSphere Application Server ve kuyruk yöneticisi aynı z/OS görüntüsünde bulunmalıdır.

*Bağ tanımları iletimi*için yerli kitaplıklar gerektiğini unutmayın.

Her iki bağlantı tipi de işlemsel uygulamaları destekler: XA protokolleri kullanılarak istemci iletimi; RRS hizmetlerini kullanan bir WebSphere Application Server kod parçası olan CSQBWSTB kullanılarak bağ tanımları iletimi.

Kuyruk bağlantısı üreticileri yapılandırma hakkında daha fazla bilgi için bkz. *IBM MQ Using Java* .

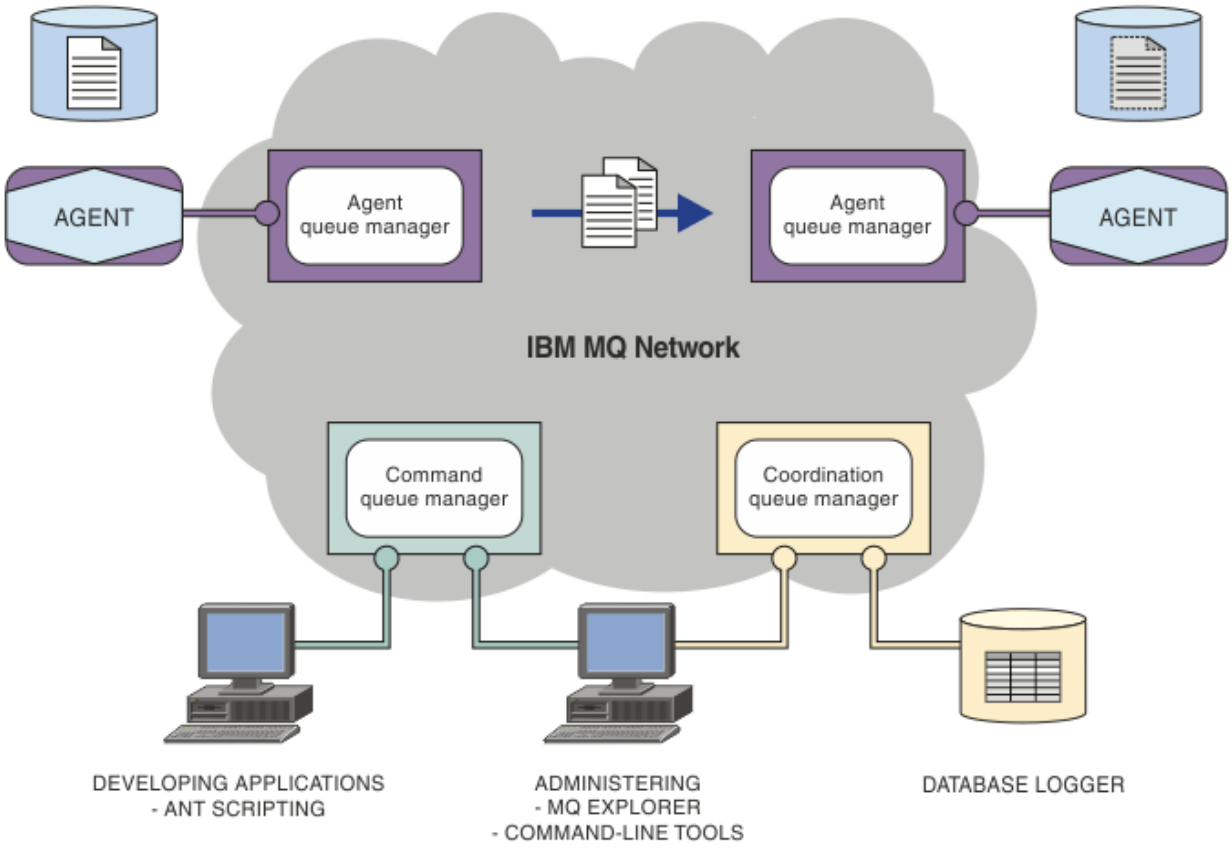
Using IBM MQ functions from JMS applications

Varsayılan olarak, IBM MQ kuyruğunda tutulan JMS iletileri, JMS ileti üstbilgisi bilgilerinin bir kısmını tutmak için bir MQRFH2 üstbilgisini kullanır. Birçok eski IBM MQ uygulaması bu üstbilgileri içeren iletileri işleyemez ve kendi karakteristik üstbilgilerini (örneğin, CICS Bridge için MQCIH ya da IBM MQ Workflow uygulamaları için MQWIH) gerektiremez. Bu özel konularla ilgili daha fazla ayrıntı için bkz. JMS iletilerini IBM MQ iletilerine eşleme.

Managed File Transfer

Managed File Transfer , dosya boyutuna ya da kullanılan işletim sistemlerine bakılmaksızın, yönetilen ve denetlenebilir bir şekilde sistemler arasında dosya aktarır.

You can use Managed File Transfer to build a customized, scalable, and automated solution that enables you to manage, trust, and secure file transfers. Managed File Transfer , yüksek maliyetli fazlalıkları ortadan kaldırır, bakım maliyetlerini düşürür ve mevcut BT yatırımlarınızı en üst düzeye çıkarır.



Çizge yalın bir Managed File Transfer topolojisini gösterir. Her biri bir IBM MQ ağında kendi aracı kuyruk yöneticilerine bağlanan iki aracı vardır. A file is transferred from the agent on the one side of the diagram, through the IBM MQ network, to the agent on the other side of the diagram. Ayrıca, IBM MQ ağında koordinasyon kuyruğu yöneticisi ve bir komut kuyruğu yöneticisi bulunur. Applications and tools connect to these queue managers to configure, administer, operate, and log Managed File Transfer activity in the IBM MQ network.

Managed File Transfer , işletim sisteminize ve genel ayarınıza bağlı olarak dört farklı seçenek olarak kurulabilir. Bu seçenekler şunlardır: Managed File Transfer Agent, Managed File Transfer Logger, Managed File Transfer Serviceya da Managed File Transfer Tools. Ek bilgi için Managed File Transfer product optionsbaşlıklı konuya bakın.

Aşağıdaki görevleri gerçekleştirmek için Managed File Transfer ' u kullanabilirsiniz:

- Yönetilen dosya aktarımları yarat
  - Linux Windows Linux ya da Windows platformlarında IBM MQ Explorer ' dan yeni dosya aktarımları oluşturun.
  - Desteklenen tüm altyapılarda, komut satırından yeni dosya aktarımları yaratın.
  - Dosya aktarma işlevini Apache Ant aracından bütünleştirin.
  - Aracı komut kuyruklarına ileti koyarak Managed File Transfer ' i denetleyen uygulamalar yazın.
  - Dosya aktarımlarını daha sonra gerçekleşecek şekilde zamanlayın. Ayrıca, bir dosya sistemi olayı aralığına dayalı olarak zamanlanan dosya aktarımlarını da tetikleyebilirsiniz; örneğin, yeni bir dosya yaratılmakta.
  - Bir kaynağı sürekli olarak izleyin; örneğin, bir dizin ve bu kaynağın içeriği önceden tanımlanmış bazı koşulları yerine getirdiğinde, bir görev başlatın. Bu görev bir dosya aktarımı, bir Ant komut dosyası ya da bir JCL işi olabilir.
  - Dosyaları IBM MQ kuyruklarına ve bu kuyruklara aktarabilirsiniz.

  - FTP, FTPS ya da SFTP sunucularına dosya aktarın.
  - Dosyaları Connect:Direct düğümlerine venodesdüğümlerine aktarabilirsiniz.
  - Hem metin hem de ikili dosyaları aktarın. Metin dosyaları, kaynak ve hedef sistemlerin kod sayfaları ile son satır sonu kuralları arasında otomatik olarak dönüştürülür.
  - Güvenli Yuva Katmanı (SSL) tabanlı bağlantılara ilişkin sektör standartlarını kullanarak aktarımlar güvenli kılınabilir.
- İlerlemedeki aktarımları görüntüleme ve ağınızdaki tüm aktarımlara ilişkin günlük bilgileri
  - Linux Windows View the status of transfers in progress from IBM MQ Explorer on Linux or Windows platforms.
  - Linux Windows Check the status of completed transfers by using the IBM MQ Explorer on Linux or Windows platforms.
  - Günlük iletilerini bir Db2 ya da Oracle veritabanına kaydetmek için Managed File Transfer veritabanı günlüğe kaydedici özelliğini kullanın.

Managed File Transfer is built on IBM MQ, which provides assured, once-only delivery of messages between applications. IBM MQ' un çeşitli özelliklerinden faydalanabilirsiniz. Örneğin, aracıları IBM MQ kanalları üzerinden gönderdiğiniz verileri sıkıştırmak ve aracılar arasında gönderdiğiniz verileri güvenli kılmak için SSL kanallarını kullanarak kanal sıkıştırma özelliğini kullanabilirsiniz. Dosyalar güvenilir bir şekilde aktarılır ve dosya aktarımın gerçekleştirildiği alt yapının hatasına göz yumabilir. Bir ağ kesintisi yaşarsanız, bağlantı geri yüklendiğinde dosya aktarımı, kapandığı yerden yeniden başlatılır.

Dosya aktarımının var olan IBM MQ ağınızla birleştirilerek, iki ayrı altyapıyı sürdürmek için gereken kaynakları harcamaktan kaçınabilirsiniz. If you are not already an IBM MQ customer, by creating an IBM MQ network to support Managed File Transfer you are building the backbone for a future SOA implementation. Zaten bir IBM MQ müşteriyseniz, Managed File Transfer , IBM MQ Internet Pass-Thru ve IBM Integration Busde içinde olmak üzere, var olan IBM MQ altyapınızdan yararlanabilir.

Managed File Transfer yapılandırmanızın esnekliğini artırmak için IBM MQ yüksek kullanılabilirlik çözümlerinden yararlanmanız gerekir. Aracılarınız eşlenmiş veri kuyruğu yöneticilerini (RDQM ' ler) kullanırsa, bu durumda kayan IP adresi özelliğini kullanmak için bunları yapılandırmanız gerekir. Başka bir deyişle, aracılar, şu anda çalışmakta olan üç RDQM eşgörünümlerinden hangisi ile iletişim kurmak için aynı IP adresini kullanır ve hata durumunda otomatik olarak yeniden bağlantı kurar (bkz. RDQM yüksek kullanılabilirliği ve Yüzer IP adresi oluşturma ve silme). Çok eşgörünümli kuyruk yöneticisi çözümünü kullanıyorsanız, uygulamalar her yönetim ortamıyla iletişim kurmak için farklı bir IP adresi kullanır; bu, istemci tarafından yedek sisteme geçiş sırasında yeniden bağlanma (bkz. Çok eşgörünümli kuyruk yöneticileri ve Kanal ve istemci yeniden bağlantısı).

Managed File Transfer , bir dizi diğer IBM ürünüyle bütünleşir:

IBM Integration Bus

Bir IBM Integration Bus akışının bir parçası olarak Managed File Transfer tarafından aktarılmış olan süreç dosyaları. Daha fazla bilgi için bkz. IBM Integration Bus' tan MFT ile çalışma.

IBM Sterling Connect:Direct

Managed File Transfer Connect:Direct köprüsünü kullanarak, var olan bir Connect:Direct ağına dosya aktarın. Daha fazla bilgi için bkz. Connect:Direct köprüsü.

IBM Tivoli Composite Application Manager

IBM Tivoli Composite Application Manager , koordinasyon kuyruğu yöneticisine yayınlanan bilgileri izlemek için kullanabileceğiniz bir aracı sağlar.

İlgili kavramlar

Yönetilen Dosya Aktarma ürün seçenekleri

“MFT topolojiye genel bakış” sayfa 284

An overview of how Managed File Transfer agents are connected with the coordination queue manager in an IBM MQ network.

“MFT , IBM MQile nasıl çalışır?” sayfa 284
Managed File Transfer , IBM MQile bir dizi şekilde etkileşimde bulunur.

MFT , IBM MQile nasıl çalışır?

Managed File Transfer , IBM MQile bir dizi şekilde etkileşimde bulunur.

- Managed File Transfer , her bir dosyayı bir ya da daha fazla iletiye bölerek ve iletileri IBM MQ ağınız aracılığıyla ileterek aracı işlemleri arasında dosya aktarır.
- The agent processes move file data by using nonpersistent messages to minimize the impact on your IBM MQ logs. Başka bir aracı işlemleriyle iletişim kurarak, dosya verilerini içeren iletilerin akışını düzenler. Bu, IBM MQ iletim kuyruklarında dosya verilerini içeren iletileri önler ve kalıcı olmayan iletilerden herhangi birinin teslim edilmemesi durumunda, dosya verilerinin yeniden gönderilmesini sağlar.
- Managed File Transfer aracıları, IBM MQ kuyruklarının sayısını kullanır. Ek bilgi için MFT sistem kuyrukları ve sistem konusubaşlıklı konuya bakın.
- Bu kuyrukların bazıları kesinlikle dahili kullanım için olsa da, aracı, aracının okuduğu belirli bir kuyruğa gönderilen özel olarak biçimlendirilmiş komut iletileri biçiminde istekleri kabul edebilir. Hem komut satırı komutları, hem de IBM MQ Explorer eklentisi, aracıya aranan işlemi gerçekleştirmesini bildirmek için IBM MQ iletilerini aracıya gönderir. Aracıyla etkileşimde bulunan IBM MQ uygulamalarını bu şekilde yazabilirsiniz. Daha fazla bilgi için bkz. Aracı komut kuyruğuna ileti yerleştirerek MFT ' i denetleme.
- Managed File Transfer aracıları durumlarıyla ilgili bilgi gönderir ve eşgüdüm kuyruk yöneticisi olarak atanan bir MQ kuyruk yöneticisine aktarımların ilerlemesi ve sonucu hakkında bilgi gönderir. Bu bilgiler, koordinasyon kuyruk yöneticisi tarafından yayınlanır ve aktarım ilerlemesini izlemek isteyen ya da oluşan aktarımların kayıtlarını tutmak isteyen uygulamalara abone olabilir. Hem komut satırı komutları, hem de IBM MQ Explorer eklentisi yayınlanan bilgileri kullanabilir. Bu bilgileri kullanan IBM MQ uygulamalarını yazabilirsiniz. Bilgilerin yayınlandığı konu hakkında daha fazla bilgi için bkz. SYSTEM.FTE konusu.
- Managed File Transfer anahtar bileşenleri, IBM MQ kuyruk yöneticilerinin iletileri saklamaya ve iletmeye ilişkin yeteneklerinden yararlanir. Bunun anlamı, bir kesinti yaşamanız durumunda altyapınızın etkilenmemiş kısımlarından dosya aktarmaya devam edebileceğinin anlamına gelir. Bu, mağaza ve ileri ve dayanıklı aboneliklerin birleşiminin, koordinasyon kuyruk yöneticisinin, gerçekleşen dosya aktarımları hakkında anahtar bilgileri kaybetmeden kullanılamamasına izin vermesi için koordinasyon kuyruk yöneticisine genişletir.

MFT topolojiye genel bakış

An overview of how Managed File Transfer agents are connected with the coordination queue manager in an IBM MQ network.

Managed File Transfer aracıları, aktarılan dosyaları gönderir ve alır. Her aracının, ilişkili kuyruk yöneticisinde kendi kuyruk kümesi vardır ve aracı, kuyruk yöneticisine bağ tanımlarında ya da istemci kipinde bağlanır. Bir aracı, kuyruk yöneticisi olarak eşgüdüm kuyruk yöneticisini de kullanabilir.

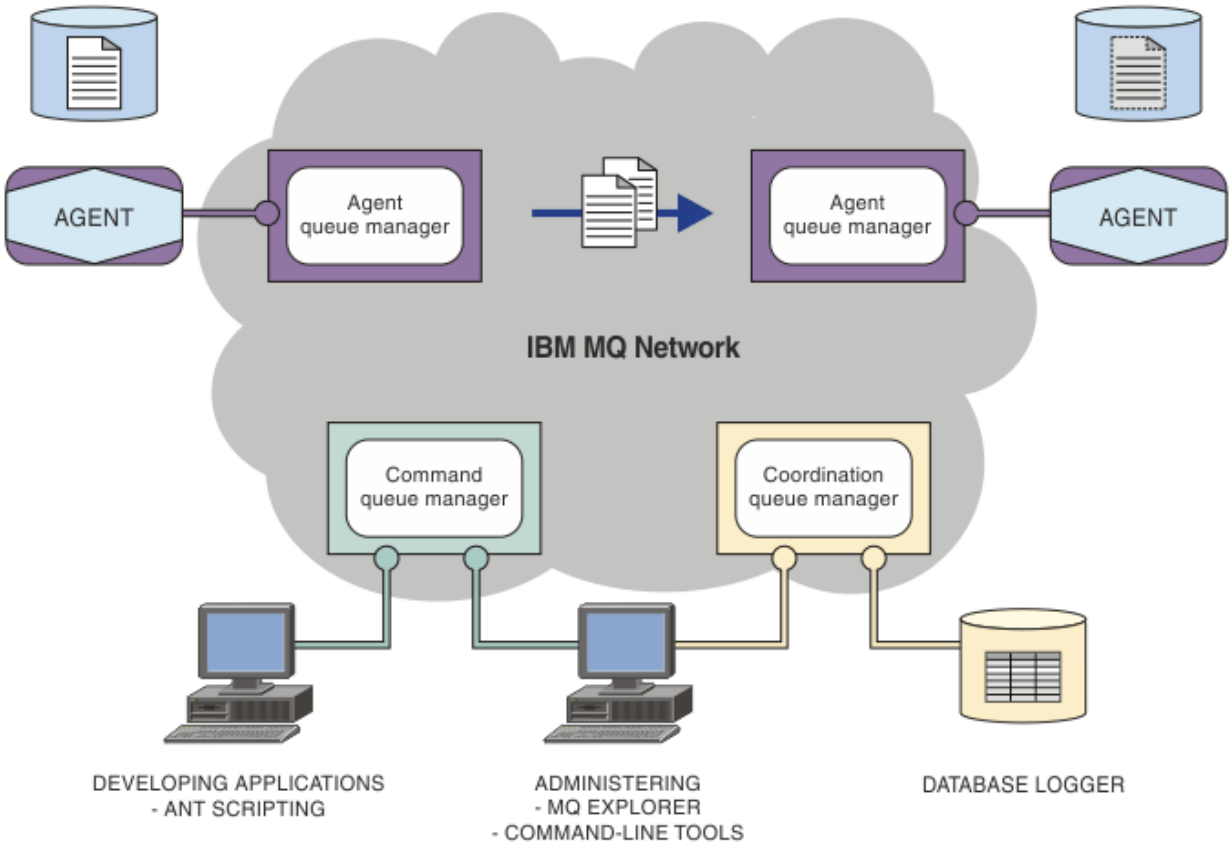
Koordinasyon kuyruk yöneticisi denetim ve dosya aktarma bilgilerini yayınlar. Eşgüdüm kuyruk yöneticisi, aracı derlemi, aktarma durumu ve aktarma denetim bilgileri için tek bir noktayı temsil eder. Aktarımların yer alması için eşgüdüm kuyruk yöneticisinin kullanılabilir durumda olması gerekmez. Eşgüdüm kuyruk yöneticisi geçici olarak kullanılamaz duruma gelirse, aktarımlar olağan şekilde devam eder. Denetim ve durum iletileri, eşgüdüm kuyruk yöneticisi kullanılabilir duruma gelinceye kadar aracı kuyruk yöneticilerinde saklanır ve daha sonra olağan şekilde işlenebilir.

Aracılar, koordinasyon kuyruğu yöneticisine kaydolur ve ayrıntılarını o kuyruk yöneticisine yayınlar. This agent information is used by the Managed File Transfer plugin to enable the start of transfers from the IBM MQ Explorer. Koordinasyon kuyruğu yöneticilesinde toplanan aracı bilgileri, aracı bilgilerini ve aracı durumunu görüntülemek için komutlar tarafından da kullanılır.

Aktarma durumu ve aktarma denetleme bilgileri eşgüdümleme kuyruk yöneticisininde yayınlanır. The transfer status and transfer audit information is used by the Managed File Transfer plug-in to monitor the

progress of transfers from the IBM MQ Explorer. Koordinasyon kuyruğu yöneticiliğine saklanan aktarma denetleme bilgileri denetime uygunluk sağlamak için saklanabilir.

Komut kuyruğu yöneticisi, IBM MQ ağına bağlanmak için kullanılır ve Managed File Transfer komutlarını verdiğinizde, kuyruk yöneticisidir.



İlgili kavramlar

“Managed File Transfer” sayfa 281
Managed File Transfer , dosya boyutuna ya da kullanılan işletim sistemlerine bakılmaksızın, yönetilen ve denetlenebilir bir şekilde sistemler arasında dosya aktarır.

“MFT , IBM MQile nasıl çalışır?” sayfa 284
Managed File Transfer , IBM MQile bir dizi şekilde etkileşimde bulunur.

İlgili başvurular

Managed File Transfer Senaryo

MFTREST API'e genel bakış

REST API , aktarımların listelenmesi de içinde olmak üzere bazı Managed File Transfer komutlarını ve dosya aktarma aracılarıyla ilgili ayrıntıları destekler.

IBM MQ 9.1.0' tan REST API , tüm geçerli Managed File Transfer aktarımlarını listelemek ve Managed File Transfer aracılarının durumunu sorgulamak için seçenekler içerir. Daha fazla bilgi için, bkz. REST API MFTile çalışmaya başlama.

IBM MQ Internet Pass-Thru

IBM MQ Internet Pass-Thru (MQIPT) is an optional component of IBM MQ that can be used to implement messaging solutions between remote sites across the internet.

V 9.2.0 From IBM MQ 9.2.0 , MQIPT is an optional component of IBM MQ. IBM MQ 9.2.xiçin MQIPT kuruluş dosyalarını edinmek için IBM Fix Central for IBM MQ' a gidin. Before IBM MQ 9.2.0, MQIPT was available as a support pack.

You do not have to be running IBM MQ 9.2.0 to use MQIPT in IBM MQ 9.2.0 or later. supported' un desteklenen herhangi bir sürümünü (IBM MQ) bağlamak için MQIPT olanağını kullanabilirsiniz; MQIPTile aynı sürüme başka bir IBM MQ bileşeni de kurmanız gerekmez.

IBM MQ yetkilendirmesi satın aldıysanız, MQIPT' un gerektiği kadar kopya kurabilirsiniz. MQIPT installations are not counted against your purchased IBM MQ entitlement. IBM MQ lisanslamasına ilişkin ek bilgi için IBM MQ lisans bilgileribaşlıklı konuya bakın.

Not: Bu belgeler, IBM MQ 9.2.0 ve sonraki yayın düzeylerindeki MQIPT ile ilgilidir. IBM Documentationiçindeki MQIPT destek paketi (sürüm 2.1) belgeleri için, IBM MQ 9.0 belgelerinde MQIPT (SupportPac MS81) başlıklı konuya bakın.

Not: If you are using MQIPT 2.1 or earlier, you are encouraged to upgrade to MQIPT for IBM MQ 9.2.x, as the end of support date for the MQIPT support pack is 30th September 2020.

IBM MQ Internet Pass-Thru , iki IBM MQ kuyruk yöneticisi arasında ya da bir IBM MQ istemcisi ile bir IBM MQ kuyruk yöneticisi arasında IBM MQ ileti akışlarını alabilen ve iletebilen bağımsız bir hizmet olarak çalışır.

MQIPT , istemci ve sunucu aynı fiziksel ağda yer almıyorsa bu bağlantıyı etkinleştirir.

MQIPT ile ilgili bir ya da daha çok örnek, iki IBM MQ kuyruk yöneticisi arasındaki iletişim yoluna ya da bir IBM MQ istemcisi ile bir IBM MQ kuyruk yöneticisi arasındaki iletişim yoluna yerleştirilebilir. MQIPT yönetim ortamları iki IBM MQ sisteminin, iki sistem arasında doğrudan TCP/IP bağlantısına gerek duymadan ileti değiş tokalmasına izin verir. Güvenlik duvarı yapılandırması, iki sistem arasında doğrudan bir TCP/IP bağlantısını engelliyorsa, bu olanak yararlı olur.

MQIPT listens on one or more TCP/IP ports for incoming connections, which can carry either normal IBM MQ messages, IBM MQ messages tunneled inside HTTP, or messages encrypted using Transport Layer Security (TLS) or Secure Sockets Layer (SSL). MQIPT birden çok eşzamanlı bağlantıyı işleyebilir.

İlk TCP/IP bağlantı isteğini yapan IBM MQ kanalı, *çağırıcı*olarak, *yanıtlayıcı*olarak bağlanmayı denediği kanal ve en sonunda *hedef kuyruk yöneticisi*olarak bağlantı kurmaya çalıştığı kuyruk yöneticisi olarak adlandırılır.

MQIPT , verileri kaynaktan hedefine iletirken belleğindeki verileri tutar. Diskteki veriler kaydedilmez (işletim sistemi tarafından diske sayfalanan bellek dışında). MQIPT ' un diske belirtik olarak eriştiği tek zaman, yapılandırma dosyasını okuması ve bağlantı günlüğü ve izleme kayıtlarını yazmasıdır.

The full range of IBM MQ channel types can be made through one or more instances of MQIPT. Bir iletişim yolunda MQIPT varlığı, bağlı IBM MQ bileşenlerinin işlevsel özelliklerini etkilemez, ancak ileti aktarımın başarımında bir etkisi olabilir.

MQIPT , “Olası MQIPTyapılandırmaları” sayfa 289içinde açıklandığı şekilde IBM MQ ve IBM Integration Busile birlikte kullanılabilir.

MQIPT'u kurmak için bkz. MQIPT' u kurma.

İlgili görevler

yapılandırmaIBM MQ Internet Pass-Thru

IBM MQ Internet Pass-Thru' in yönetilmesi ve yapılandırılması

İlgili başvurular

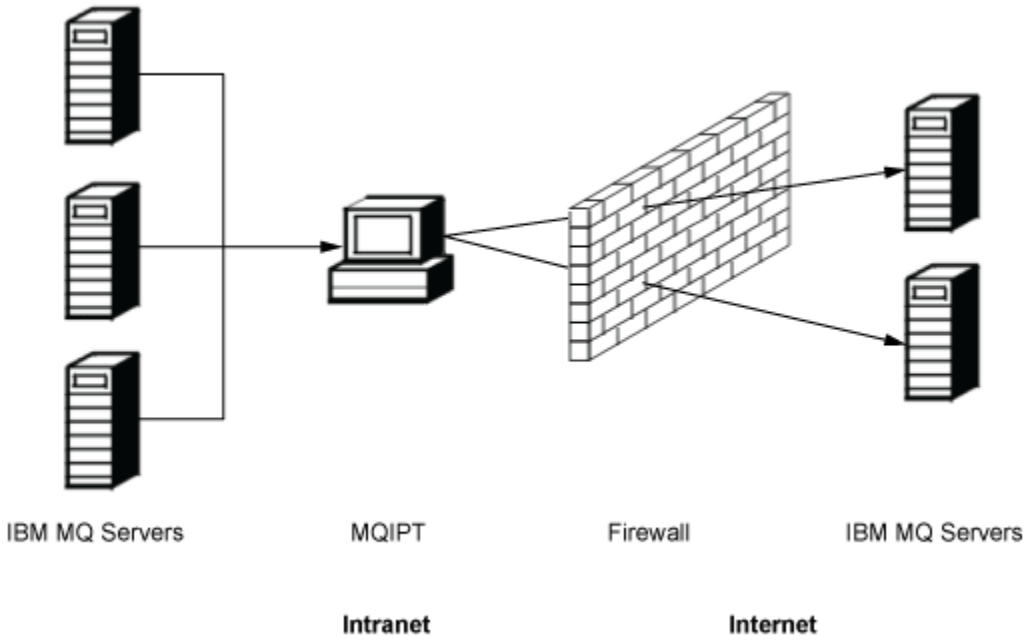
IBM MQ Internet Pass-Thru yapılandırma başvurusu

MQIPTkullanımı

There are a number of potential uses for IBM MQ Internet Pass-Thru (MQIPT).

MQIPT , kanal yoğunlaştırıcısı olarak kullanılabilir

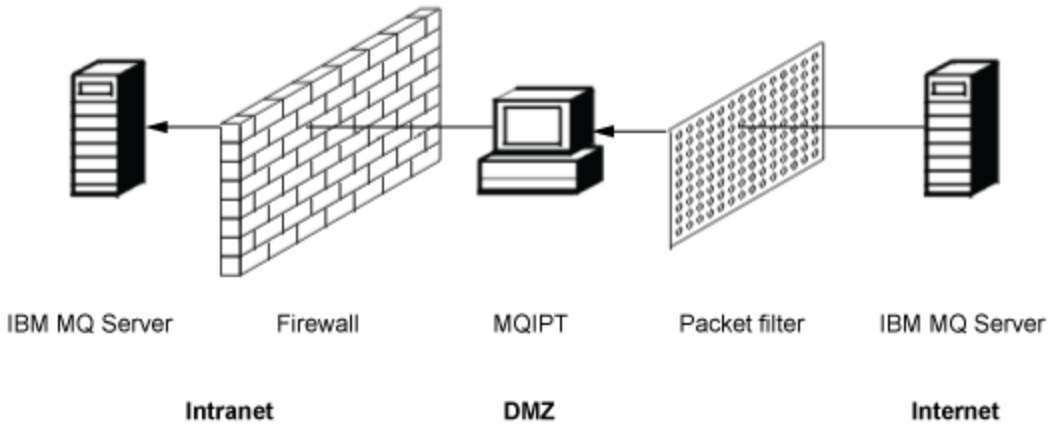
Bu şekilde MQIPT komutunu kullanarak, birden çok anasisteme ya da birden çok anasistemden kanal,allanasisteminden ya da tümü MQIPT anasisteminden geldikleri gibi bir güvenlik duvarına da görünebilirler. Bu, güvenlik duvarı süzme kurallarını tanımlamayı ve yönetmeyi kolaylaştırır.



Şekil 80. Kanal yoğunlaştırıcısı olarak MQIPT örneği

MQIPT , tek erişim noktası sağlamak için DMZ ' ye yerleştirilebilir

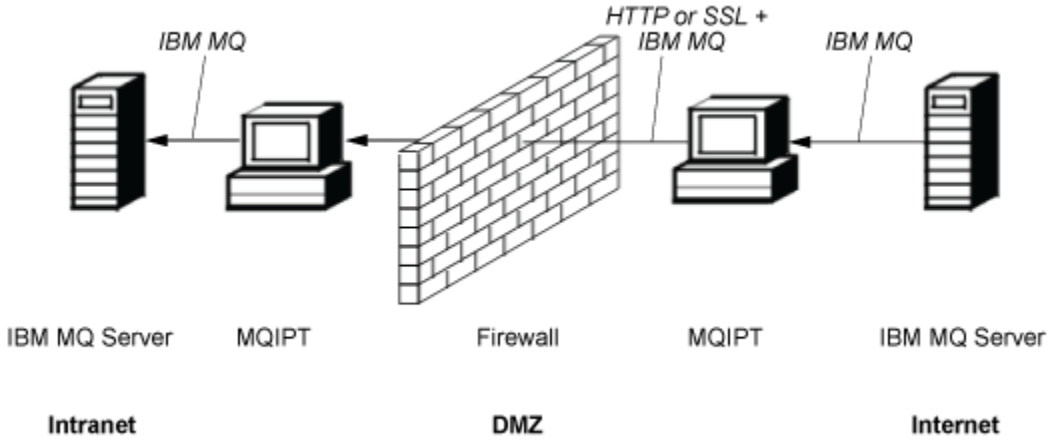
If MQIPT is placed within a DMZ firewall (a firewall configuration for securing local area networks), on a computer with a known and trusted internet protocol (IP) address, MQIPT can be used to listen for incoming IBM MQ channel connections which it can then forward to the trusted intranet; the inner firewall must allow this trusted computer to make inbound connections. Bu yapılandırmada, MQIPT , dış isteklerin güvenilir şirket içi ağdaki bilgisayarların gerçek IP adreslerini almasını önler. Bu şekilde MQIPT , tek bir erişim noktası sağlar.



Şekil 81. Bir DMZ güvenlik duvarındaki MQIPT örneği

MQIPT , HTTP tünelleme yoluyla iletişim kurabilir

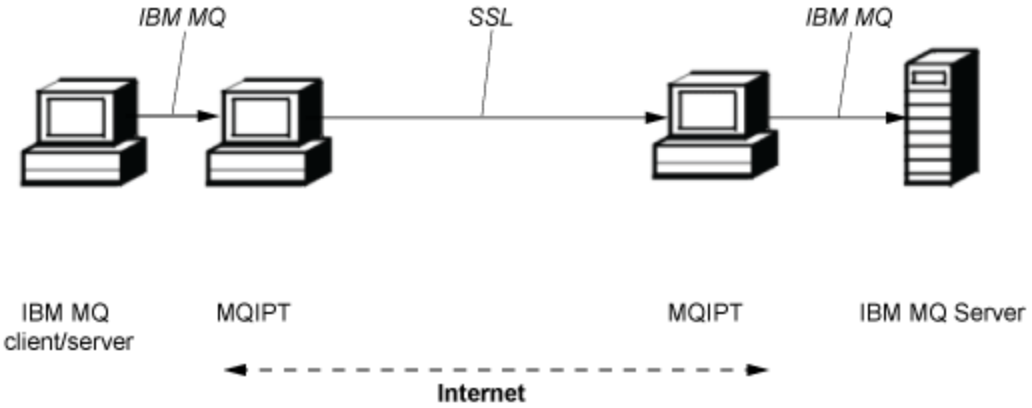
If two instances of MQIPT are deployed in line, they can communicate by using HTTP. HTTP tünelleme özelliği, var olan HTTP yetkili sunucularının kullanılmasıyla güvenlik duvarları yoluyla iletilebilmesini sağlar. İlk MQIPT , IBM MQ protokolünü HTTP ' ye ekler ve ikincisi, IBM MQ iletişim kuralını HTTP sarıcısından çıkarır ve hedef kuyruk yöneticisine iletir.



Şekil 82. MQIPT ve HTTP tünelleme örneği

MQIPT iletileri şifreleyebilir

MQIPT önceki örnekteki gibi yapılandırıldıysa, istekler güvenlik duvarlarından iletilmeden önce şifrelenebilir. İlk MQIPT verileri şifreler ve ikinci şifre, hedef kuyruk yöneticisine göndermeden önce SSL/TLS ' yi kullanarak şifreleri çözer.



Şekil 83. MQIPT ve SSL/TLS örneği

MQIPT nasıl çalışır

En basit yapılandırmasında, MQIPT bir IBM MQ iletişim kuralı ileticisi olarak işlev görür. Bir TCP/IP kapısını dinler ve IBM MQ kanallarından bağlantı isteklerini kabul eder.

Düzgün biçimlendirmiş bir istek alınırsa, MQIPT , kendisi ile hedef IBM MQ kuyruk yöneticisi arasında daha fazla TCP/IP bağlantısı kurar. Daha sonra, gelen bağlantısından hedef kuyruk yöneticisine aldığı tüm protokol paketlerini geçirir ve hedef kuyruk yöneticisinden özgün gelen bağlantıya geri dönüş protokolü paketlerini döndürür.

No change to the IBM MQ protocol (client/server or queue manager to queue manager) is involved because neither end is directly aware of the presence of the intermediary. IBM MQ istemcisi ya da sunucu kodunun yeni sürümleri gerekli değildir.

MQIPT' u kullanmak için, çağıran kanal, hedef kuyruk yöneticisinin anasistem adı ve kapısı değil, MQIPT anasistem adını ve kapısını kullanacak şekilde yapılandırılmalıdır. Bu, IBM MQ kanalının **CONNAME** özelliği ile tanımlanır. MQIPT , gelen verileri okur ve hedef kuyruk yöneticisine iletir. Bir istemci/sunucu kanalındaki kullanıcı kimliği ve parola gibi diğer yapılandırma alanları da hedef kuyruk yöneticisine aktarılır.

Birden çok kuyruk yöneticisi

MQIPT , birden çok hedef kuyruk yöneticisine erişilmesine izin vermek için kullanılabilir. Bunun çalışması için, MQIPT ' a hangi kuyruk yöneticisinin bağlanacağı, bu nedenle MQIPT gelen TCP/IP kapı numarasını kullanarak bağlanacağı kuyruk yöneticisini saptamak için bir mekanizma olmalıdır.

Bu nedenle, birden çok TCP/IP kapısını dinlemek için MQIPT konfigürasyonunu tanımlayabilirsiniz. Her bir dinleme kapısı, bir MQIPT *rotası*aracılığıyla bir hedef kuyruk yöneticiyle eşlenir. Dinleme TCP/IP kapısını hedef kuyruk yöneticisinin anasistem adı ve kapısıyla ilişkilendiren 100 'e kadar yol tanımlayabilirsiniz. Bu, hedef kuyruk yöneticisinin anasistem adının (IP adresi) kaynak kanal tarafından hiçbir zaman görülemeyeceği anlamına gelir. Her rota, dinleme kapısı ve hedefi arasındaki birden çok bağlantıyı, her bir bağlantı bağımsız olarak işlev görerek işleyebilir.

MQIPT yapılandırma dosyası

MQIPT , `mqipt.conf`adlı bir yapılandırma dosyasını kullanır. Bu dosya, tüm rotalara ve ilişkili özelliklere ilişkin tanımlamaları içerir. `mqipt.conf`ile ilgili daha fazla bilgi için bkz. IBM MQ Internet Pass-Thru' in yönetilmesi ve yapılandırılması .

MQIPT başlatıldığında, yapılandırma dosyasında listelenen her bir rota başlatılır. İletiler, her rotanın durumunu gösteren sistem konsoluna yazılır. Bir rota için MQCPI078 iletisi gösterildiğinde, bu rota bağlantı isteklerini kabul etmeye hazırdır.

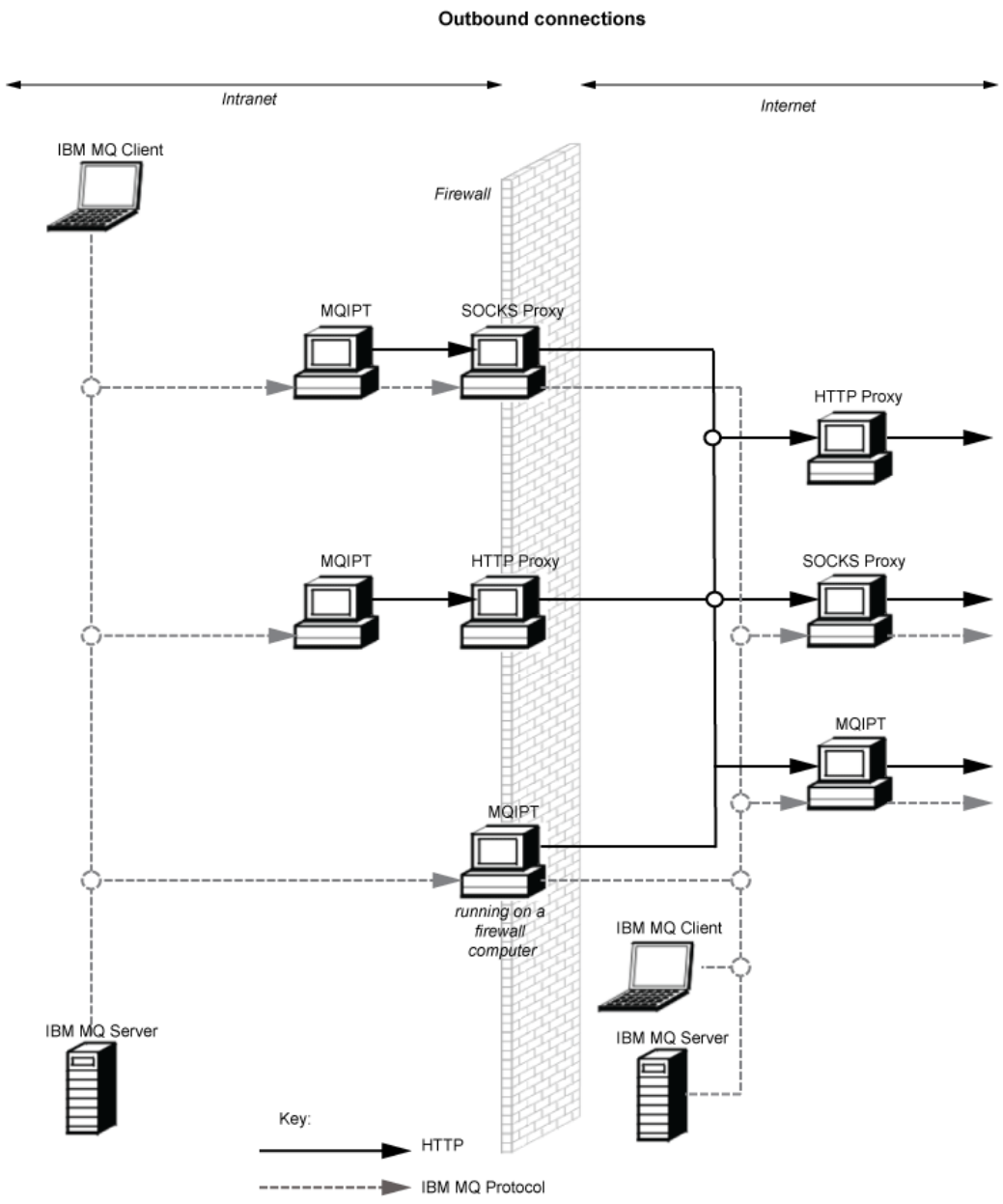
Olası MQIPTyapılandırmaları

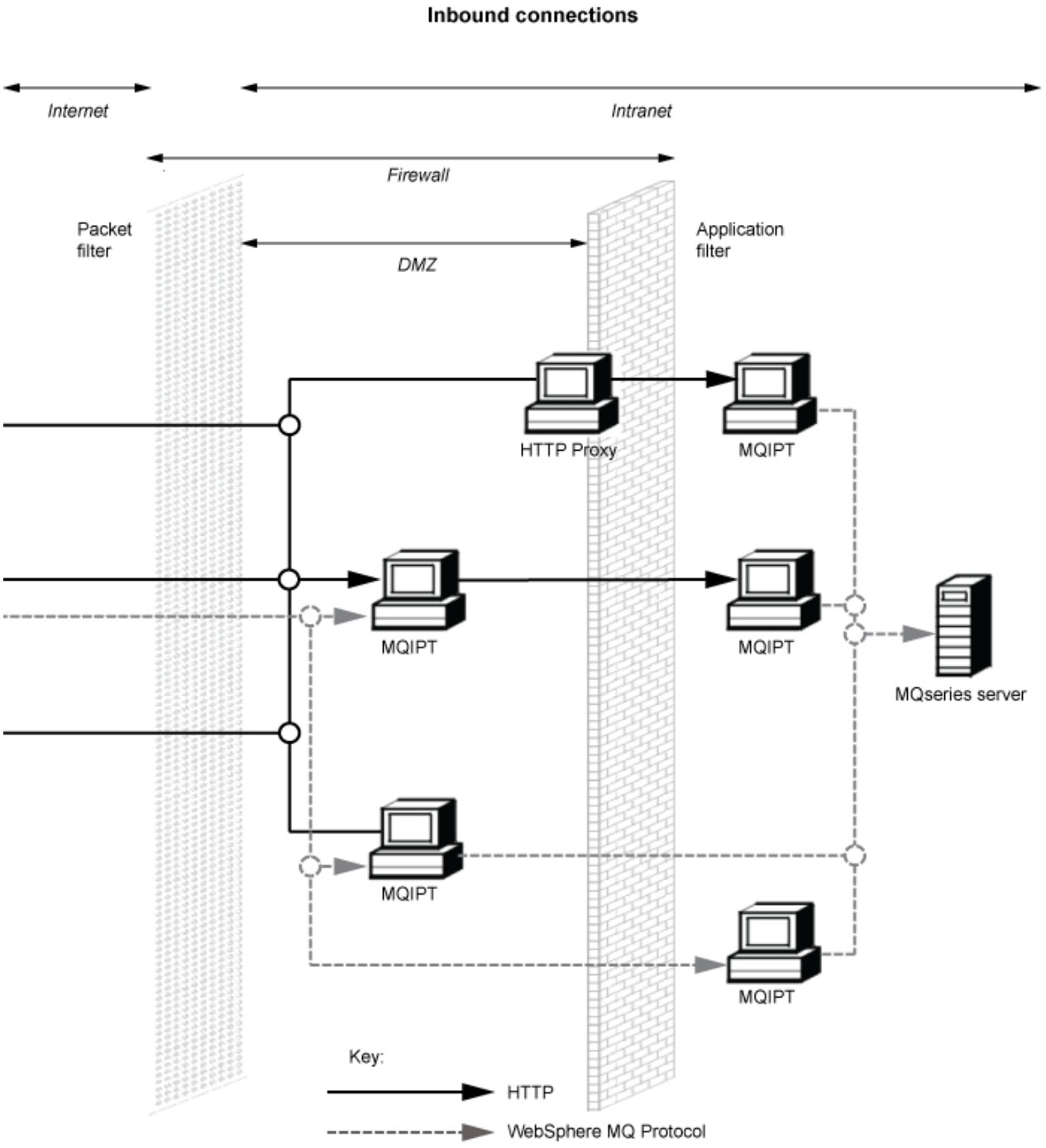
MQIPT , IBM MQ ve IBM Integration Busile bağlantılı olarak kullanılabilir.

Aşağıdaki çok parçalı şekil, IBM MQ topolojisinde MQIPT ' un olası yapılandırmalarının çoğunu gösterir. Bu, MQIPT ' in iletileri gönderebileceği farklı yollar gösterir. It shows clients and servers on an intranet, inside a firewall, and on the Internet outside the firewall, passing messages to MQIPT, HTTP proxy, or SOCKS proxy, which forwards them.

The messages are received by an MQIPT proxy or an HTTP proxy in a DMZ before passing the message through the inbound firewall to a server.

Güvenlik duvarının şirket içi ağ tarafındaki HTTP yetkili sunucusu, SOCKS yetkili sunucusu ve MQIPT bilgisayarlarının birden çok bilgisayarın İnternet 'te birbirine zincirleme olasılığını temsil ettiğini unutmayın. Örneğin, bir MQIPT bilgisayarı, hedefine ulaşmadan önce bir ya da daha fazla SOCKS ya da HTTP yetkili bilgisayar ya da daha fazla MQIPT bilgisayarı aracılığıyla iletişim kurabilir.


Outbound connections
Intranet
Internet
IBM MQ Client
Firewall
MQIPT
SOCKS Proxy
HTTP Proxy
MQIPT
HTTP Proxy
SOCKS Proxy
MQIPT
MQIPT
running on a firewall computer
IBM MQ Client
IBM MQ Server
IBM MQ Server
Key:
HTTP
IBM MQ Protocol



Uyumlu yapılandırmalar

Bir IBM MQ istemcisi ya da kuyruk yöneticisinin MQIPTile iletişim kurduğu uyumlu bağlantı senaryoları. Aynı ya da ikinci bir MQIPT rotası, hedef kuyruk yöneticisiyle iletişim kurmak için kullanılır.

Tek bir MQIPT rotasına sahip uyumlu yapılandırmalar

IBM MQile iletişim kurmak için tek bir MQIPT rotasını kullanabilirsiniz.

Çizelge 25 sayfa 292 içindeki sütunlar aşağıdaki bilgileri içerir:

1. IBM MQ ile MQIPT rotası arasında kullanılan iletişim kuralı. The connection can be created by either an IBM MQ client or queue manager, and can use either IBM MQ Formats and Protocols (FAP) or a SSL/TLS protocol.
2. MQIPT rotasının çalışacağı kip. The format of the communication across the Internet between MQIPT and IBM MQ, is determined by the configuration of the MQIPT route. Tablonun SSL 'den söz ettirdiğine dikkat edin, TLS' yi de kullanabilirsiniz.
3. MQIPT rotası ile hedef kuyruk yöneticisi arasında kullanılan iletişim kuralı.

Çizelge 25. Tek bir MQIPTyönetim ortamı ile geçerli yapılandırmalar

1. IBM MQ kaynak iletişim kuralı	**2. MQIPT rotasının kipi**	**3. IBM MQ hedef iletişim kuralı**
YOK	FAP-yetkili sunucu (varsayılan)	YOK
	FAP-sunucu ve SSL-istemci	SSL/TLS
SSL/TLS	SSL yetkili sunucusu	SSL/TLS
	SSL-sunucu ve FAP-client	YOK
	SSL sunucusu ve SSL istemcisi	SSL/TLS

Birden çok MQIPT rotasına sahip uyumlu yapılandırmalar

IBM MQile iletişim kurmak için MQIPT' in bir ya da daha çok örneğinde birden çok rota kullanmayı seçebilirsiniz.

Çizelge 26 sayfa 293 içindeki sütunlar aşağıdaki bilgileri içerir:

1. IBM MQ ile ilk MQIPT rotası arasında kullanılan iletişim kuralı. The connection can be created by either an IBM MQ client or queue manager, and can use either IBM MQ Formats and Protocols (FAP) or a SSL/TLS protocol.
2. İlk MQIPT rotasının çalıştığı kip. The format of the communication across the Internet between MQIPT and IBM MQ, is determined by the configuration of the MQIPT route. Tablonun SSL 'den söz ettirdiğine dikkat edin, TLS' yi de kullanabilirsiniz.
3. İkinci MQIPT rotasının çalışacağı kip.
4. İkinci MQIPT rotası ile hedef kuyruk yöneticisi arasında kullanılan iletişim kuralı.

Çizelge 26. Birden çok MQIPTeşgörünümü içeren geçerli yapılandırmalar			
1. IBM MQ kaynak iletişim kuralı	**2. İlk MQIPT rotasının kipi**	**3. İkinci MQIPT rotasının kipi**	**4. IBM MQ hedef iletişim kuralı**
FAP (varsayılan)	FAP-yetkili sunucu (varsayılan)	FAP-yetkili sunucu (varsayılan)	YOK
	FAP-sunucu ve SSL-istemci	SSL yetkili sunucusu	SSL/TLS
		SSL-sunucu ve FAP-client	YOK
		SSL sunucusu ve SSL istemcisi	SSL/TLS
	HTTP istemcisi	HTTP sunucusu ve SSL istemcisi	SSL/TLS
	HTTPS-istemci	HTTPS-server ve SSL-client	SSL/TLS
	HTTP istemcisi	HTTP sunucusu	YOK
	HTTPS-istemci	HTTPS-sunucu	YOK
SSL/TLS	SSL yetkili sunucusu	SSL yetkili sunucusu	SSL/TLS
		SSL-sunucu ve FAP-client	YOK
		SSL sunucusu ve SSL istemcisi	SSL/TLS
	HTTP istemcisi	HTTP sunucusu	YOK
	HTTPS-istemci	HTTPS-sunucu	SSL/TLS
	HTTP istemcisi	HTTP sunucusu ve SSL istemcisi	YOK
	HTTPS-istemci	HTTPS-server ve SSL-client	SSL/TLS

Desteklenen kanal yapılandırmaları

Tüm IBM MQ kanal tipleri desteklenir, ancak yapılandırma TCP/IP bağlantılarıyla sınırlandırılmıştır. Bir IBM MQ istemcisi ya da kuyruk yöneticisine, MQIPT hedef kuyruk yöneticisinmiş gibi görünür. Kanal konfigürasyonunun bir hedef anasistem ve kapı numarası gerektirdiği durumlarda, MQIPT anasistem adı ve dinleyici kapı numarası belirtilir.

İstemci/sunucu kanalları

MQIPT , gelen istemci bağlantısı isteklerini dinler ve daha sonra, HTTP tünelleme, SSL/TLS ya da standart IBM MQ iletişim kuralı paketleri kullanarak bunları iletir. MQIPT , HTTP tünelleme ya da SSL/TLS kullanıyorsa, bunları ikinci bir MQIPTile bağlantıda iletir. HTTP tünellemesi kullanmıyorsa, hedef kuyruk yöneticisi olarak görebileceği bir bağlantıya iletir (bununla birlikte, daha ileri bir MQIPTolma durumu olabilir). Hedef kuyruk yöneticisi istemci bağlantısını kabul ettiğinde, paketler istemci ile sunucu arasında aktarılır.

Küme gönderen/alıcı kanalları

MQIPT , bir küme gönderen kanalından gelen isteği alırsa, kuyruk yöneticisinin SOCKS etkin durumda olduğunu ve SOCKS el sıkışılması işlemi sırasında doğru hedef adresinin alınacağını varsayar. İsteği sonraki MQIPT ya da hedef kuyruk yöneticisine, istemci bağlantı kanallarıyla aynı şekilde iletir. Bu, otomatik olarak tanımlanmış küme gönderici kanallarını da içerir.

Gönderen/alıcı
MQIPT , bir gönderen kanalından gelen bir istek alırsa, bunu bir sonraki MQIPT ya da hedef kuyruk yöneticisine, istemci bağlantı kanallarıyla tam olarak aynı şekilde iletir. Hedef kuyruk yöneticisi, gelen isteği doğrular ve uygunsa, alıcı kanalını başlatır. Gönderen ve alıcı kanalı (güvenlik akışları da içinde olmak üzere) arasındaki tüm iletişimler aktarılır.

İsteyen/sunucu
Bu birleşim, önceki yapılandırmalarla aynı şekilde ele alınır. Bağlantı isteğini doğrulama, hedef kuyruk yöneticisinde sunucu kanalı tarafından gerçekleştirilir.

İsteyen/gönderen
İki kuyruk yöneticisinin birbiriyle doğrudan bağlantı kurmasına izin verilmiyorsa, ancak her ikisinin de MQIPT ' a bağlanmasına ve bu yapılandırmadan bağlantıları kabul etmesine izin verilmiyorsa, "geri bildirme" yapılandırması kullanılabilir.

Sunucu/istek sahibi ve sunucu/alıcı
Bunlar MQIPT ile aynı şekilde ele alınır ve `Sender/Receiver` yapılandırmasını işleyerek gerçekleştirir.

Kanal sonlandırma ve arıza koşulları

MQIPT , bir IBM MQ kanalının kapatılmasını (olağan ya da olağan dışı) algıladığında, kanal kapanımı yayılır. MQIPTkomutunu kullanarak bir rotayı kapadıysanız, bu rotanın tüm kanalları kapatılır.

MQIPT , isteğe bağlı bir boşta durma zamanaşımı olanağı sağlar. MQIPT , bir kanalın zamanaşımını aşan bir süre boyunca boşta olduğunu saptarsa, söz konusu iki bağlantıda hemen sona erme işlemi gerçekleştirir.

Kanalın her iki ucundaki IBM MQ sistemleri, bu olağandışı sona erdirme koşullarını ağ hataları olarak ya da kanalın iş ortağı tarafından sona erdirilmesi olarak gözlemlemektedir. Daha sonra kanal, MQIPT kullanılmadığı gibi, yeniden başlatma ve kurtarma işlemini (örneğin, bir protokol belirsiz bir iletişim kuralı sırasında gerçekleşirse) kurtarır.

İletilerin güvenliği

IBM MQ dağıtılmış kuyruk yönetimi, iletilerin doğru bir şekilde teslim edilmesini sağlar. This is still the case when MQIPT is present between the two ends of the channel. MQIPT , ileti verilerini saklamaz ya da ileti tesliminin doğru olmasını sağlayan eşitleme noktası yordamında yer almaz.

Hızlı, kalıcı olmayan IBM MQ iletileri kullanırken, MQIPT yönlendirmesi başarısız olursa ya da bir IBM MQ iletisi geçişken yeniden başlatılırsa, ileti kaybolabilir. Rotayı yeniden başlatmadan önce MQIPT rotasını kullanan tüm IBM MQ kanallarının devre dışı olduğundan emin olun.

IBM MQiçindeki iletilerin güvenliği hakkında daha fazla bilgi için bkz. İletilerin güvenliği.

Çok eşgörünümli kuyruk yöneticileri ve yüksek kullanılabilirlik

MQIPT , yüksek kullanılabilirlik ortamlarındaki çok eşgörünümli kuyruk yöneticileriyle kullanılabilir.

MQIPT 'in kalıcı durumu yoktur ve bu nedenle MQIPT ' un başka bir sisteme geçmesinde yarar yoktur. Bunun yerine, farklı sistemlerde çalışan aynı `mqipt.conf` yapılanış kütüğüyle birden çok MQIPT yönetim ortamı vardır. Her MQIPT yönetim ortamını kullanılabilirlik için izleyin ve gerekirse yeniden başlatın (aynı sistemde). Bu, bağlantıları yönlendirmek için kullanılabilecek özdeş MQIPT yönetim ortamları kümesi sağlar. Bundan sonra IBM MQ 'in bağlantıları MQIPT ile yönlendirebileceğini ve MQIPT ' in bu bağlantıları hedef kuyruk yöneticisine iletebileceğini doğrulamalısınız.

Giden IBM MQ kanalları, çeşitli şekillerde kullanılabilir bir MQIPT yönetim ortamına yönlendirilebilir, örneğin:

- Use a load balancer or high availability router, such as the IBM Network Dispatcher from the WebSphere Edge Components product.

- Virgülle ayrılmış bir liste kullanarak, IBM MQ kanalı tanımlamasında birden çok bağlantı adı belirtin. IBM MQ daha sonra, kullanılabilir bir MQIPT yönetim ortamı buluncaya kadar sırayla her bir MQIPT adresine bağlanmayı dener.

You must also direct connections from MQIPT to the destination queue manager. Yüksek kullanılabilirlik yapılandırması, IP adresinin hedef kuyruk yöneticisiyle birlikte başarısız olmasını güvenceye alır, daha sonra özel bir MQIPT yapılandırması gerekmez: **Destination** rota özelliğinde hedef IP adresini belirtin ve hata durumunda yedek sisteme geçiş işleminin IP adresini kuyruk yöneticisiyle birlikte hareket ettirmesini sağlayın.

However, if the IP address of the queue manager changes after a failover then you must arrange for MQIPT to forward the connection to the correct destination. Bu işlem birkaç yoldan birinde yapılabilir:

- Hangi IP adresinin ve kapı numarasının erişilebilir olduğunu kontrol eden bir yöneltme çıkışı yazın ve sonra her bağlantı için rota hedefini geçersiz kılın. Bazı örnek yöneltme çıkışları MQIPTile birlikte verilir; bu çıkışlar bu amaçla uyarlanabilir.
- Bağlantıyı yeniden yönlendirmek için yüksek kullanılabilirlikli yük dengeleyici kullanın.
- Her bir IP adresi ve kapı için kuyruk yöneticisinin çalıştırılabileceği birden çok MQIPT rotası tanımlayın. Then direct the IBM MQ connections to the various MQIPT routes, for example by listing all of the route IP addresses and port numbers in a comma-separated list in the connection name of the outbound channel.

Ağ yolundaki uçtan uca bileşenlerin tümü için ayar yapmak da önemlidir:

1. Kullanılamayan sistemlerle bağlantı kurma girişimleri, yeniden bağlanma girişimlerinin kullanılabilir ilk hedefe taşınabilmesi için hemen başarısız olmalıdır.

   MQIPT SSL rotaları için, kullanılamayan hedefler için bilgi istemi bağlantısı hatasının sağlanması için **SSLClientConnectTimeout** rota özelliğini ayarlayın. IBM MQ ayarlama parametrelerinin ayrıntıları için IBM MQ belgelerine bakın. Ayrıca, işletim sistemine ilişkin TCP/IP ayarına ilişkin ayrıntılar için işletim sistemi belgelerinize bakın. Tüm durumlarda, başarısız bağlantı denemeleri bir ağ hatasını (örneğin, bir TCP ilk duruma getirme paketi) hızla geri döndürmelidir ya da süresi dolmadan zamanaşımına neden olmalıdır.
2. Yeni bağlantıların kurulabilmesi için, hatalı bir sisteme ilişkin etkin bağlantıların hemen kesilmesi gerekir.

   You should also consider the impact of a failover at a time when connections are actively using MQIPT. Hata durumunda yedek sisteme geçiş işlemi sırasında ağ bağlantılarının kopması olasıdır. İstemci uygulamaları için, kopmuş bağlantıları yeniden kurmak için IBM MQ otomatik istemci yeniden bağlantı özelliğini kullanabilirsiniz. İleti kanalları için, kanal yeniden bağlantı kurmasını sağlamak için kısa bir yeniden deneme aralığı belirtebilirsiniz. Otomatik istemci yeniden bağlantısı ve ileti kanalı yeniden deneme yapılandırması hakkında daha fazla bilgi için IBM MQ belgelerine bakın.

Özel notlar

Bu belge, ABD'de kullanıma sunulan ürünler ve hizmetler için hazırlanmıştır.

IBM, bu belgede sözü edilen ürün, hizmet ya da özellikleri diğer ülkelerde kullanıma sunmayabilir. Bulunduğunuz yerde kullanıma sunulan ürün ve hizmetleri yerel IBM müşteri temsilcisinden ya da çözüm ortağından öğrenebilirsiniz. Bir IBM ürün, program ya da hizmetine gönderme yapılması, açık ya da örtük olarak, yalnızca o IBM ürünü, programı ya da hizmetinin kullanılabileceğini göstermez. Aynı işlevi gören ve IBM'in fikri mülkiyet haklarına zarar vermeyen herhangi bir ürün, program ya da hizmet de kullanılabilir. Ancak, IBM dışı ürün, program ya da hizmetlerle gerçekleştirilen işlemlerin değerlendirilmesi ve doğrulanması kullanıcının sorumluluğundadır.

IBM'in, bu belgedeki konularla ilgili patentleri ya da patent başvuruları olabilir. Bu belgenin size verilmiş olması, patentlerin izinsiz kullanım hakkının da verildiği anlamına gelmez. Lisansla ilgili sorularınızı aşağıdaki adrese yazabilirsiniz:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Çift byte (DBCS) bilgilerle ilgili lisans soruları için, ülkenizdeki IBM'in Fikri Haklar (Intellectual Property) bölümüyle bağlantı kurun ya da sorularınızı aşağıda adrese yazın:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japonya

Aşağıdaki paragraf, İngiltere ya da bu tür hükümlerin yerel yasalarla uyuşmadığı diğer ülkelerde geçerli değildir: INTERNATIONAL BUSINESS MACHINES CORPORATION BU YAYINI, HAK İHLALİ YAPILMAYACAĞINA DAİR GARANTİLERLE TİCARİLİK VEYA BELİRLİ BİR AMACA UYGUNLUK İÇİN ZIMNİ GARANTİLER DE DAHİL OLMAK VE FAKS BUNLARLA SINIRLI OLMAMAK ÜZERE AÇIK YA DA ZIMNİ HİÇBİR GARANTİ VERMEKSİZİN "OLDUĞU GİBİ" ESASIYLA SAĞLAMAKTADIR. Bazı ülkeler bazı işlemlerde garantinin açık ya da örtük olarak reddedilmesine izin vermez; dolayısıyla, bu bildirim sizin için geçerli olmayabilir.

Bu yayın teknik yanlışlar ya da yazım hataları içerebilir. Buradaki bilgiler üzerinde düzenli olarak değişiklik yapılmaktadır; söz konusu değişiklikler sonraki basımlara yansıtılacaktır. IBM, önceden bildirimde bulunmaksızın, bu yayında açıklanan ürünler ve/ya da programlar üzerinde iyileştirmeler ve/ya da değişiklikler yapabilir.

Bu belgede IBM dışı Web sitelerine yapılan göndermeler kullanıcıya kolaylık sağlamak içindir ve bu Web sitelerinin onaylanması anlamına gelmez. Bu Web sitelerinin içerdiği malzeme, bu IBM ürününe ilişkin malzemenin bir parçası değildir ve bu tür Web sitelerinin kullanılmasının sorumluluğu size aittir.

IBM'e bilgi ilettiğinizde, IBM bu bilgileri size karşı hiçbir yükümlülük almaksızın uygun gördüğü yöntemlerle kullanabilir ya da dağıtabilir.

(i) Bağımsız olarak yaratılan programlarla, bu program da içinde olmak üzere diğer programlar arasında bilgi değiş tokuşuna ve (ii) değiş tokuş edilen bilginin karşılıklı kullanımına olanak sağlamak amacıyla bu program hakkında bilgi sahibi olmak isteyen lisans sahipleri şu adrese yazabilirler:

IBM Corporation
Yazılım Birlikte Çalışabilirlik Koordinatörü, Bölüm 49XA
3605 Highway 52 N

Rochester, MN 55901
U.S.A.

Bu tür bilgiler, ilgili kayıt ve koşullar altında ve bazı durumlarda bedelli olarak edinilebilir.

Bu belgede açıklanan lisanslı program ve bu programla birlikte kullanılabilecek tüm lisanslı malzeme, IBM tarafından, IBM Müşteri Sözleşmesi, IBM Uluslararası Program Lisansı Sözleşmesi ya da eşdeğer herhangi bir sözleşmenin kayıt ve koşulları altında sağlanır.

Burada belirtilen performans verileri denetimli bir ortamda elde edilmiştir. Bu nedenle, başka işletim ortamlarında çok farklı sonuçlar alınabilir. Bazı ölçümler geliştirilme düzeyindeki sistemlerde yapılmıştır ve bu ölçümlerin genel kullanıma sunulan sistemlerde de aynı olacağı garanti edilemez. Ayrıca, bazı sonuçlar öngörü yöntemiyle elde edilmiş olabilir. Dolayısıyla, gerçek sonuçlar farklı olabilir. Bu belgenin kullanıcıları, kendi ortamları için geçerli verileri kendileri doğrulamalıdırlar.

IBM dışı ürünlerle ilgili bilgiler, bu ürünleri sağlayan firmalardan, bu firmaların yayın ve belgelerinden ve genel kullanıma açık diğer kaynaklardan alınmıştır. IBM bu ürünleri sınamamıştır ve IBM dışı ürünlerle ilgili performans doğruluğu, uyumluluk gibi iddiaları doğrulayamaz. IBM dışı ürünlerin yeteneklerine ilişkin sorular, bu ürünleri sağlayan firmalara yöneltilmelidir.

IBM'in gelecekteki yönelim ve kararlarına ilişkin tüm bildirimler değişebilir ve herhangi bir duyuruda bulunulmadan bunlardan vazgeçilebilir; bu yönelim ve kararlar yalnızca amaç ve hedefleri gösterir.

Bu belge, günlük iş ortamında kullanılan veri ve raporlara ilişkin örnekler içerir. Örneklerin olabildiğince açıklayıcı olması amacıyla kişi, şirket, marka ve ürün adları belirtilmiş olabilir. Bu adların tümü gerçek dışıdır ve gerçek iş ortamında kullanılan ad ve adreslerle olabilecek herhangi bir benzerlik tümüyle rastlantıdır.

YAYIN HAKKI LİSANSI:

Bu belge, çeşitli işletim platformlarında programlama tekniklerini gösteren, kaynak dilde yazılmış örnek uygulama programları içerir. Bu örnek programları, IBM'e herhangi bir ödemede bulunmadan, örnek programların yazıldığı işletim altyapısına ilişkin uygulama programlama arabirimiyle uyumlu uygulama programlarının geliştirilmesi, kullanılması, pazarlanması ya da dağıtılması amacıyla herhangi bir biçimde kopyalayabilir, değiştirebilir ve dağıtabilirsiniz. Bu örnekler her koşul altında tüm ayrıntılarıyla sınanmamıştır. Dolayısıyla, IBM bu programların güvenilirliği, bakım yapılabilirliği ya da işlevleri konusunda açık ya da örtük güvence veremez.

Bu bilgileri elektronik kopya olarak görüntülediyseniz, fotoğraflar ve renkli resimler görünmeyebilir.

Programlama arabirimi bilgileri

Programlama arabirimi bilgileri (sağlandıysa), bu programla birlikte kullanılmak üzere uygulama yazılımları yaratmanıza yardımcı olmak üzere hazırlanmıştır.

Bu kitap, müşterinin WebSphere MQhizmetlerini edinmek üzere program yazmasına olanak tanıyan, amaçlanan programlama arabirimlerine ilişkin bilgiler içerir.

Ancak, bu bilgiler tanılama, değiştirme ve ayarlama bilgilerini de içerebilir. Tanılama, değiştirme ve ayarlama bilgileri, uygulama yazılımlarınızda hata ayıklamanıza yardımcı olur.

Önemli: Bu tanılama, değiştirme ve ayarlama bilgilerini bir programlama arabirimi olarak kullanmayın; bu, değişiklik söz konusu olduğunda kullanılır.

Ticari Markalar

IBM, IBM logosu, ibm.com, IBM Corporation 'ın dünya çapında birçok farklı hukuk düzeninde kayıtlı bulunan ticari markalarıdır. IBM ticari markalarının güncel bir listesini Web üzerinde "Telif hakkı ve ticari marka bilgileri"www.ibm.com/legal/copytrade.shtmladresinde bulabilirsiniz. Diğer ürün ve hizmet adları IBM'in veya diğer şirketlerin ticari markaları olabilir.

Microsoft ve Windows, Microsoft Corporation'ın ABD ve/veya diğer ülkelerdeki ticari markalarıdır.

UNIX, The Open Group şirketinin ABD ve diğer ülkelerdeki tescilli ticari markasıdır.

Linux, Linus Torvalds'ın ABD ve/ya da diğer ülkelerdeki tescilli ticari markasıdır.

Bu ürün, Eclipse Project (https://www.eclipse.org/) tarafından geliştirilen yazılımları içerir.

Java ve Java tabanlı tüm markalar ve logolar, Oracle firmasının ve/ya da iştiraklerinin markaları ya da tescilli markalarıdır.

Parça numarası:

(1P) P/N: