9.3

IBM MQ -Teknik genel bakış

Not

Bu bilgileri ve desteklediği ürünü kullanmadan önce, “Özel notlar” sayfa 295 bölümündeki bilgileri okuyun.

Bu basım, yeni basımlarda tersi belirtilmedikçe, IBM® MQ sürüm 9 yayın düzeyi 3 ve sonraki tüm yayınlar ve değişiklikler için geçerlidir.

IBM'e bilgi gönderdiğinizde, IBM ' e bu bilgileri size hiçbir sorumluluk yüklemeden uygun gördüğü yöntemlerle kullanması ya da dağıtması için münhasır olmayan bir hak verirsiniz.

© Copyright International Business Machines Corporation 2007, 2024.

İçindekiler

Teknik genel bakış..... 5
İleti kuyruklama girişine giriş.....5
İleti kuyruğuna almanın başlıca özellikleri ve avantajları.....7
İleti kuyruklama terminolojisi..... 9
İletiler ve kuyruklar..... 12
IBM MQ nesneler..... 14
Nesne tipleri..... 15
IBM MQ nesnelerini adlandırma..... 35
Dağıtılmış kuyruğa alma ve kümeler..... 41
Dağıtılmış kuyruğa alma bileşenleri.....45
Küme bileşenleri.....54
İleti alışverişi yayınla/abone ol..... 59
Bileşenleri yayınla/abone ol.....60
Tek bir kuyruk yöneticisi yayınlama/abone olma yapılandırması örneği..... 84
Dağıtılmış yayınlama/abone olma ağları.....85
IBM MQ Çok hedefli.....101
İlk çoklu yayın kavramları..... 101
MQ Telemetry'e genel bakış..... 102
MQ Telemetry ' a Giriş..... 104
Telemetri kullanım senaryoları..... 105
Telemetri aygıtlarının kuyruk yöneticisine bağlanması.....111
Telemetri bağlantı iletişim kuralları.....112
Telemetry (MQXR) hizmeti.....112
Telemetri kanalları..... 112
IBM MQ Telemetry Transport iletişim kuralı..... 113
MQTT müşterileri.....113
MQTT istemcisine ileti gönderilmesi..... 114
MQTT istemcisinden IBM MQ uygulamasına ileti gönderilmesi..... 122
MQTT yayınlama/abone olma uygulamaları.....123
Telemetri uygulamaları..... 124
MQ Telemetry ' in kuyruk yöneticileriyle bütünleştirilmesi.....124
MQTT durumsuz ve durumlu oturumlar..... 127
Bir MQTT istemcisi bağlı olmadığında..... 127
MQTT istemcileri ve IBM MQ uygulamaları arasındaki gevşek bağlaşım..... 128
MQ Telemetry güvenliği..... 128
MQ Telemetry küreselleşme.....129
MQ Telemetry performansı ve ölçeklenebilirliği..... 129
MQ Telemetry tarafından desteklenen aygıtlar.....131
IBM MQ içinde güvenlik.....132
IBM MQ.NET yönetilen istemci TLS desteği..... 133
IBM MQ MQI clients.....134
Neden IBM MQ istemcileri kullanılıyor?..... 136
Genişletilmiş işlem istemcisi nedir?..... 137
İstemcinin sunucuya bağlanma şekli..... 138
İşlem yönetimi ve desteği..... 140
Kuyruk yöneticisi olanaklarını genişletme..... 141
IBM MQ Java dil arabirimleri.....142
IBM MQ classes for JMS/Jakarta Messaging..... 143
IBM MQ ileti alışverişi sağlayıcısı.....153
IBM MQ for z/OS kavramlar.....154
z/OS üzerindeki kuyruk yöneticisi.....155
z/OS üzerindeki kanal başlatıcısı..... 156

IBM MQ for z/OS yönetimi için koşullar ve görevler....158
Paylaşılan kuyruklar ve kuyruk paylaşım grupları....160
Grup içi kuyruğa alma....204
z/OS üzerinde depolama yönetimi....217
oturum açmaIBM MQ for z/OS....221
z/OS üzerinde sistem tanımı....232
z/OS üzerinde kurtarma ve yeniden başlatma....242
IBM MQ for z/OS içindeki güvenlik kavramları....258
z/OS üzerinde kullanılabilirlik....265
IBM MQ for z/OS üzerinde izleme ve istatistik....268
z/OS üzerinde kurtarma birimi yok etme....269
IBM MQ ve diğer z/OS ürünleri....272
IBM MQ ve CICS....272
IBM MQ ve IMS....273
IBM MQ ve z/OS Toplu İş, TSO ve RRS bağdaştırıcıları....277
IBM MQ for z/OS ve WebSphere Application Server....278
Managed File Transfer....279
MFT IBM MQile nasıl çalışır?....281
MFT topolojiye genel bakış....282
MFTREST API'e genel bakış....283
IBM MQ Internet Pass-Thru....283
MQIPT kullanımı....284
MQIPT nasıl çalışır....286
Olası MQIPT yapılandırmaları....287
Uyumlu yapılandırmalar....289
Desteklenen kanal yapılandırmaları....291
Kanal sonlandırma ve arıza koşulları....292
İletilerin güvenliği....292
Çok eşgörünümlü kuyruk yöneticileri ve yüksek kullanılabilirlik....292
IBM MQ Console ve REST API....293

Özel notlar....295
Programlama arabirimi bilgileri....296
Ticari Markalar....296

IBM MQ Teknik genel bakış

Uygulamalarınızı bağlamak ve bilgilerin kuruluşunuz genelindeki dağıtımını yönetmek için IBM MQ kullanın.

IBM MQ , programların tutarlı bir uygulama programlama arabirimi kullanarak farklı bileşenlerden (işlemciler, işletim sistemleri, altsistemler ve iletişim protokolleri) oluşan bir ağ üzerinden birbirleriyle iletişim kurmalarını sağlar. Bu arabirim kullanılarak tasarlanan ve yazılan uygulamalar, ileti kuyruklama uygulamaları olarak bilinir.

IBM MQtarafından sağlanan ileti kuyruklama ve diğer özellikler hakkında bilgi edinmek için aşağıdaki alt konuları kullanın.

İlgili kavramlar

IBM MQ ' a Giriş

Ürün gereksinimlerinin ve destek bilgilerinin bulunduğu yer

İlgili görevler

IBM MQ mimarisinin planlanması

İlgili başvurular

“İleti kuyruğuna almanın başlıca özellikleri ve avantajları” sayfa 7

Bu bilgiler, ileti kuyruğuna almanın bazı özelliklerini ve yararlarını vurgular. İleti kuyruklama güvenlik ve veri bütünlüğü gibi özellikleri açıklar.

İleti kuyruklama girişine giriş

IBM MQ ürünleri, programların tutarlı bir uygulama programlama arabirimi kullanarak farklı bileşenlerden (işlemciler, işletim sistemleri, altsistemler ve iletişim protokolleri) oluşan bir ağ üzerinden birbirleriyle iletişim kurmalarını sağlar.

Bu arabirim kullanılarak tasarlanan ve yazılan uygulamalar, *ileti alışverişi* ve *kuyruklama* biçemini kullandıkları için *ileti kuyruklama* uygulamaları olarak bilinir:

- İleti alışverişi, programların birbirlerini doğrudan çağırmak yerine iletilerde birbirlerine veri göndererek iletişim kurabilecekleri anlamına gelir.
- Kuyruğa alma, iletilerin depolamadaki kuyruklara yerleştirilmesi, programların birbirinden bağımsız olarak, farklı hızlarda ve saatlerde, farklı konumlarda ve aralarında mantıksal bir bağlantı olmadan çalışmasına izin verilmesi anlamına gelir.

İleti kuyruklama, uzun yıllar veri işlemede kullanıldı. Günümüzde elektronik postada yaygın olarak kullanılmaktadır. Kuyruğa alma olmadan, uzun mesafeler üzerinden elektronik bir ileti göndermek, rotadaki her düğümün iletilerin iletilmesi için kullanılabilir olmasını ve adreslerin oturum açmasını ve onlara bir mesaj göndermeye çalıştığın gerçeğinin bilincinde olmasını gerektirir. Kuyruğa alma sisteminde iletiler, sistem bunları iletmeye hazır oluncaya kadar ara düğümlerde saklanır. Son varış yerlerinde, alıcı onları okumaya hazır olana kadar elektronik posta kutusunda depolanır.

Yine de, bugün birçok karmaşık iş işlemi kuyruğa alınmadan işlenir. Büyük bir ağda, sistem kullanıma hazır durumda binlerce bağlantıyı sürdürüyor olabilir. Sistemin bir parçası bir sorun yaşarsa, sistemin birçok bölümü kullanılamaz duruma gelir.

İletilerin kuyruğa alınmasının programlar için elektronik posta olduğunu düşünebilirsiniz. İleti kuyruğa alma ortamında, bir uygulama grubunun bir parçasını oluşturan her program, belirli bir isteğe yanıt olarak iyi tanımlanmış, bağımsız bir işlev gerçekleştirir. Başka bir programla iletişim kurmak için, bir programın önceden tanımlanmış bir kuyruğa bir ileti koyması gerekir. Diğer program iletiyi kuyruktan alır ve iletide bulunan istekleri ve bilgileri işler. Yani mesaj kuyruklama, programdan programa iletişimin bir stilidir.

Kuyruğa alma, bir uygulama bunları işlemeye hazır oluncaya kadar iletilerin tutulduğu düzenektir. Kuyruğa alma, aşağıdakileri yapmanızı sağlar:

© Copyright IBM Corp. 2007, 2024

- İletişim kodunu yazmak zorunda kalmadan programlar arasında iletişim kurun (her biri farklı ortamlarda çalışıyor olabilir).
- Bir programın iletileri işleme sırasını seçin.
- İleti sayısı bir eşiği aştığında kuyruğa hizmet vermek üzere birden çok program ayarlayarak sistemdeki yükleri dengeleyebilirsiniz.
- Birincil sisteminiz kullanılamıyorsa, kuyruklara hizmet vermek için alternatif bir sistem ayarlayarak uygulamalarınızın kullanılabilirliğini artırın.

İleti kuyruğu nedir?

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

Kuyruklar bir kuyruk yöneticisinde bulunur ve bu yönetici tarafından yönetilir (bkz. "İleti kuyruklama terminolojisi" sayfa 9). Bir kuyruğun fiziksel yapısı, kuyruk yöneticisinin çalıştığı işletim sistemine bağlıdır. Kuyruk, bir bilgisayarın belleğindeki geçici bir arabellek alanı ya da kalıcı bir depolama aygıtındaki (disk gibi) bir veri kümesi olabilir. Kuyrukların fiziksel yönetimi kuyruk yöneticisinin sorumluluğundadır ve katılan uygulama programları tarafından anlaşılmaz.

Programlar kuyruklara yalnızca kuyruk yöneticisinin dış hizmetleri aracılığıyla erişir. Bir kuyruğu açabilir, üzerine ileti koyabilir, ondan ileti alabilir ve kuyruğu kapatabilirler. Ayrıca, kuyrukların özniteliklerini ayarlayabilir ve sorgulayabilirler.

Farklı ileti kuyruklama biçemleri

Noktadan noktaya

Kuyruğa bir ileti yerleştirilir ve bir uygulama bu iletiyi alır.

Noktadan noktaya iletişim iletisinde, gönderen uygulamanın o uygulamaya ileti gönderebilmesi için önce alan uygulamayla ilgili bilgileri bilmesi gerekir. Örneğin, gönderen uygulamanın, bilgilerin gönderileceği kuyruğun adını bilmesi gerekebilir ve bir kuyruk yöneticisi adı da belirtebilir.

Yayınla/Abone Ol

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

Yayınlama/abone olma ileti sistemi, bilgi sağlayıcısını bu bilgilerin kullanıcılarından ayıklamanızı sağlar. Gönderilen ve alınan bilgilerin gönderilmesi ve alınması için gönderen uygulamanın birbirleri hakkında çok şey bilmesine gerek yoktur. Daha fazla bilgi için "İleti alışverişi yayınla/abone ol" sayfa 59 başlıklı konuya bakın.

Uygulama tasarımcısına ve geliştiriciye ileti kuyruğuna alınmasının avantajları

IBM MQ , uygulama programlarının ileti odaklı işlemeye katılmak için *ileti kuyruklama* özelliğini kullanmasını sağlar. Uygulama programları, uygun ileti kuyruklama yazılım ürünlerini kullanarak farklı platformlar arasında iletişim kurabilir. Örneğin, z/OS uygulamaları IBM MQ for z/OSaracılığıyla iletişim kurabilir. Uygulamalar, altta yatan iletişimin mekaniğinden korunmaktadır. İleti kuyruğuna almanın diğer yararlarından bazıları şunlardır:

- Birçok uygulama arasında paylaşabileceğiniz küçük programları kullanarak uygulamalar tasarlayabilirsiniz.
- Bu yapı taşlarını yeniden kullanarak hızla yeni uygulamalar oluşturabilirsiniz.
- İleti kuyruklama tekniklerini kullanmak üzere yazılan uygulamalar, kuyruk yöneticilerinin çalışma biçimindeki değişikliklerden etkilenmez.
- Herhangi bir iletişim protokolünü kullanmanız gerekmez. Kuyruk yöneticisi, sizin için iletişimin tüm yönleriyle ilgilenir.

- İletileri alan programların, iletilerin gönderildiği sırada çalıştırılması gerekmez. İletiler kuyruklarda saklanır.

Tasarımcılar, geliştirme daha hızlı, daha az geliştirici gerektiğinden ve programlama becerisine ilişkin talepler, ileti kuyruklama özelliğini kullanmayan uygulamalardan daha düşük olduğundan, uygulamalarının maliyetini azaltabilir.

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

MQI ile ilgili ayrıntılar için İleti Kuyruğu Arabirimi 'ne genel bakışbaşlıklı konuya bakın.

İleti kuyruğuna almanın başlıca özellikleri ve avantajları

Bu bilgiler, ileti kuyruğuna almanın bazı özelliklerini ve yararlarını vurgular. İleti kuyruklama güvenlik ve veri bütünlüğü gibi özellikleri açıklar.

İleti kuyruklama tekniklerini kullanan uygulamaların temel özellikleri şunlardır:

- “Programlar arasında doğrudan bağlantı yok” sayfa 7
- “Zamandan bağımsız iletişim” sayfa 8
- “Küçük programlar” sayfa 8
- “İletiyle yönlendirilen işleme” sayfa 8
- “Olaya dayalı işleme” sayfa 8
- “İleti önceliği” sayfa 9
- “Durumu” sayfa 9
- “Veri bütünlüğü” sayfa 9
- “Kurtarma desteği” sayfa 9

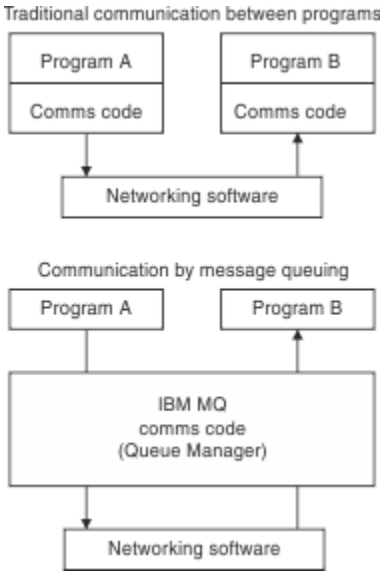
Not: IBM MQ istemcileri ve sunucuları göz önünde bulundurulduğunda, yeni altyapılarda ek IBM MQ MQI clients destekleyecek şekilde bir sunucu uygulamasını değiştirmeniz gerekmez. Benzer şekilde, IBM MQ MQI client ek sunucu tipleriyle işlev görür.

Programlar arasında doğrudan bağlantı yok

İleti kuyruklama, dolaylı programdan programa iletişim için bir tekniktir. Bu, programların birbiriyle iletişim kurduğu herhangi bir uygulamada kullanılabilir. İletişim, iletileri bir kuyruğa (bir kuyruk yöneticisinin iyeliğinde) ve iletileri kuyruktan alan başka bir program tarafından gerçekleştirilir.

Programlar, diğer programlar tarafından kuyruğa konan iletileri alabilir. Diğer programlar, alan programla aynı kuyruk yöneticisine ya da başka bir kuyruk yöneticisine bağlanabilir. Bu diğer kuyruk yöneticisi başka bir sistemde, farklı bir bilgisayar sisteminde ya da farklı bir işletmenin ya da işletmenin içinde olabilir.

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



Şekil 1. Geleneksel iletişimle karşılaştırıldığında ileti kuyruğa alma

Elektronik postada olduğu gibi, bir işlemin parçası olan iletiler, bir mağaza ve iletme ağındaki bir ağ üzerinden seyahat eder.temel olarak. Düğümler arasındaki bir bağlantı başarısız olursa, bağlantı geri yükleninceye ya da işletmen ya da program iletiyi yeniden yönlendirinceye kadar ileti alıkonur.

Bir iletinin kuyruktan kuyruğa geçmesini sağlayan mekanizma programlardan gizlenir. Bu nedenle programlar daha basittir.

Zamandan bağımsız iletişim

Başkalarının iş yapmasını isteyen programlar, isteğin yanıtını beklemek zorunda değildir. Başka bir iş yapabilir ve yanıt geldiğinde ya da daha sonra işlenebilir. Bir ileti sistemi uygulaması yazarken, bir programın ne zaman ileti gönderdiğini ya da hedefin iletiyi ne zaman alabildiğini bilmenize gerek yoktur. İleti kaybolmaz; hedef iletiyi işlemeye hazır oluncaya kadar kuyruk yöneticisi tarafından alıkonur. İleti, bir program tarafından kaldırılıncaya kadar kuyrukta kalır. Bu, gönderen ve alan uygulama programlarının ayrılması anlamına gelir; gönderen, alıcının iletiyi aldığını kabul etmesini beklemeden işlemeye devam edebilir. İleti gönderildiğinde hedef uygulamanın çalışması bile gerekmez. Başlatıldıktan sonra iletiyi alabilir.

Küçük programlar

İleti kuyruklama, küçük, bağımsız programlar kullanmanın avantajlarını kullanmanızı sağlar. Bir işin tüm bölümlerini sıralı olarak gerçekleştiren tek, büyük bir program yerine, işi birkaç küçük, bağımsız programa dağıtabilirsiniz. İstekte bulunan program, ayrı programların her birine kendi işlevlerini gerçekleştirmelerini isteyen iletiler gönderir; her program tamamlandığında, sonuçlar bir ya da daha çok ileti olarak geri gönderilir.

İletiyle yönlendirilen işleme

Bir kuyruğa gelen iletiler, *triggering*komutunu kullanarak otomatik olarak bir uygulamayı başlatabilir. Gerekiyorsa, ileti (ya da iletiler) işlendiğinde uygulamalar durdurulabilir.

Olaya dayalı işleme

Programlar, kuyrukların durumuna göre denetlenebilir. Örneğin, bir ileti kuyruğa gelir gelmez programın başlamasını sağlayabilir ya da örneğin, kuyruktaki belirli bir önceliğin üzerinde 10 ileti ya da kuyruktaki herhangi bir önceliğe sahip 10 ileti bulununcaya kadar programın başlamamasını belirleyebilirsiniz.

İleti önceliği

Bir program, iletiyi kuyruğa koyduğunda iletiye öncelik atayabilir. Bu, yeni iletinin ekleneceği kuyruktaki konumu belirler.

Programlar, kuyruktaki iletileri, iletilerin kuyruktaki sırasına göre ya da belirli bir iletiyi alarak alabilir. (Bir program, daha önce gönderdiği bir isteğe yanıt arıyorsa, belirli bir iletiyi almak isteyebilir.)

Durumu

Bir kuyruk yöneticisini kullandıklarında uygulamaların kimlik doğrulaması, kuyruk yöneticisinde kuyruk gibi kaynakları kullandıklarında yetkilendirme denetimleri ve ağ üzerinde dolaşırken ileti verilerinin şifrelenmesi ve kuyruklarda bulunmaları da dahil olmak üzere güvenlik olanakları sağlanır. Güvenlikle ilgili daha fazla bilgi için bkz. Güvenliğe Genel Bakış.

Veri bütünlüğü

Veri bütünlüğü, iş birimleri tarafından sağlanır. İş biriminin başlangıç ve bitiş birimlerinin uyumlulaştırılması, her MQGET ya da MQPUT için bir seçenek olarak tam olarak desteklenir ve iş biriminin sonuçlarının kesinleştirilmesine ya da geriye işlenmesine olanak sağlar. Sync Point Support, uygulama için seçilen eşitleme noktası koordinasyonu biçimine bağlı olarak IBM MQ ' e dahili ya da harici olarak çalışır.

Kurtarma desteği

Kurtarmanın mümkün olması için tüm kalıcı IBM MQ güncellemeleri günlüğe kaydedilir. Kurtarma gerekirse, tüm kalıcı iletiler geri yüklenir, tüm devam eden hareketler geriye işlenir ve tutarlılık noktası kesinleştirme ve geriletme işlemleri, denetimdeki Sync Point Manager 'in olağan yöntemiyle işlenir. Kalıcı iletiler hakkında daha fazla bilgi için bkz. İleti kalıcılığı.

İleti kuyruklama terminolojisi

Bu bilgiler, ileti kuyruklama sırasında kullanılan bazı terimlere ilişkin bir öngörü sağlar.

Bunlar şunları içerir:

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

Kanallar

Kanallar, iletileri bir kuyruk yöneticisinden diğerine taşımak için kullanılır ve uygulamaları temel iletişim protokollerinden korur. Kuyruk yöneticileri aynı sistemde ya da aynı platformda ya da farklı platformlarda var olabilir. Gönderilen iletiler birçok yerden kaynaklanabiliyor:

- Verileri bir düğümden diğerine aktaran kullanıcı tarafından yazılan uygulama programları.
- PCF komutlarını ya da MQAI ' yi kullanan kullanıcı tarafından yazılan denetim uygulamaları.
- IBM MQ Explorer.
- Başka bir kuyruk yöneticisine özel işlemden geçirme olayı iletileri gönderen kuyruk yöneticileri.
- Uzak denetim komutlarını başka bir kuyruk yöneticisine gönderen kuyruk yöneticileri. Örneğin, MQSC komutları ya da administrative REST APIkullanılarak.

Kanallar hakkında daha fazla bilgi için bkz. “Kanal tanımları” sayfa 31.

Küme

Küme , mantıksal olarak ilişkili kuyruk yöneticilerinden oluşan bir ağdır.

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

Kümelerin kullanılmasının iki farklı nedeni vardır: sistem yönetimini azaltmak ve kullanılabilirliği ve iş yükü dengelemesini geliştirmek.

En küçük kümeyi bile kurarken, basitleştirilmiş sistem yönetiminden yararlanıyorsunuz. Bir kümenin parçası olan kuyruk yöneticileri için daha az tanımlama gerekir; bu nedenle, tanımlamalarınızda hata yapma riski azalır.

Kümeleme hakkında daha fazla bilgi için bkz. Kümeler.

IBM MQ MQI client

IBM MQ MQI *istemcileri* , IBM MQ' in bağımsız olarak kurulabilir bileşenleridir. MQI istemcisi, IBM MQ uygulamalarını bir iletişim protokolüyle çalıştırmanızı, diğer altyapılarda bir ya da daha çok MQI (Message Queue Interface; İleti Kuyruğu Arabirimi) sunucusuyla etkileşimde bulunmanızı ve kuyruk yöneticilerine bağlanmanızı sağlar.

IBM MQ MQI client bileşenlerinin nasıl kurulacağına ve kullanılacağına ilişkin tüm ayrıntılar için aşağıdaki konulara bakın:

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

ve Sunucu ile istemci arasındaki bağlantıların yapılandırılması.

Grup içi kuyruğa alma

z/OS

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

Grup içi kuyruğa alma hakkında daha fazla bilgi için bkz. “Grup içi kuyruğa alma” sayfa 204.

İleti

İleti kuyruklanmasında, ileti, bir program tarafından gönderilen ve başka bir program için tasarlanmış bir veri toplamıdır. Bkz. IBM MQ iletileri.

İleti tipleriyle ilgili bilgi için İleti tipleribaşlıklı konuya bakın.

İleti kanalı aracısı

Bir ileti kanalı aracısı, bir kanalın bir ucudur. Bir çift ileti kanalı aracısı, biri gönderen, biri alan, bir kanal oluşturup iletileri bir kuyruk yöneticisinden diğerine taşır.

İleti kanalı aracılarının nasıl kullanıldığına ilişkin bilgi için Dağıtılmış kuyruk yönetimine girişbaşlıklı konuya bakın.

İleti tanımlayıcı

IBM MQ iletisi, denetim bilgileri ve uygulama verilerinden oluşur.

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

- İletinin tipi
- İletinin tanıtıcısı
- İletinin teslimine ilişkin öncelik

Uygulama verilerinin yapısı ve içeriği, IBM MQtarafından değil, katılan programlar tarafından belirlenir.

Daha fazla bilgi için bkz. MQMD.

Noktadan noktaya ileti sistemi

Noktadan noktaya iletide, her ileti bir üreten uygulamadan tek bir tüketen uygulamaya gider. İletiler, iletileri kuyruğa koyan üretici uygulama aracılığıyla aktarılır ve kullanan uygulama bunları o kuyruktan alır.

İleti alışverişi yayınla/abone ol

Yayınlama/abone olma ileti sisteminde, bir yayınlama uygulaması tarafından yayınlanan her iletinin bir kopyası ilgili her uygulamaya teslim edilir. İlgilenen çok sayıda uygulama olabilir, bir ya da hiç olmayabilir. Yayınlama/abone olma durumunda, ilgili bir uygulama abone olarak bilinir ve iletiler bir abonelik tarafından tanımlanan bir kuyrukta kuyruğa alınır.

Daha fazla bilgi için bkz “İleti alışverişi yayınla/abone ol” sayfa 59.

Kuyruk

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

Daha fazla bilgi için bkz “Kuyruklar” sayfa 19.

Kuyruk yöneticisi

Kuyruk yöneticisi , uygulamalara kuyruk hizmetleri sağlayan bir sistem programıdır.

Programların kuyruklara ileti koyabilmesi ve kuyruklardan ileti alabilmesi için bir uygulama programlama arabirimi sağlar. Kuyruk yöneticisi, denetimcilerin yeni kuyruklar yaratması, varolan kuyrukların özelliklerini değiştirmesi ve kuyruk yöneticisinin işlemini denetleyebilmesi için ek işlevler sağlar.

IBM MQ ileti kuyruklama hizmetlerinin bir sistemde kullanılabilmesi için çalışmakta olan bir kuyruk yöneticisi olmalıdır. Tek bir sistemde çalışan birden çok kuyruk yöneticiniz olabilir (örneğin, bir test sistemini *canlı* sistemden ayırmak için). Bir uygulamada, her kuyruk yöneticisi bir *bağlantı tanıtıcısı* (*Hconn*) ile tanımlanır.

Birçok farklı uygulama aynı anda kuyruk yöneticisinin hizmetlerini kullanabilir ve bu uygulamalar tamamen ilgisiz olabilir. Bir programın bir kuyruk yöneticisinin hizmetlerini kullanabilmesi için, o kuyruk yöneticisiyle bağlantı kurması gerekir.

Uygulamaların diğer kuyruk yöneticilerine bağlı uygulamalara ileti gönderebilmesi için, kuyruk yöneticilerinin kendi aralarında iletişim kurabilmeleri gerekir. IBM MQ , bu tür uygulamalar arasında iletilerin güvenli bir şekilde teslim edilmesini sağlamak için bir *saklama ve iletme* iletişim kuralı uygular.

Daha fazla bilgi için bkz "Kuyruk yöneticileri" sayfa 27.

Kuyruk paylaşım grubu

z/OS

Aynı paylaşılan kuyruk kümesine erişebilen kuyruk yöneticileri, *kuyruk paylaşım grubu* (QSG) adı verilen bir grup oluşturur. Paylaşılan kuyrukları saklayan bir bağlaşım olanağıyla (CF) iletişim kurarlar. Bu yalnızca IBM MQ for z/OSiçin geçerlidir.

Daha fazla bilgi için bkz "Paylaşılan kuyruklar ve kuyruk paylaşım grupları" sayfa 160.

Paylaşılan kuyruk

z/OS

Paylaşılan kuyruk , sistem şebekesinde bulunan bir ya da daha çok kuyruk yöneticisi tarafından erişilebilen iletilerin bulunduğu bir yerel kuyruk tipidir. Bu, aynı kuyruk yöneticisi kullanılarak birden çok uygulama tarafından paylaşılan bir kuyrukla aynı değildir. Bu yalnızca IBM MQ for z/OSiçin geçerlidir.

Abonelik

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

Abonelik, abonenin kimliğine ve yayınların yerleştirileceği hedef kuyruğun kimliğine ilişkin bilgileri içerir. Ayrıca, bir yayının hedef kuyruğa nasıl yerleştirileceğine ilişkin bilgileri de içerir.

Daha fazla bilgi için bkz "Aboneler ve abonelikler" sayfa 62.

Konu

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

Konular, bir yayınlama/abone olma sistemindeki iletilerin başarılı bir şekilde teslim edilmesinin anahtarıdır. Her iletiye belirli bir hedef adresi dahil etmek yerine, bir yayınlayıcı her iletiye bir konu atar. Kuyruk yöneticisi, konuyu, o konuya abone olan abonelerin listesiyle eşleştirir ve iletiyi bu abonelerin her birine teslim eder.

Daha fazla bilgi için bkz "Konular" sayfa 65.

İletiler ve kuyruklar

İletiler ve kuyruklar, ileti kuyruklama sisteminin temel bileşenleridir.

Mesaj nedir?

İleti , onu kullanan uygulamalar için anlamlı olan bir bayt dizgisidir. İletiler, bir uygulama programından diğerine (ya da aynı uygulamanın farklı bölümleri arasında) bilgi aktarmak için kullanılır. Uygulamalar aynı platformda ya da farklı platformlarda çalışıyor olabilir.

IBM MQ iletisi şunlardan oluşmuş:

- *Uygulama verileri.*Uygulama verilerinin içeriği ve yapısı, bunu kullanan uygulama programları tarafından tanımlanır.

- *İleti tanımlayıcı.*İleti tanımlayıcı iletiyi tanıtır ve ileti tipi ve gönderen uygulama tarafından iletiye atanan öncelik gibi ek denetim bilgilerini içerir.

  İleti tanımlayıcısının biçimi IBM MQtarafından tanımlanır. İleti tanımlayıcısının tam açıklaması için MQMD-Message descriptorbaşlıklı konuya bakın.
- *İleti özellikleri.* İletiyle ilgili meta veriler. İleti özelliklerinin içeriği, bunları kullanan uygulama programları tarafından tanımlanır. Daha fazla bilgi için bkz. İleti özellikleri.

İleti uzunlukları

Varsayılan ileti uzunluğu üst sınırı 4 MB 'dir, ancak bunu 100 MB ' lik bir uzunluğa (burada 1 MB, 1 048 576 bayta eşittir) kadar artırabilirsiniz. Uygulamada, ileti uzunluğu aşağıdakilerle sınırlı olabilir:

- Alma kuyruğu için tanımlanan ileti uzunluğu üst sınırı
- Kuyruk yöneticisi için tanımlanan ileti uzunluğu üst sınırı
- Kuyruk tarafından tanımlanan ileti uzunluğu üst sınırı
- Gönderen ya da alan uygulama tarafından tanımlanan ileti uzunluğu üst sınırı
- İleti için kullanılabilir saklama alanı miktarı

Bir uygulamanın gerektirdiği tüm bilgileri göndermek için birkaç ileti gerekebilir.

Uygulamalar nasıl ileti gönderir ve alır?

Uygulama programları, **MQI çağrılarını**kullanarak ileti gönderir ve alır.

Örneğin, bir iletiyi 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 çağrılarıyla ilgili daha fazla bilgi için bkz. MQI çağrıları.

Kuyruk nedir?

Kuyruk , iletileri saklamak için kullanılan bir veri yapısıdır.

Her kuyruk bir *kuyruk yöneticisine*aittir. Kuyruk yöneticisi, sahip olduğu kuyrukların korunmasından ve aldığı tüm iletilerin uygun kuyruklarda saklanmasından sorumludur. İletiler, uygulama programları ya da olağan işleminin bir parçası olarak kuyruk yöneticisi tarafından kuyruğa konabilir.

Önceden tanımlanmış kuyruklar ve dinamik kuyruklar

Kuyruklar, yaratılma biçimleriyle tanımlanabilir:

- **Önceden tanımlanmış kuyruklar** , bir denetimci tarafından uygun MQSC ya da PCF komutlarını kullanarak yaratılır.Önceden tanımlanmış kuyruklar kalıcıdır; bunları kullanan ve hayatta kalan uygulamalardan bağımsız olarak IBM MQ yeniden başlatılır.
- **Dinamik kuyruklar** , bir uygulama yaratıldığında oluşturulur Bir *model kuyruğunun*adını belirten bir MQOPEN isteği yayınlar. Yaratılan kuyruk, model kuyruğu adı verilen bir *şablon kuyruğu tanımlamasına*dayalıdır.DEFINE QMODEL MQSC komutunu kullanarak bir model kuyruğu yaratabilirsiniz. Model kuyruğunun öznitelikleri (örneğin, üzerinde saklanabilecek ileti sayısı üst sınırı), kuyruktan yaratılan herhangi bir dinamik kuyruk tarafından devralınır.

  Model kuyruklarının, dinamik kuyruğun kalıcı mı, yoksa geçici mi olacağını belirten bir özniteliği vardır. Kalıcı kuyruklar uygulamayı ve kuyruk yöneticisini yeniden başlatır; yeniden başlatma sırasında geçici kuyruklar kaybolur.

Kuyruklardan ileti alınıyor

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

- İlk giren ilk çıkar (FIFO).
- İleti tanımlayıcıda tanımlandığı şekilde, ileti önceliği. Aynı önceliğe sahip iletiler FIFO temelinde alınır.
- Belirli bir ileti için program isteği.

Uygulamadan gelen MQGET isteği, kullanılan yöntemi belirler.

IBM MQ nesneler

Kuyruk yöneticileri IBM MQ nesnelerinin özelliklerini tanımlar. Bu özelliklerin değerleri, IBM MQ ' in bu nesneleri işleme şeklini etkiler. IBM MQ komutlarını ve arabirimlerini kullanarak nesneleri oluşturabilir ve yönetebilirsiniz. Uygulamalarınızda, nesneleri denetlemek için İleti Kuyruğu Arabirimi 'ni (MQI) kullanırsınız. Nesneler, bir programdan adreslendiğinde bir IBM MQ *nesne tanımlayıcısı* (MQOD) tarafından tanımlanır.

Nesne yönetimi

Nesnelerin yönetimi aşağıdaki görevleri içerir:

- Kuyruk yöneticileri başlatılıyor ve durduruluyor.
- Uygulamalar için özellikle kuyruklar olmak üzere nesneler oluşturma.
- Nesnelerin özniteliklerini görüntüleme ya da değiştirme.
- Nesneler siliniyor.
- Diğer (uzak) sistemlerde kuyruk yöneticilerine iletişim yolları oluşturmak için kanallarla çalışma.
- Genel yönetim sürecini basitleştirmek ve iş yükünü dengelemek için kuyruk yöneticilerinin *kümelerini* oluşturma.

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

Bir nesne yönetimi işlemini gerçekleştirmek için IBM MQ komutunu kullandığınızda, kuyruk yöneticisi işlemi gerçekleştirmek için gereken yetki düzeyine sahip olup olmadığınızı denetler. Benzer şekilde, bir uygulama bir nesneyi açmak için MQOPEN çağrısı kullandığında, kuyruk yöneticisi, uygulamanın o nesneye erişime izin vermeden önce gerekli yetki düzeyine sahip olup olmadığını denetler. Denetimler, açılmakta olan nesnenin adına yapılır.

Aşağıdaki yöntemleri kullanarak nesneleri tanımlayabilir ve yönetebilirsiniz:

- Programlanabilir komut biçimleri başvurusu ve Denetim görevlerinin otomatikleştirilmesi başlıklı kısımlarda açıklanan PCF komutları
- MQSC komutları içinde açıklanan MQSC komutları
- z/OS IBM MQ for z/OS işlemleri ve denetim panoları, Yönetme IBM MQ for z/OS başlıklı konuda açıklanmıştır.
- Linux Windows IBM MQ Explorer (Yalnızca Intel sistemleri için Windows ve Linux). Daha fazla bilgi için bkz. Introduction to MQ Explorer.

Nesneleri aşağıdaki yöntemleri kullanarak da yönetebilirsiniz:

- Klavyeden yazılan denetim komutları. Bkz. Denetim komutlarını kullanarak IBM MQ for Multiplatforms yönetme.
- IBM MQ Yönetim Arabirimi (MQAI) bir programdaki çağrıları. Bkz. IBM MQ Yönetim Arabirimi (MQAI).

ALW AIX, Linux, and Windowsüzerindeki IBM MQ komutlarının sıraları için, MQSC olanağını kullanarak bir dosyada tutulan bir dizi komutu çalıştırabilirsiniz. Daha fazla bilgi için bakınız: Administering IBM MQ using MQSC commands.

IBM i Düzenli olarak kullandığınız IBM MQ for IBM i komutlarının sıraları için CL programlarını yazabilirsiniz. Daha fazla bilgi için bkz. Denetim dili (CL) komutlarını kullanarak IBM MQ for IBM i yönetme.

z/OS Düzenli olarak kullandığınız IBM MQ for z/OS komutları sıraları için, komutları içeren iletiler yaratan ve bu iletileri sistem komutu giriş kuyruğuna koyan denetim programları yazabilirsiniz. Kuyruk yöneticisi, bu kuyruktaki iletileri, komut satırından ya da işlem ve denetim panolarından girilen komutları işlediği gibi işler. Bu teknik, Programları yönetmek için yazma IBM MQbaşlıklı konuda açıklanmıştır ve IBM MQ for z/OSile birlikte gönderilen Mail Manager örnek uygulamasında gösterilmiştir. Bu örneğe ilişkin açıklamalar için bkz. IBM MQ for z/OS için örnek programlar .

Nesne öznitelikleri

Bir nesnenin özellikleri, öznitelikleriyle tanımlanır. Bazılarını belirtebilirsiniz, diğerlerini yalnızca görüntüleyebilirsiniz.

Örneğin, bir kuyruğun sığabileceği ileti uzunluğu üst sınırı **MaxMsgLength** özniteliğiyle tanımlanır; bir kuyruk yaratırken bu özniteliği belirtebilirsiniz. **DefinitionType** özniteliği, kuyruğun nasıl yaratıldığını belirtir; yalnızca bu özniteliği görüntüleyebilirsiniz.

IBM MQiçinde bir özniteliğe başvurmanın iki yolu vardır:

- Örneğin, **MaxMsgLength**gibi PCF adını kullanarak.
- MQSC komut adı kullanılıyor; örneğin, MAXMSGL.

Kuyruk paylaşım grupları

z/OS

Aynı paylaşılan kuyruk kümesine erişebilen kuyruk yöneticileri, *kuyruk paylaşım grubu* (QSG) adı verilen bir grup oluşturur ve paylaşılan kuyrukları saklayan bir bağlaşım olanağı (CF) kullanarak birbirleriyle iletişim kurar. Bir QSG ' nin kesinlikle bir nesne olmadığını unutmayın.

Paylaşılan kuyruk, bir kuyruk paylaşım grubundaki bir ya da daha çok kuyruk yöneticisi tarafından erişilebilen iletilere sahip bir yerel kuyruk tipidir. Bu, aynı kuyruk yöneticisi kullanılarak birden çok uygulama tarafından paylaşılan bir kuyrukla aynı değildir.

Kuyruk paylaşım gruplarının adı en çok dört karakterdir. Ad ağınızda benzersiz olmalı ve herhangi bir kuyruk yöneticisi adından farklı olmalıdır.

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

Ek bilgi için bkz. “Paylaşılan kuyruklar ve kuyruk paylaşım grupları” sayfa 160 .

Sistem varsayılan nesneleri

Sistem varsayılan nesneleri , bir kuyruk yöneticisi yaratıldığında otomatik olarak yaratılan nesne tanımlamaları kümeleridir. Kuruluşunuzda bu nesne tanımlamalarından herhangi birini uygulamalarda kullanmak üzere kopyalayabilir ve değiştirebilirsiniz.

Varsayılan nesne adları, kök SYSTEM ' i içerir; örneğin, varsayılan yerel kuyruk SYSTEM.DEFAULT.LOCAL.QUEUEve varsayılan alıcı kanalı SYSTEM.DEF.RECEIVER. Bu nesneleri yeniden adlandıramazsınız; bu adların varsayılan nesneleri gereklidir.

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

Ek bilgi için Sistem ve varsayılan nesneler konusuna bakın.

Nesne tipleri

Yönetim görevlerinin çoğu, çeşitli IBM MQ *nesneleri*tiplerinin işlenmesini içerir.

IBM MQ nesnelerini adlandırma hakkında bilgi için bkz. “IBM MQ nesnelerini adlandırma” sayfa 35.

Bir kuyruk yöneticisinde yaratılan varsayılan nesnelere ilişkin bilgi için bkz. “Sistem varsayılan nesneleri” sayfa 15.

IBM MQ nesnelerinin farklı tipleriyle ilgili bilgi için aşağıdakilere bakın:

Kimlik doğrulama bilgileri nesneleri

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

Kuyruk yöneticisi kimlik doğrulama bilgileri nesnesi, TLS (Transport Layer Security; İletim Katmanı Güvenliği) için IBM MQ desteğinin bir bölümünü oluşturur. İptal edilen sertifikaları denetlemek için gereken tanımları sağlar. Sertifikasyon Yetkilileri, artık güvenilemeyen sertifikaları iptal ediyor.

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

Aşağıdaki IBM MQ denetim komutlarını bir kimlik doğrulama bilgi nesnesiyle kullanabilirsiniz:

- **setmqaut** (yetkiyi ver ya da iptal et)
- **dspmqaut** (nesne yetkilendirmesini görüntüle)
- **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. Aktarım Katmanı Güvenliği (TLS) kavramları ve IBM MQ içinde TLS güvenlik iletişim kuralları.

Kanallar

Kanallar, bir kuyruk yöneticisinden diğerine iletişim yolu sağlayan nesnelerdir.

Ek bilgi için bkz. “Kanallar” sayfa 28 .

İletişim bilgileri nesneleri

IBM MQ Multicast, düşük gecikme süresi, yüksek fanout, güvenilir çok noktaya yayın ileti sistemi sunar. Çoklu yayın iletimini kullanmak için bir iletişim bilgisi (COMMINFO) nesnesi gerekir.

Ek bilgi için bkz. “IBM MQ Çok hedefli” sayfa 101 .

COMMINFO nesnesi, çoklu yayın iletimi ile ilişkili öznitelikleri içeren bir IBM MQ nesnesidir. Bu özniteliklere ilişkin ek bilgi için DEFINE COMMINFObaşlıklı konuya bakın. COMMINFO nesnesi yaratılmasıyla ilgili ek bilgi için Getting started with multicastbaşlıklı konuya bakın.

Dinleyiciler

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

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

Dinleyici nesneleri , bir kuyruk yöneticisi kapsamı içinden dinleyici işlemlerinin başlatılmasını ve durdurulmasını yönetmenizi sağlayan IBM MQ nesnelerdir. Bir dinleyici nesnesinin özniteliklerini tanımlayarak aşağıdakileri yaparsınız:

- Dinleyici işlemini yapılandırın.

- Kuyruk yöneticisi başlatıldığında ve durduğunda dinleyici işleminin otomatik olarak başlatılıp durdurulmayacağını belirtin.

Önemli: z/OS IBM MQ for z/OSüzerinde dinleyici nesneleri desteklenmez. IBM MQ for z/OS ' in dinlemeyi nasıl uyguladığı hakkında daha fazla bilgi için kanal başlatıcısını kullanarak bkz. “z/OS üzerindeki kanal başlatıcısı” sayfa 156.

Namelistler

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

Ad listesi, diğer IBM MQ nesnelerinin listesini içeren bir IBM MQ nesnesidir. Genellikle, ad alanları, bir kuyruk grubunu tanımlamak için kullanıldıkları tetikleyici izleme programları gibi uygulamalar tarafından kullanılır. Bir ad melist kullanmanın avantajı, uygulamalardan bağımsız olarak korunmasıdır; bunu kullanan uygulamalardan herhangi biri durdurulmadan güncellenebilir. Ayrıca, bir uygulama başarısız olursa, ad listesi etkilenmez ve diğer uygulamalar bunu kullanmaya devam edebilir.

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

DEFINE NAMELIST ve ALTER NAMELIST MQSC komutlarını kullanarak ad listesi tanımlayabilir ve değiştirebilirsiniz.

Not: z/OS Alternatif olarak, z/OSüzerinde IBM MQ for z/OS işlemlerini ve denetim panolarını kullanabilirsiniz

Programlar, bu ad çizelgelerinde hangi kuyrukların bulunduğunu öğrenmek için MQI 'ı kullanabilir. Ad melistlerinin organizasyonu, uygulama tasarımcısının ve sistem yöneticisinin sorumluluğundadır.

Kullanılabilecek ad listesi özniteliklerinin listesi için bkz. Ad listesi öznitelikleri,

Süreç tanımlamaları

Süreç tanımlaması nesneleri, kuyruk yöneticisi tarafından kullanılmak üzere uygulamanın özniteliklerini tanımlayarak, uygulamaların işletmen müdahalesine gerek kalmadan başlatılmasını sağlar.

Süreç tanımlaması nesnesi, IBM MQ kuyruk yöneticisindeki bir tetikleyici olayına yanıt olarak başlayan bir uygulamayı tanımlar. Süreç tanımlaması öznitelikleri, uygulamaya özgü uygulama tanıtıcısını, uygulama tipini ve verileri içerir. Daha fazla bilgi için “IBM MQ tarafından belirli amaçlarla kullanılan kuyruklar” sayfa 26içindeki *Başlatma kuyrukları* konusuna bakın.

Bir uygulamanın işletmen müdahalesine gerek kalmadan başlatılabilmesi için, Tetikleyicileri kullanarak IBM MQ uygulamaların başlatılmasıbaşlıklı konuda açıklandığı gibi, uygulamanın öznitelikleri kuyruk yöneticisi tarafından bilinmelidir. Bu öznitelikler bir *süreç tanımlaması nesnesinde*tanımlanır.

Nesne yaratıldığında **ProcessName** özniteliği düzeltilebilir. Ancak, IBM MQ komutlarını kullanarak diğer öznitelikleri değiştirebilirsiniz.

Not: z/OS Alternatif olarak, z/OSüzerinde IBM MQ for z/OS işlemlerini ve denetim panolarını kullanabilirsiniz.

MQINQ-Inquire object attributesöğesini kullanarak tüm özniteliklerin değerlerini sorgulayabilirsiniz.

Kullanılabilecek süreç tanımlaması özniteliklerinin listesi için Süreç tanımlamaları için özniteliklerkonusuna bakın.

Kuyruklar

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

Ek bilgi için bkz. “Kuyruklar” sayfa 19 .

Kuyruk yöneticileri

IBM MQ kuyruk yöneticileri, uygulamalara kuyruğa alma hizmetleri sağlar ve bunlara ait olan kuyrukları yönetir.

Ek bilgi için bkz. “Kuyruk yöneticileri” sayfa 27 .

Hizmetler

Hizmet nesneleri, kuyruk yöneticisi başlatıldığında ya da durduğunda çalıştırılacak programları tanımlamanın bir yoludur.

Programlar aşağıdaki tiplerden biri olabilir:

Sunucular
Sunucu , SERVER olarak belirtilen SERVTYPE parametresine sahip bir hizmet nesnesidir. Sunucu hizmet nesnesi, belirtilen bir kuyruk yöneticisi başlatıldığında yürütülecek bir programın tanımıdır. Eşzamanlı olarak yalnızca bir sunucu işlemi eşgörünümü yürütülebilir. Çalışırken, bir sunucu işleminin durumu MQSC komutu DISPLAY SVSTATUS kullanılarak izlenebilir. Genellikle sunucu hizmeti nesneleri, gitmeyen harf işleyicileri ya da tetikleyici izleme programları gibi programların tanımlarıdır, ancak çalıştırılabilir programlar IBM MQile birlikte sağlananlarla sınırlı değildir. Ayrıca, bir sunucu hizmet nesnesi, belirtilen kuyruk yöneticisi programı sona erdirmek üzere kapatıldığında çalıştırılacak bir komutu içerecek şekilde tanımlanabilir.

Komutlar
Komut , SERVTYPE parametresi COMMAND olarak belirlenmiş bir hizmet nesnesidir. Komut hizmeti nesnesi, belirtilen bir kuyruk yöneticisi başlatıldığında ya da durdurulduğunda yürütülecek bir programın tanımlamasıdır. Bir komut işleminin birden çok eşgörünümü eşzamanlı yürütülebilir. Komut hizmeti nesneleri, program yürütüldüğünde kuyruk yöneticisinin programı izlememesinden dolayı sunucu hizmet nesnelerinden farklıdır. Genellikle komut hizmeti nesneleri, kısa ömürlü olan programların tanımlarıdır ve bir ya da daha fazla başka görev başlatma gibi belirli bir görevi gerçekleştirir.

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

Daha fazla bilgi için bkz. Hizmetlerle çalışma.

Depolama sınıfları

z/OS

Depolama sınıfı, bir ya da daha çok kuyruğu bir sayfa kümesiyle eşleyebilir.

Bu, söz konusu kuyruğa ilişkin iletilerin o sayfa kümesinde saklandığı (arabelleğe almaya tabi olduğu) anlamına gelir.

Depolama sınıfları yalnızca IBM MQ for z/OSüzerinde desteklenir.

Depolama sınıflarıyla ilgili daha fazla bilgi için bkz. z/OS üzerinde IBM MQ ortamınızın planlanması.

Konu nesneleri

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

Konu , belirli bir *konu dizgisine*abone olan ya da yayınlayan bir uygulama tarafından tanımlanır. Konu dizgisi, konuları eğik çizgi karakteriyle (/) ayırarak bir konu sıradüzeni belirtebilir. Bu, bir *konu ağacı*tarafından görselleştirilebilir. Örneğin, bir uygulama `/Sport/American Football` ve `/Sport/Soccer`konu dizgilerinde yayınlandıysa, `Sport` üst düğümü iki alt öğesi olan `American Football`ve `Soccer`olan bir konu ağacı yaratılır.

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

Konu nesnesinin TOPICSTR özniteliğinde o düğümün konu dizgisini belirterek, konu ağacındaki herhangi bir düğümde bir konu nesnesi yaratabilirsiniz. Yönetimle ilgili konu düğümü için başka öznitelikler de tanımlayabilirsiniz. Bu özniteliklere ilişkin ek bilgi için MQSC komutlarıya da PCF komutlarını kullanarak denetimin otomatikleştirilmesibaşlıklı konuya bakın. Her konu nesnesi, varsayılan olarak, özniteliklerini en yakın üst yönetim konusu düğümünden devralır.

Konu nesneleri, uygulama geliştiricilerinden tüm konu ağacını gizlemek için de kullanılabilir. `/Sport/American Football`konusu için `FOOTBALL.US` adlı bir konu nesnesi yaratıldıysa, bir uygulama aynı sonuçla `/Sport/American Football` dizgisi yerine `FOOTBALL.US` adlı nesneyi yayınlayabilir ya da bu nesneye abone olabilir.

Bir konu nesnesindeki konu dizgisine #, +,/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 nesneleriyle ilgili daha fazla bilgi için bkz. “İleti alışverişi yayınla/abone ol” sayfa 59.

İlgili kavramlar

“İleti kuyruklama girişine giriş” sayfa 5
IBM MQ ürünleri, programların tutarlı bir uygulama programlama arabirimi kullanarak farklı bileşenlerden (işlemciler, işletim sistemleri, altsistemler ve iletişim protokolleri) oluşan bir ağ üzerinden birbirleriyle iletişim kurmalarını sağlar.

İlgili başvurular

MQSC komutları

Kuyruklar

IBM MQ kuyruklarına ve kuyruk özniteliklerine giriş.

İletiler bir kuyrukta saklanır; böylece, koyma uygulaması iletisine yanıt bekliyorsa, o yanıtı beklerken başka işler de yapabilir. Uygulamalar, İleti Kuyruğu Arabirimi 'ne genel bakışbaşlıklı konuda açıklanan İleti Kuyruğu Arabirimi 'ni (MQI) kullanarak bir kuyruğa erişir.

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

Kuyruk, bir kuyruk yöneticisi aracılığıyla korunur. Çoğu durumda, her kuyruk fiziksel olarak kuyruk yöneticisi tarafından yönetilir, ancak bu bir uygulama programı tarafından anlaşılır değildir. IBM MQ for z/OS paylaşılan kuyruklar, kuyruk paylaşım grubundaki 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 özgüdür.

Uygulamanızdan geçici işler için *dinamik olarak* yerel kuyruklar yaratabilirsiniz. Örneğin, *yanıtlama* kuyrukları yaratabilirsiniz (uygulama sona erdikten sonra bu kuyruklar gerekmez). Daha fazla bilgi için “Dinamik ve Model kuyrukları” sayfa 24 başlıklı konuya bakın.

Bir kuyruğu kullanmadan önce, kuyrukla ne yapmak istediğinizi belirterek kuyruğu açmanız gerekir. Örneğin, aşağıdakiler için bir kuyruk açabilirsiniz:

- Yalnızca iletilere göz atılıyor (iletileri almıyor)
- İletileri alma (ve erişimi diğer programlarla paylaşma ya da özel erişimle)
- İletileri kuyruğa koyma
- Kuyruğun öznitelikleri hakkında sorma
- Kuyruğun özniteliklerini ayarlama

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

Kuyrukların öznitelikleri

Kuyruk tanımlandığında bir kuyruğun bazı öznitelikleri belirlenir ve daha sonra değiştirilemez (örneğin, kuyruğun tipi). Kuyrukların diğer öznitelikleri, değiştirilebilecek öznitelikler içinde gruplanabilir:

- Kuyruk işlenirken kuyruk yöneticisi tarafından (örneğin, bir kuyruğun yürürlükteki derinliği)
- Yalnızca komutlarla (örneğin, kuyruğun tanımlama metni)
- Uygulamalar tarafından, MQSET çağrısıyla (örneğin, kuyrukta koyma işlemlerine izin verilip verilmediği)

MQINQ çağrıyı kullanarak tüm özniteliklerin değerlerini bulabilirsiniz.

Birden çok kuyruk tipi için ortak olan öznitelikler şunlardır:

QName
Kuyruğun adı.

QTür
Kuyruğun tipi.

QDesc
Kuyruğun tanımlama metni.

InhibitGet
Programların kuyruktan ileti almasına izin verilip verilmediğini belirler. Ancak, uzak kuyruklardan hiçbir zaman ileti almazsınız.

InhibitPut
Programların kuyruğa ileti koymasına izin verilip verilmediğini belirler.

DefPriority
Kuyruğa konan iletilere ilişkin varsayılan öncelik.

DefPersistence
Kuyruğa konan iletiler için varsayılan kalıcılık

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

z/OS **Scope** özniteliği z/OS üzerinde desteklenmez

Bu özniteliklerin tam açıklaması için Kuyruklara ilişkin özniteliklerbaşlıklı konuya bakın.

Kuyrukları tanımlama yolları

MQSC DEFINE komutunu ya da PCF Kuyruk Yarat komutunu kullanarak IBM MQ kuyruklarını tanımlayabilirsiniz. Komutlar, kuyruğun tipini ve özniteliklerini belirler. Örneğin, bir yerel kuyruk nesnesi, uygulamalar MQI çağrılarında o kuyruğa başvurduğunda ne olacağını belirten özniteliklere sahiptir. Öznitelik örnekleri şunlardır:

- Uygulamaların kuyruktan ileti alıp alamayacağını belirler (GET enabled)
- Uygulamaların kuyruğa ileti yerleştirip yerleştiremeyeceğini belirler (PUT etkin)
- Kuyruğa erişimin bir uygulama için özel mi, yoksa uygulamalar arasında paylaşılan mi olduğunu belirler
- Aynı anda kuyrukta saklanabilecek ileti sayısı üst sınırı (kuyruk derinliği üst sınırı)
- Kuyruğa konabilecek ileti uzunluğu üst sınırı

Kuyrukları tanımlamak için kullanabileceğiniz platforma özgü çeşitli arabirimler de vardır.

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

"Gönderilmeyen ileti kuyrukları" sayfa 47
Teslim edilmeyen iletiler kuyruğu (ya da teslim edilmeyen iletiler kuyruğu), iletilerin doğru hedeflerine yöneltilemeyecekleri durumlarda gönderildikleri kuyruktur. Her kuyruk yöneticisinin genellikle bir gönderilmeyen ileti kuyruğu vardır.

PCF komutlarını kullanarak yönetimi otomatikleştirme

IBM MQ Console içindeki kuyruklarla çalışma

İlgili görevler

IBM MQ ' nin MQSC komutlarını kullanarak denetlenmesi

MQ Explorer ile kuyruk yöneticilerinin ve nesnelerin yaratılması ve yapılandırılması

IBM i CL komutlarını kullanarak IBM MQ for IBM i yönetimi

z/OS IBM MQ for z/OS üzerinde MQSC ve PCF komutlarını yayınlayabileceğiniz kaynaklar

İlgili başvurular

"Paylaşılan kuyruklar ve küme kuyrukları arasındaki karşılaştırma" sayfa 56
Bu bilgiler, paylaşılan kuyrukları ve küme kuyruklarını karşılaştırmanıza ve hangilerinin sisteminiz için daha uygun olduğuna karar vermenize yardımcı olmak üzere tasarlanmıştır.

İlgili bilgiler

"Paylaşılan kuyruk nedir?" sayfa 160

Yerel kuyruklar

İletim, başlatma, gönderilmeyen harf, 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 yöneticisinin özel amaçlarla kullandığı bazı yerel kuyrukları olabilir:

İletim kuyrukları

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

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

1. Uzak bir kuyruğun yerel tanımının XMITQ özniteliğinde adı belirtilen iletim kuyruğu.
2. Uzak kuyruk yöneticisiyle aynı adı taşıyan bir iletim kuyruğu. Bu değer, uzak kuyruğun yerel tanımının XMITQ ' daki varsayılan değeridir.
3. Yerel kuyruk yöneticisinin DEFXMITQ özniteliğinde belirtilen iletim kuyruğu.

İleti kanalı aracısı , iletim kuyruğuyla ilişkilendirilmiş bir kanal programıdır ve iletiyi sonraki hedefine iletir. Sonraki hedef, ileti kanalının bağlı olduğu kuyruk yöneticisidir. İletinin son hedefiyle aynı kuyruk yöneticisi olması gerekmez. İleti bir sonraki hedefine teslim edildiğinde, iletim kuyruğundan silinir. İleti, son hedefine giden yolculuğunda birçok kuyruk yöneticisini geçmek zorunda olabilir. Rota boyunca her bir kuyruk yöneticisinde, sonraki hedefe iletilmeyi bekleyen iletileri tutan bir iletim kuyruğu tanımlamanız gerekir. Normal bir iletim kuyruğu, iletiler farklı hedeflere sahip olsa da, sonraki hedefe ilişkin iletileri tutar. Küme iletim kuyruğu, birden çok hedefe ilişkin iletileri tutar. Her iletinin `correlID` , iletiyi sonraki hedefine aktarmak için iletinin yerleştirildiği kanalı tanımlar.

Bir kuyruk yöneticisinde birden çok iletim kuyruğu tanımlayabilirsiniz. Her biri farklı bir hizmet sınıfı için kullanılan aynı hedef için birden çok iletim kuyruğu tanımlayabilirsiniz. Örneğin, aynı hedefe giden küçük iletiler ve büyük iletiler için farklı iletim kuyrukları yaratmak isteyebilirsiniz. Daha sonra büyük iletilerin daha küçük iletileri tutmaması için iletileri farklı ileti kanalları kullanarak aktarabilirsiniz. Küme kuyruklarına ya da küme konularına ilişkin tüm iletiler varsayılan olarak tek bir küme iletim

kuyruğuna SYSTEM.CLUSTER.TRANSMIT.QUEUEyerleştirilir. Bir seçenek olarak, varsayılan değeri değiştirebilir ve farklı küme kuyruğu yöneticilerine giden ileti trafiğini farklı küme iletim kuyruklarına ayırabilirsiniz. DEFCLXQ kuyruk yöneticisi özniteliğini CHANNELolarak ayarlarsanız, her küme gönderen kanalı ayrı bir küme iletim kuyruğu yaratır. Diğer bir seçenek olarak, küme gönderen kanallarının kullanması için küme iletim kuyruklarını el ile tanımlayabilirsiniz.

İletim kuyrukları, bir ileti kanalı aracısının iletileri ileriye doğru göndermesini tetikleyebilir; bkz. Tetikleyicileri kullanarak IBM MQ uygulamalarının başlatılması.

z/OS IBM MQ for z/OSişletim sistemlerinde, grup içi kuyruğa alma özelliğini kullanıyorsanız, iletim kuyruğuna *grup içi kuyruğa alma aracısı*tarafından hizmet gönderilir. IBM MQ for z/OSüzerinde grup içi kuyruğa alma kullanılırken paylaşılan iletim kuyruğu kullanılır.

Başlatma kuyrukları

Başlatma kuyruğu , bir uygulama kuyruğunda tetikleyici olay oluştuğunda kuyruk yöneticisinin tetikleyici iletisi koyduğu yerel bir kuyruktur.

Tetikleme olayı, bir programın bir kuyruğu işlemeye başlamasına neden olması amaçlanan bir olaydır. Örneğin, bir olay gelen 10 'dan fazla ileti olabilir. Tetikleyicilerin nasıl çalıştığına ilişkin ek bilgi için Tetikleyicileri kullanarak IBM MQ uygulamalarının başlatılmasıbaşlıklı konuya bakın.

Teslim edilmeyen (teslim edilmeyen) kuyruk

Teslim edilmeyen ileti kuyruğu , kuyruk yöneticisinin teslim edemediği iletileri koyduğu yerel bir kuyruktur.

Kuyruk yöneticisi, ileti gönderilmeyen iletiler kuyruğuna bir ileti yerleştirdiğinde, iletiye bir üstbilgi ekler. Üstbilgi bilgileri, kuyruk yöneticisinin iletiyi ileti gönderilmeyen iletiler kuyruğuna koymasının nedenini içerir. Ayrıca, özgün iletinin hedefini, tarihi ve kuyruk yöneticisinin iletiyi gönderilmeyen iletiler kuyruğuna koymuş olduğu saati de içerir.

Uygulamalar, teslim edemediği iletiler için kuyruğu da kullanabilir. Daha fazla bilgi için Teslim edilmeyen ileti kuyruğunun kullanılmasıbaşlıklı konuya bakın.

Sistem komut kuyruğu

Sistem komut kuyruğu , uygun yetkili uygulamaların IBM MQ komutlarını gönderebildiği bir kuyruktur. Bu kuyruklar, kuyruk yöneticisinin bunları işlemesi için hazır durumda, altyapınızda desteklendiği şekilde PCF, MQSC ve CL komutlarını alır.

z/OS IBM MQ for z/OS ' da kuyruk SYSTEM.COMMAND.INPUT olarak adlandırılır; diğer platformlarda buna SYSTEM.ADMIN.COMMAND.QUEUEdenir. Kabul edilen komutlar platforma göre değişir. Ayrıntılar için bkz. Programlanabilir komut biçimleri başvurusu.

Sistemin varsayılan kuyrukları

Sistem varsayılan kuyrukları , sisteminize ilişkin kuyrukların ilk tanımlarını içerir. Bir kuyruk tanımlaması yarattığınızda, kuyruk yöneticisi tanımlamayı uygun sistem varsayılan kuyruğundan kopyalar. Kuyruk tanımlaması yaratılması, dinamik kuyruk yaratılmasından 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 bildirilir.

Uzak kuyruklar

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

Bir iletişim bağlantısı kurulduğunda, bir program uzak kuyruğa ileti gönderebilir. Bir program uzak kuyruktan hiçbir zaman ileti almaz.

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

Bir uzak kuyruğu açarken, kuyruğu tanımlamak için aşağıdakilerden birini belirtmeniz gerekir:

- Uzak kuyruğu tanımlayan yerel tanımlamanın adı. Bir uygulamanın bakış açısından bu, yerel bir kuyruk açmayla aynıdır. Bir uygulamanın bir kuyruğun yerel mi, yoksa uzak mı olduğunu bilmesine gerek yoktur.

  IBM idışındaki tüm altyapılarda uzak kuyruk için yerel bir tanımlama yaratmak üzere DEFINE QREMOTE komutunu kullanın.

  IBM i IBM isistemlerinde CRTMQMQ komutunu kullanın.
- Uzak kuyruk yöneticisinin adı ve uzak kuyruk yöneticisinin tanıdığı kuyruk adı.

Uzak kuyruklara ilişkin yerel tanımlamalar, “Kuyrukların öznitelikleri” sayfa 20içinde açıklanan ortak özniteliklere ek olarak üç özniteliğe de sahiptir. Bu üç öznitelik şunlardır:

RemoteQName
Kuyruğun sahip olduğu kuyruk yöneticisinin tanıdığı ad.

RemoteQMgrAdı
Sahip olan kuyruk yöneticisinin adı.

XmitQName
İletiler diğer kuyruk yöneticilerine iletilirken kullanılan yerel iletim kuyruğunun adı.

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

MQINQ çağrısıyla uzak bir kuyruğun yerel tanımlamasını karşılaştırırsanız, kuyruk yöneticisi yalnızca yerel tanımlamanın özniteliklerini döndürür; uzak kuyruk adı, uzak kuyruk yöneticisi adı ve iletim kuyruğu adı, uzak sistemdeki eşleşen yerel kuyruğun özniteliklerini döndürmez.

Ayrıca bkz. İletim kuyrukları.

Diğer ad kuyrukları

Diğer ad kuyruğu , başka bir kuyruğa ya da konuya erişmek için kullanabileceğiniz bir IBM MQ nesnesidir. Bu, birden çok programın farklı adlar kullanarak aynı kuyrukla çalışabileceği anlamına gelir.

Temel kuyruk olarak bilinen bir diğer adın çözülmesinden kaynaklanan kuyruk, platform tarafından desteklendiği şekilde, aşağıdaki kuyruk tiplerinden herhangi biri olabilir:

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

Diğer ad bir konuya da çözülebilir. Bir uygulama şu anda bir kuyruğa ileti gönderiyorsa, kuyruk adını konu için diğer ad yaparak bir konuda yayınlama yapılabilir. Uygulama kodunda değişiklik yapılması gerekmez.

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

Diğer ad kuyruklarının kullanımına örnek olarak, bir sistem denetimcisinin temel kuyruk adı (diğer adın çözüldüğü kuyruk) ve diğer ad kuyruğu adı için farklı erişim yetkileri vermesi verilebilir. Bu, bir programın ya da kullanıcının diğer ad kuyruğunu kullanma yetkisi olduğu, ancak temel kuyruğu kullanma yetkisi olmadığı anlamına gelir.

Diğer bir seçenek olarak, yetkilendirme diğer ad için koyma işlemlerini engelleyecek şekilde ayarlanabilir, ancak temel kuyruk için bunlara izin verebilir.

Bazı uygulamalarda, diğer ad kuyruklarının kullanılması, sistem denetimcilerinin uygulamayı değiştirmek zorunda kalmadan bir diğer ad kuyruğu nesnesinin tanımını kolayca değiştirebileceği anlamına gelir.

IBM MQ , programlar bu adı kullanmayı denediğinde diğer ad için yetki denetimi yapar. Programın, diğer adın çözüldüğü ada erişme yetkisinin olup olmadığını denetlemez. Bu nedenle, bir programın diğer ad kuyruğu adına erişim yetkisi olabilir, ancak çözülen kuyruk adına erişim yetkisi verilmez.

"Kuyruklar" sayfa 19içinde açıklanan genel kuyruk özniteliklerine ek olarak, diğer ad kuyruklarının bir **BaseQName** özniteliği vardır. Bu, diğer adın çözümleneceği temel kuyruğun adıdır. Bu özniteliğin tam açıklaması için bkz. BaseQName (MQCHAR48).

InhibitGet ve **InhibitPut** öznitelikleri (bkz. "Kuyruklar" sayfa 19) diğer ad kuyruklarına ait. Örneğin, ALIAS1 diğer ad kuyruk adı BASE temel kuyruk adına çözümlenirse, ALIAS1 üzerindeki kısıtlamalar yalnızca ALIAS1 değerini etkiler ve BASE engellenmez. Ancak, BASE üzerindeki kısıtlamalar ALIAS1' i de etkiler.

DefPriority ve **DefPersistence** öznitelikleri de diğer ada aittir. 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 kuyrukların özelliklerine, dinamik kuyrukların kullanımına, dinamik kuyrukları kullanırken dikkat edilmesi gereken bazı noktalara ve model kuyruklarına ilişkin bir öngörü sağlar.

Bir uygulama programı model kuyruğu açmak için bir MQOPEN çağrısı verdiğinde, kuyruk yöneticisi model kuyruğuyla aynı özniteliklere sahip bir yerel kuyruk eşgörünümü yaratır. Model kuyruğunun *DefinitionType* alanının değerine bağlı olarak, kuyruk yöneticisi 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 paylaşılan kuyruklar olamaz.

  Kuyruk paylaşım gruplarının yalnızca IBM MQ for z/OSüzerinde kullanılabildiğine dikkat edin.
- Yalnızca kalıcı olmayan iletileri tutarlar.
- Bunlar kurtarılamaz.
- Bunlar, kuyruk yöneticisi başlatıldığında silinir.
- Kuyruğu yaratan MQOPEN çağrıyı yayınlayan uygulama kuyruğu kapattığında ya da sonlandırdığında bunlar silinir.
  - Kuyrukta kesinleştirilen iletiler varsa, bunlar silinir.
  - Şu anda kuyruğa karşı bekleyen kesinleştirilmemiş MQGET, MQPUT ya da MQPUT1 çağrıları varsa, kuyruk mantıksal olarak siliniyor olarak işaretlenir ve yalnızca kapatma işleminin bir parçası olarak fiziksel olarak silinir (bu çağrılar kesinleştirildikten sonra) ya da uygulama sona erdiğinde.
  - Kuyruk şu anda kullanılıyorsa (yaratma ya da başka bir uygulama tarafından), kuyruk mantıksal olarak siliniyor olarak işaretlenir ve yalnızca kuyruğu kullanan son uygulama tarafından kapatıldığında fiziksel olarak silinir.
  - Mantıksal olarak silinen bir kuyruğa (onu kapatmaktan başka) erişme girişimleri başarısız oldu. Neden kodu: MQRC_Q_DELETED.
  - MQCO_NONE, MQCO_DELETE ve MQCO_DELETE_PURGE, kuyruğu yaratan ilgili MQOPEN çağrısı için bir MQCLOSE çağrısında belirtildiğinde MQCO_NONE olarak işlenir.

Kalıcı dinamik kuyrukların özellikleri

Kalıcı dinamik kuyruklar aşağıdaki özelliklere sahiptir:

- Kalıcı ya da kalıcı olmayan iletileri tutarlar.

- Bunlar, sistem arızası durumunda kurtarılabilir.
- Bir uygulama (kuyruğu yaratan MQOPEN çağrısıyla ilgili değil) MQCO_DELETE ya da MQCO_DELETE_PURGE seçeneğini kullanarak kuyruğu başarıyla kapattığında bunlar silinir.
  - Kuyrukta hala ileti (kesinleştirilmiş ya da kesinleştirilmemiş) varsa, MQCO_DELETE seçeneğiyle bir kapatma isteği başarısız olur. Kuyrukta kesinleştirilmiş iletiler (kapanışın bir parçası olarak silinen iletiler) olsa da, kuyrukta kesinleştirilmemiş MQGET, MQPUT ya da MQPUT1 çağrıları varsa, MQCO_DELETE_PURGE seçeneğiyle bir kapatma isteği başarılı olur.
  - Silme isteği başarılı olursa, ancak kuyruk kullanımda olursa (yaratma ya da başka bir uygulama tarafından), kuyruk mantıksal olarak siliniyor olarak işaretlenir ve yalnızca, kuyruk kullanılarak son uygulama tarafından kapatıldığında fiziksel olarak silinir.
- Kuyruğu silme yetkisi olmayan bir uygulama kapattıysa, bunlar silinmez; ancak, kapanan uygulama kuyruğu yaratan MQOPEN çağrısına izin vermediyse. İlgili MQOPEN çağrısının geçerliliğini denetlemek için kullanılan kullanıcı kimliği (ya da MQOO_ALTERNATE_USER_AUTHORITY belirtildiyse, diğer kullanıcı kimliği) için yetki denetimleri gerçekleştirilir.
- Normal bir kuyruktakilerle aynı şekilde silinebilir.

Dinamik kuyrukların kullanımı

Aşağıdakiler için dinamik kuyrukları kullanabilirsiniz:

- Uygulama sonlandırıldıktan sonra kuyrukların tutulmasını gerektirmeyen uygulamalar.
- Başka bir uygulama tarafından işlenecek iletilere yanıt gerektiren uygulamalar. Bu tür uygulamalar, bir model kuyruğu açarak dinamik olarak bir yanıt kuyruğu oluşturabilir. Örneğin, bir istemci uygulaması şunları yapabilir:
  1. Dinamik bir kuyruk oluşturun.
  2. İstek iletisinin ileti tanımlayıcı yapısının **ReplyToQ** alanında adını belirtin.
  3. İsteği, sunucu tarafından işlenmekte olan bir kuyruğa yerleştirin.

  Sunucu, yanıt iletisini yanıt kuyruğuna yerleştirebilir. Son olarak, istemci yanıtı işleyebilir ve silme seçeneğiyle yanıt kuyruğunu kapatabilir.

Dinamik kuyruklar kullanılırken dikkat edilecek noktalar

Dinamik kuyrukları kullanırken aşağıdaki noktaları göz önünde bulundurun:

- Bir istemci-sunucu modelinde, her istemci kendi dinamik yanıt kuyruğunu yaratmalı ve kullanmalıdır. Dinamik bir yanıt kuyruğu birden çok istemci arasında paylaşılıyorsa, kuyruğa yönelik kesinleştirilmemiş etkinlik olduğundan ya da kuyruk başka bir istemci tarafından kullanıldığından, yanıt kuyruğunun silinmesi gecikebilir. Buna ek olarak, kuyruk mantıksal olarak silinmiş olarak ve sonraki API istekleri (MQCLOSE dışında) için erişilemez olarak işaretlenmiş olabilir.
- Uygulama ortamınız, uygulamalar arasında dinamik kuyrukların paylaşılmasını gerektiriyorsa, kuyruğa ilişkin tüm etkinlik kesinleştirildiğinde, kuyruğun yalnızca kapalı (silme seçeneğiyle) olmasını sağlayın. Bu, son kullanıcı tarafından olmalıdır. Bu, kuyruğun silinmesinin geciktirilmemesini sağlar ve mantıksal olarak siliniyor olarak işaretlendiği için kuyruğun erişilemez olduğu süreyi en aza indirir.

Model kuyrukları

Model kuyruğu , dinamik kuyruk yaratırken kullandığınız bir kuyruk tanımlaması şablonudur.

Kuyruk öznitelikleri için şablon olarak kullanmak istediğiniz model kuyruğunu adlandırmak üzere bir IBM MQ programından dinamik olarak yerel bir kuyruk oluşturabilirsiniz. Bu noktada, yeni kuyruğun bazı özniteliklerini değiştirebilirsiniz. Ancak, **DefinitionType**öğesini değiştiremezsiniz. Örneğin, kalıcı bir kuyruk gerekiyorsa, tanımlama tipi kalıcı olarak ayarlanmış bir model kuyruğu seçin. Bazı etkileşimli uygulamalar, yanıtları işledikten sonra bu kuyrukları tutmaları gerekmediğinden, sorgularına yanıt tutmak için dinamik kuyrukları kullanabilir.

MQOPEN çağrısının *nesne tanımlayıcısında* (MQOD) bir model kuyruğunun adını belirtirsiniz. Kuyruk yöneticisi, model kuyruğunun özniteliklerini kullanarak sizin için dinamik olarak yerel bir kuyruk yaratır.

Dinamik kuyruk için bir ad (tam olarak) ya da bir adın kökü (örneğin, ABC) belirtebilir ve kuyruk yöneticisinin buna benzersiz bir bölüm eklemesine izin verebilir ya da kuyruk yöneticisinin sizin için tam bir benzersiz ad atamasına izin verebilirsiniz. Kuyruk yöneticisi adı atarsa, adı MQOD yapısına koyar.

Bir model kuyruğuna doğrudan MQPUT1 çağrısı veremezsiniz; ancak, bir model kuyruğu açarak yaratılan dinamik kuyruğa bir MQPUT1 verebilirsiniz.

Bir model kuyruğu için MQSET ve MQINQ yayınlanamaz. MQOO_INQUIRE ya da MQOO_SET ile bir model kuyruğunun açılması, devingen olarak yaratılan kuyruk için sonraki MQINQ ve MQSET çağrılarının yapılmasıyla sonuçlanır.

Bir model kuyruğunun öznitelikleri, yerel kuyruktakilerin bir altkümesidir. Tam tanımlama için Kuyruklara ilişkin özniteliklerbaşlıklı konuya bakın.

IBM MQ tarafından belirli amaçlarla kullanılan kuyruklar

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

IBM MQ ' u kullanabilmek için önce bu kuyrukları tanımlamanız gerekir.

Başlatma kuyrukları

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

Bir kuyruk yöneticisi tetikleme kullanacaksa, o kuyruk yöneticisi için en az bir başlatma kuyruğu tanımlanmalıdır. Tetikleyici için nesnelerin yönetilmesi, runmqtrmve Tetikleyici kullanarak IBM MQ uygulamalarının başlatılması başlıklı konuya bakın.

İletim kuyrukları

İletim kuyrukları, uzak kuyruk yöneticisine yönlendirilmiş iletileri geçici olarak saklayan kuyruklardır. Yerel kuyruk yöneticisinin doğrudan ileti göndereceği her uzak kuyruk yöneticisi için en az bir iletim kuyruğu tanımlamanız gerekir. Bu kuyruklar uzak yönetimde de de kullanılır; bkz. Yerel bir kuyruk yöneticisinden uzaktan yönetim. Dağıtılmış kuyruğa alma işleminde iletim kuyruklarının kullanımına ilişkin bilgi için bkz. IBM MQ dağıtılmış kuyruğa alma teknikleri.

Her kuyruk yöneticisinin varsayılan bir iletim kuyruğu olabilir. Kümenin bir parçası olmayan bir kuyruk yöneticisi uzak kuyruğa bir ileti yerleştirirse, varsayılan işlem varsayılan iletim kuyruğunun kullanılmasıdır. Hedef kuyruk yöneticisiyle aynı adı taşıyan bir iletim kuyruğu varsa, ileti o iletim kuyruğuna yerleştirilir. **RQMNAME** parametresinin hedef kuyruk yöneticisiyle eşleştiği bir kuyruk yöneticisi diğer adı tanımlaması varsa ve **XMITQ** parametresi belirtilirse, ileti **XMITQ**adlı iletim kuyruğuna yerleştirilir. **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 `SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE`adlı bir model küme iletim kuyruğu vardır. Bir kuyruk yöneticisi tanımladığınızda, bu kuyrukların tanımlamaları varsayılan olarak yaratılır. **DEFCLXQ**kuyruk yöneticisi özniteliği `CHANNEL`olarak ayarlanırsa, yaratılan her küme gönderen kanal 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 birindeki iletileri aynı kümedeki diğer kuyruk yöneticilerine gönderir.

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

Gönderilmeyen ileti kuyrukları
Teslim edilmeyen (teslim edilmeyen) kuyruk, doğru hedeflerine yöneltilemeyen iletileri saklayan bir kuyruktur. Örneğin, hedef kuyruk doluyken bir ileti yöneltilemez. Sağlanan gönderilmeyen iletiler kuyruğuna SYSTEM.DEAD.LETTER.QUEUEadı verilir.

Dağıtılmış kuyruğa alma için, dahil olan her kuyruk yöneticisinde bir teslim etmeyen ileti kuyruğu tanımlayın.

Komut kuyrukları
SYSTEM.ADMIN.COMMAND.QUEUEkomut kuyruğu, uygun yetkili uygulamaların işlenmek üzere MQSC komutları gönderebildiği yerel bir kuyruktur. 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 yanıt kuyruğuna yanıt döndürür.

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

Yanıt kuyrukları
Bir uygulama bir istek iletisi gönderdiğinde, iletiyi alan uygulama gönderen uygulamaya bir yanıt iletisi gönderebilir. Bu ileti, genellikle gönderen uygulamanın yerel kuyruğu olan yanıtlama kuyruğu adı verilen bir kuyruğa konmuştur. Yanıt kuyruğunun adı, gönderen uygulama tarafından ileti tanımlayıcısının bir parçası olarak belirlenir.

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

Bir izleme kodu ekleme olayı oluştuğunda, kuyruk yöneticisi olay kuyruğuna bir olay iletisi koyar. Bu ileti daha sonra bir izleme uygulaması tarafından okunabilir; bu, bir yöneticiyi bilgilendirebilir ya da olay bir sorun olduğunu gösterirse bir çözüm işlemi başlatabilir.

Not: Tetikleme olayları, özel işlemden geçirme olaylarından farklıdır. Tetikleyici olaylara aynı koşullar neden olmaz ve olay iletileri oluşturulmaz.

Özel işlemden geçirme olaylarıyla ilgili daha fazla bilgi için Özel İşlemden geçirme olaylarıbaşlıklı konuya bakın.

Kuyruk yöneticileri

Kuyruk yöneticilerine ve uygulamalara sağladıkları kuyruğa alma hizmetlerine giriş.

Bir programın, o kuyruk yöneticisinin hizmetlerini kullanabilmesi için önce bir kuyruk yöneticisiyle bağlantısı olmalıdır. Bir program bu bağlantıyı belirtik olarak yapabilir (MQCONN ya da MQCONNX çağrısıyla) ya da örtük olarak bağlantı kurulabilir (bu, programın çalıştığı altyapıya ve ortama bağlıdır).

IBM MQ kuyruk yöneticisi aşağıdaki işlemleri sağlar:

- Nesne öznitelikleri, alınan komutlara göre değişir.
- Tetikleyici olayları ya da özel işlemden geçirme olayları gibi özel olaylar, uygun koşullar karşılandığında oluşturulur.
- MQPUT çağrısı yapan uygulamanın istediği gibi, iletiler doğru kuyruğa konuyor. Bunun yapılamayacağı konusunda uygulamaya bilgi verilir ve uygun bir neden kodu verilir.

Her kuyruk tek bir kuyruk yöneticisine aittir ve bu kuyruk yöneticisinin *yerel kuyruğu* olduğu söylenmektedir. Bir uygulamanın bağlı olduğu kuyruk yöneticisinin, o uygulamanın *yerel kuyruk yöneticisi* olduğu söylenmektedir. Uygulama için, yerel kuyruk yöneticisine ait olan kuyruklar yerel kuyruklardır.

Uzak kuyruk , başka bir kuyruk yöneticisine ait olan bir kuyruktur. *Uzak kuyruk yöneticisi* , yerel kuyruk yöneticisi dışında herhangi bir kuyruk yöneticisidir. Uzak kuyruk yöneticisi, ağ üzerindeki uzak bir makinede var olabilir ya da yerel kuyruk yöneticisiyle aynı makinede var olabilir. IBM MQ , aynı makinede birden çok kuyruk yöneticilerini destekler.

Bazı MQI çağrılarında bir kuyruk yöneticisi nesnesi kullanılabilir. Örneğin, MQINQMQI çağrılarını kullanarak kuyruk yöneticisi nesnesinin öznitelikleri hakkında bilgi alabilirsiniz.

Kuyruk yöneticilerinin öznitelikleri

Her kuyruk yöneticisiyle ilişkilendirilmiş, özelliklerini tanımlayan bir öznitelik (ya da özellikler) kümesidir.Bir kuyruk yöneticisinin bazı öznitelikleri, yaratıldığında düzeltilmiştir; diğerlerini IBM MQ komutlarını kullanarak değiştirebilirsiniz. MQINQ aramasını kullanarak TLS (Transport Layer Security; İletim Katmanı Güvenliği) şifrelemesi için kullanılanlar dışında tüm özniteliklerin değerlerini sorgulayabilirsiniz.

Sabit öznitelikler şunlardır:

- Kuyruk yöneticisinin adı
- Kuyruk yöneticisinin çalıştığı platform (örneğin, Windows)
- Kuyruk yöneticisinin desteklediği sistem denetimi komutlarının düzeyi
- Kuyruk yöneticisi tarafından işlenen iletilere atayabileceğiniz öncelik üst sınırı
- Programların IBM MQ komutlarını gönderebildiği kuyruğun adı
- Kuyruk yöneticisinin işleyebileceği ileti uzunluğu üst sınırı z/OS (yalnızca IBM MQ for z/OS içinde değişmez)
- Programlar ileti yerleştirdiğinde ve aldığında kuyruk yöneticisinin eşitlemeyi destekleyip desteklemediği

Değiştirilebilir öznitelikler şunlardır:

- Kuyruk yöneticisinin tanımlama metni
- Kuyruk yöneticisinin MQI çağrılarını işlerken karakter dizgileri için kullandığı karakter kümesinin tanıtıcısı
- Kuyruk yöneticisinin tetikleyici ileti sayısını sınırlamak için kullandığı zaman aralığı
- z/OS Kuyruk yöneticisinin süresi dolan iletiler için hangi sıklıkta taranacağını saptamak için kullandığı zaman aralığı (yalnızca IBM MQ for z/OS)
- Kuyruk yöneticisinin teslim edilmeyen ileti kuyruğunun adı
- Kuyruk yöneticisinin varsayılan iletim kuyruğunun adı
- Herhangi bir bağlantı için açık tutamaç sayısı üst sınırı
- Çeşitli olay raporlaması kategorilerinin etkinleştirilmesi ve devre dışı bırakılması
- Bir iş birimindeki kesinleştirilmemiş ileti sayısı üst sınırı

Kuyruk yöneticileri ve iş yükü yönetimi

Aynı kuyruk için birden çok tanımlaması olan bir kuyruk yöneticileri kümesi ayarlayabilirsiniz (örneğin, kümedeki kuyruk yöneticileri birbirinin klonları olabilir). Belirli bir kuyruğa ilişkin iletiler, kuyruğun bir eşgörünümünü barındıran herhangi bir kuyruk yöneticisi tarafından işlenebilir. İş yükü yönetimi algoritması, iletiyi hangi kuyruk yöneticisinin işleyeceğine karar verir ve kuyruk yöneticileriniz arasında iş yükünü yayar; ek bilgi için Küme iş yükü yönetimi algoritması konusuna bakın.

Kanallar

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

Kanallar, iletileri bir kuyruk yöneticisinden diğerine taşımak için kullanılır ve uygulamaları temel iletişim protokollerinden korur. Kuyruk yöneticileri aynı sistemde ya da aynı platformda ya da farklı platformlarda var olabilir. Gönderilen iletiler birçok yerden kaynaklanabiliyor:

- Verileri bir düğümden diğerine aktaran kullanıcı tarafından yazılan uygulama programları.
- PCF komutlarını ya da MQAI ' yi kullanan kullanıcı tarafından yazılan denetim uygulamaları.
- IBM MQ Explorer.
- Başka bir kuyruk yöneticisine özel işlemden geçirme olayı iletileri gönderen kuyruk yöneticileri.

- Uzak denetim komutlarını başka bir kuyruk yöneticisine gönderen kuyruk yöneticileri. Örneğin, MQSC komutları ya da administrative REST APIkullanılarak.

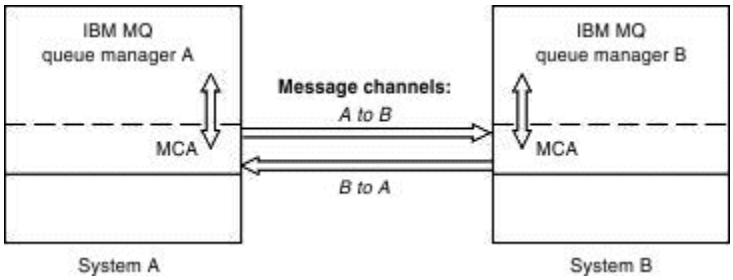
Bir kanalın iki tanımı vardır: bağlantının her bir ucunda bir tane. Kuyruk yöneticilerinin birbiriyle iletişim kurması için, kuyruk yöneticisinde ileti gönderecek bir kanal nesnesi tanımlamanız ve bunları alacak kuyruk yöneticisinde tamamlayıcı bir nesne tanımlamanız gerekir. Bağlantının her bir ucunda aynı *kanal adı* kullanılmalı ve kullanılan *kanal tipi* uyumlu olmalıdır.

IBM MQiçinde, bu kategoriler içinde farklı kanal tiplerine sahip üç kanal kategorisi vardır:

- Tek yönlü olan ileti kanalları ve iletileri bir kuyruk yöneticisinden diğerine aktarma.
- İki yönlü olan MQI kanalları ve MQI çağrılarını IBM MQ MQI client ' den bir kuyruk yöneticisine ve yanıtları bir kuyruk yöneticisinden IBM MQ istemcisine aktarması.
- İki yönlü olan ve AMQP istemcisini bir sunucu makinesindeki kuyruk yöneticisine bağlayan AMQP kanalları. IBM MQ , AMQP çağrılarını ve yanıtlarını AMQP uygulamaları ile kuyruk yöneticileri arasında aktarmak için AMQP kanallarını kullanır

İleti kanalları

Bir ileti kanalının amacı, iletileri bir kuyruk yöneticisinden diğerine aktarmaktır. İstemci sunucusu ortamı ileti kanallarını gerektirmez.



Şekil 2. İki kuyruk yöneticisi arasındaki ileti kanalları

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

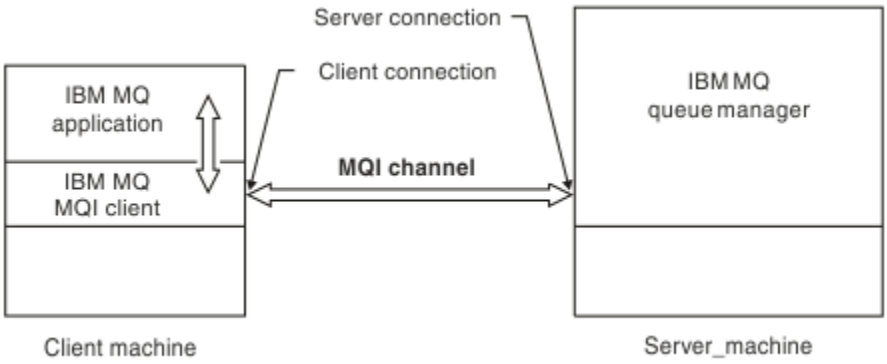
Bir ileti kanalı, *ileti kanalı aracılarını* (MCA ' lar) kullanarak iki kuyruk yöneticisini bağlar. Bir kanalın her ucunda bir ileti kanalı aracısı vardır. MCA ' nın birden çok iş parçacığı kullanarak ileti aktarmasına izin verebilirsiniz. Bu işlem *ardışık işlem*olarak bilinir. Pipelining, MCA ' nın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

Kanallar hakkında daha fazla bilgi için bkz. Kanal çıkışı çağrıları ve veri yapılarıve “Dağıtılmış kuyruğa alma bileşenleri” sayfa 45.

MQI kanalları

İleti Kuyruğu Arabirimi (MQI) kanalı, IBM MQ MQI client ' i sunucu makinesindeki bir kuyruk yöneticisine bağlar ve IBM MQ MQI client uygulamasından bir `MQCONN` ya da `MQCONNX` çağrısı yayınladığınızda oluşturulur.

İki yönlü bir bağlantıdır ve ileti verilerini içeren `MQPUT` çağrıları ve `MQGET` çağrıları da içinde olmak üzere, yalnızca MQI çağrılarının ve yanıtlarının aktarılması için kullanılır. Kanal tanımlamalarını yaratmanın ve kullanmanın farklı yolları vardır (bkz. MQI kanallarını tanımlama).



Şekil 3. MQI kanalında istemci bağlantısı ve sunucu bağlantısı

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

MQI kanal tanımlamaları için iki kanal tipi vardır. İki yönlü MQI kanalını tanımlar.

İstemci-bağlantı kanalı
Bu tip IBM MQ MQI clientiçindir.

Sunucu bağlantısı kanalı
Bu tip, IBM MQ MQI client ortamında çalışan IBM MQ uygulamasının iletişim kuracağı kuyruk yöneticisini çalıştıran sunucu içindir.

AMQP kanalları

Multi

Tek tip AMQP kanalı vardır.

Kanalı, bir AMQP ileti sistemi uygulamasını bir kuyruk yöneticisine bağlamak ve uygulamanın IBM MQ uygulamalarıyla ileti değiş tokuşu gerçekleştirmesini sağlamak için kullanabilirsiniz. AMQP kanalı, MQ Lightkullanarak bir uygulama geliştirmenize ve daha sonra, IBM MQtarafından sağlanan kurumsal düzeydeki olanaklardan yararlanarak bunu kurumsal uygulama olarak devreye almanıza olanak sağlar.

İstemci bağlantı kanalları

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

İstemci bağlantı kanalları, iletileri bir kuyruk yöneticisiyle istemci arasında taşımak için dağıtılmış kuyrukta kullanılır. Temel iletişim protokollerinden uygulamaları koruyorlar. İstemci, kuyruk yöneticisiyle aynı ya da farklı bir altyapıda var olabilir.

Kanal tanımları

Her kanal tipine ilişkin açıklamalar için bkz. “Kanal tanımları” sayfa 31 .

İlgili kavramlar

“Dağıtılmış kuyruğa alma ve kümeler” sayfa 41
Dağıtılmış kuyruğa alma, bir kuyruk yöneticisinden diğerine ileti gönderilmesi anlamına gelir. Alıcı kuyruk yöneticisi aynı makinede ya da başka bir makinede olabilir; yakınlarda ya da dünyanın diğer tarafında. Yerel kuyruk yöneticisiyle aynı altyapıda ya da IBM MQtarafından desteklenen herhangi bir altyapıda çalışabilir. Dağıtılmış bir kuyruğa alma ortamındaki tüm bağlantıları el ile tanımlayabilir ya da bir küme yaratabilir ve IBM MQ ' un bağlantı ayrıntılarının çoğunu sizin için tanımlamasına izin verebilirsiniz.

İleti Kuyruğu Arabirimi-Genel Bakış

İlgili görevler

Uzak IBM MQ nesnelerini yönetme

MQI kanalları durduruluyor

Sunucu ve istemci arasındaki bağlantıların yapılandırılması

İlgili başvurular

Kanal çıkışı çağrıları ve veri yapıları

“İletişim Ürünleri” sayfa 34

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

Kanal tanımları

IBM MQ ' in kullandığı farklı tipte ileti kanallarını ve MQI kanallarını tanımlayan çizelgeler.

İleti kanallarına atıfta bulunurken, sözcük kanalı genellikle bir kanal tanımının eşanlamlısı olarak kullanılır. Genellikle iki ucu olan tam bir kanaldan mı, yoksa tek ucu olan bir kanal tanımından mı bahsettiğimiz bağlamından anlaşılır.

İleti kanalları

İleti kanalı tanımlamaları aşağıdaki tiplerden biri olabilir:

İleti kanalı tanımlaması tipi	**Açıklama**
Gönderen	Gönderen kanal, kuyruk yöneticisinin diğer kuyruk yöneticilerine ileti göndermek için kullandığı bir ileti kanalıdır. Bir gönderen kanalını kullanarak ileti göndermek için, diğer kuyruk yöneticisinde, gönderen kanalıyla aynı ada sahip bir alıcı kanal da oluşturmanız gerekir. Bir "geri çağırma" mekanizması uyguluyorsanız, istekte bulunan kanallarla gönderen kanallarını da kullanabilirsiniz.
Sunucu	Sunucu kanalı, kuyruk yöneticisinin diğer kuyruk yöneticilerine ileti göndermek için kullandığı bir ileti kanalıdır. Bir sunucu kanalını kullanarak ileti göndermek için, diğer kuyruk yöneticisinde de sunucu kanalıyla aynı adı taşıyan bir alıcı kanalı yaratmanız gerekir. Sunucu kanallarını istekte bulunan kanallarla da kullanabilirsiniz. Bu durumda, kanalın diğer ucundaki istekte bulunan kanal tanımı, sunucu kanalı tanımının başlatılamasını ister. Sunucu, istekte bulunana ileti gönderir. Sunucu, ortak kanalın bağlantı adını bildiği sürece iletişimi başlatabilir.
Alıcı	Alıcı kanalı, kuyruk yöneticisinin diğer kuyruk yöneticilerinden ileti almak için kullandığı bir ileti kanalıdır. Alıcı kanalı kullanarak ileti almak için, diğer kuyruk yöneticisinde, bu alıcı kanalla aynı adı taşıyan bir gönderen ya da sunucu kanalı yaratmanız gerekir.

İleti kanalı tanımlaması tipi	Açıklama
İsteyen	İstekte Bulunan kanal, kuyruk yöneticisinin diğer kuyruk yöneticilerinden ileti almak için kullandığı bir ileti kanalıdır. İstekte bulunan bir kanal, uzak uçta tanımlanan ortak kanalı başlatmak için isteyebilir. Ortak kanal bir Sunucu kanalıysa, Sunucu kanalı başlatma isteğini kabul eder ve Sunucu kanal tanımında tanımlanan iletim kuyruğundan İstekte Bulunan kanalına ileti göndermeye başlar. Ortak kanal bir Gönderen kanalıysa, Gönderen kanalı başlatma isteğini kabul eder, ancak daha sonra İstek Gönderen ile olan bağlantıyı kapatır. Gönderen kanalı daha sonra başlar, ortak İstek Gönderen kanalı ile bir oturum üzerinde anlaşır ve Gönderen kanal tanımında tanımlanan iletim kuyruğundan ileti göndermeye başlar. Bu ikinci durum, İstek Gönderen kanalının Gönderen kanalını geri çağırması için bir geri arama mekanizması sağlar.
Küme-gönderen	Küme gönderen (CLUSSDR) kanal tanımı, küme kuyruk yöneticisinin küme bilgilerini tam havuzlardan birine gönderebildiği bir kanalın gönderme sonunu tanımlar. Kümeyi gönderen kanal, kuyruk yöneticisinin durumundaki değişiklikleri (örneğin, bir kuyruğun eklenmesi ya da kaldırılması) havuza bildirmek için kullanılır. İletileri iletmek için de kullanılır. Tam havuz kuyruğu yöneticilerinin, birbirlerini işaret eden küme gönderen kanalları vardır. Küme durumu değişikliklerini birbirine iletmek için bunları kullanırlar. Bir kuyruk yöneticisinin CLUSSDR kanal tanımının gösterdiği tam havuz çok önemli değildir. İlk iletişim kurulduktan sonra, kuyruk yöneticisinin küme bilgilerini her tam havuza ve iletileri her kuyruk yöneticisine gönderebilmesi için küme kuyruk yöneticisi nesneleri otomatik olarak gerektiği şekilde tanımlanır.
Küme-günlük nesnesi	Küme alıcısı (CLUSRCVR) kanal tanımlaması, küme kuyruğu yöneticisinin kümedeki diğer kuyruk yöneticilerinden ileti alabileceği bir kanalın alma sonunu tanımlar. Bir küme-alıcı kanalı, havuz için hedeflenmiş küme bilgileri hakkında bilgi de taşıyabilir. Kuyruk yöneticisi, küme-alıcı kanalını tanımlayarak, diğer küme kuyruğu yöneticilerine ileti alabileceğini belirtir. Her küme kuyruğu yöneticisi için en az bir küme alıcı kanalı gerekir.

Her bir kanal için, kanalın her bir ucu için bir kanal tanımına sahip olmak üzere her iki ucu da tanımlamanız gerekir. Kanalın iki ucu uyumlu tipler olmalıdır.

Aşağıdaki kanal tanımı birleşimlerine sahip olabilirsiniz:

- Gönderen-Alıcı
- Sunucu-Alıcı
- İstekte Bulunan-Sunucu
- İstekçi-Gönderen (geri arama)
- Küme-gönderen-Küme-alıcı

İleti kanalı aracıları

Yarattığınız her kanal tanımlaması belirli bir kuyruk yöneticisine aittir. Bir kuyruk yöneticisinin aynı ya da farklı tiplerde birden çok kanalı olabilir. Kanalın her ucunda bir program vardır, ileti kanalı aracısı (MCA). Kanalın bir ucunda, arayan MCA iletileri iletim kuyruğundan alır ve kanal üzerinden gönderir. Kanalın diğer ucunda, yanıt veren MCA iletileri alır ve uzak kuyruk yöneticisine teslim eder.

Çağıran MCA, bir gönderen, sunucu ya da istekte bulunan kanalla ilişkilendirilebilir. Bir 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	Yanıt Verene çağıran	Evet	Alıcı
Sunucu	Hayır	Yanıt Verene çağıran	Evet	Alıcı
Sunucu	Hayır	Yanıt Verene çağıran	Evet	İsteyen
İsteyen	Hayır	Arayanıta Yanıt Veren	Evet	Sunucu
İsteyen	Evet	Arayanıta Yanıt Veren	Evet	Gönderen

MQI kanalları

MQI kanalları aşağıdaki tiplerden biri olabilir:

MQI kanal tipi	**Açıklama**
Sunucu bağlantısı	Sunucu bağlantısı kanalı, bir IBM MQ istemcisini IBM MQ sunucusuna bağlamak için kullanılan iki yönlü bir MQI kanalıdır. Sunucu bağlantısı kanalı, kanalın sunucu ucudur.
İstemci bağlantısı	İstemci bağlantı kanalı, bir IBM MQ istemcisini IBM MQ sunucusuna bağlamak için kullanılan iki yönlü MQI kanalıdır. IBM MQ Explorer , uzak kuyruk yöneticilerine bağlanmak için istemci bağlantılarını da kullanır. İstemci bağlantı kanalı, kanalın istemci ucudur. Bir istemci-bağlantı kanalı yarattığınızda, kuyruk yöneticisini barındıran bilgisayarda bir dosya yaratılır. Daha sonra istemci-bağlantı dosyasını IBM MQ istemci bilgisayarına kopyalamanız gerekir.

Multi *Birden çok iş parçacığı desteği-veri bağlantısı*

İsteğe bağlı olarak, bir ileti kanalı aracısının (MCA) birden çok iş parçacığı kullanarak ileti aktarmasına izin verebilirsiniz. *Pipelining*adı verilen bu işlem, MCA ' nın daha az bekleme durumuyla iletileri daha verimli bir şekilde aktarmasını sağlar ve bu da kanal performansını artırır. Her MCA en çok iki iş parçacığıyla sınırlıdır.

qm.ini dosyasındaki *PipeLineLength* değiştirgesiyle veri bağlantısını denetliyorsunuz. Bu parametre Kanallar kısmınaeklenir.

Not: Boru hattı yalnızca TCP/IP kanalları için geçerlidir.

Boruhattı kullandığınızda, kanalın her iki ucundaki kuyruk yöneticileri 1 'den büyük bir *PipeLineUzunluğu* olacak şekilde yapılandırılmalıdır.

Kanal çıkışında dikkat edilmesi gereken noktalar

Veri bağlantısı bazı çıkış programlarının başarısız olmasına neden olabilir, çünkü:

- Çıkışlar dizisel olarak çağrılmayabilir.
- Çıkışlar farklı iş parçacıklarından alternatif olarak çağrılabilir.

Borulama kullanmadan önce çıkış programlarınızın tasarımını denetleyin:

- Çıkışlar, infazlarının her aşamasında yeniden girişlenmelidir.
- MQI çağrılarını kullandığınızda, çıkış farklı iş parçacıklarından çağrıldığında aynı MQI tanıtıcısını kullanamayacağınız unutmayın.

Bir kuyruğu açan ve çıkışın sonraki tüm çağrılarında MQPUT çağrıları için tanıtıcısını kullanan bir ileti çıkışını düşünün. Çıkış farklı iş parçacıklarından çağrıldığından, ardışık işlem kipinde bu işlem başarısız olur. Bu hatayı önlemek için, her iş parçacığı için bir kuyruk tanıtıcısı tutun ve çıkışın her çağrılışında iş parçacığı tanıtıcısını denetleyin.

İletişim Ürünleri

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

Bağlantının hem IBM MQ MQI client hem de sunucu uçlarında bir kanal tanımı oluşturulmalıdır. Kanal tanımlamalarının nasıl yaratılacağı MQI kanallarının tanımlanmasıbaşlığı altında açıklanmıştır.

Olası iletim protokolleri aşağıdaki çizelgede gösterilmiştir:

Çizelge 1. MQI kanalları için iletim protokolleri

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

Not:

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

İletim protokolleri- IBM MQ MQI client ve sunucu platformlarının birleşimi , bu iletim protokollerini kullanarak IBM MQ MQI client ve sunucu platformlarının olası birleşimlerini gösterir.

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

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

Başarımla ilgili önemli noktalar

Kullandığınız iletim protokolü, IBM MQ istemcisi ve sunucu sisteminin başarımını etkileyebilir. İletim yavaş olduğu belirli durumlarda, IBM MQ kanal sıkıştırmasını kullanabilirsiniz.

IBM MQ nesnelerini adlandırma

IBM MQ nesneleri için benimsenen adlandırma kuralı nesneye bağlıdır. IBM MQ ile kullandığınız makinelerin ve kullanıcı kimliklerinin adı da bazı adlandırma kısıtlamalarına tabidir.

Bir kuyruk yöneticisinin her bir eşgörünümü kendi adıyla tanınır. Bir kuyruk yöneticisinin belirli bir iletinin gönderildiği hedef kuyruk yöneticisini belirsiz bir şekilde saptayabilmesi için, bu adın birbirine bağlı kuyruk yöneticileri ağı içinde benzersiz olması gerekir.

Diğer nesne tipleri için, her nesneyle ilişkilendirilmiş bir ad vardır ve bu ad bu nesneye gönderme yapabilir. Bu adlar, bir kuyruk yöneticisi ve nesne tipi içinde benzersiz olmalıdır. Örneğin, aynı ada sahip bir kuyruğunuz ve bir süreciniz olabilir, ancak aynı ada sahip iki kuyruğunuz olamaz.

IBM MQiçinde, en fazla 20 karakter içeren *kanallar* dışında, adlar en fazla 48 karakterden oluşabilir. IBM MQ nesnelerini adlandırma hakkında daha fazla bilgi için bkz. “IBM MQ nesnelerini adlandırma kuralları” sayfa 35.

IBM MQ ile kullandığınız makinelerin ve kullanıcı kimliklerinin adı da bazı adlandırma kısıtlamalarına tabidir:

- Makine adının boşluk içermediğinden emin olun. IBM MQ , boşluk içeren makine adlarını desteklemez. Böyle bir makineye IBM MQ kurarsanız, kuyruk yöneticisi yaratamamazsınız.
- IBM MQ yetkileri için, kullanıcı kimliklerinin ve grupların adları 20 karakterden uzun olmamalıdır (boşluklara izin verilmez).
- 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 38
Her IBM MQ kuyruk yöneticisi, kuyruk, süreç tanımı, ad listesi, kanal, istemci bağlantısı kanalı, dinleyici, hizmet ve kimlik doğrulama bilgileri nesnesi bir dosya tarafından temsil edilir. Nesne adlarının geçerli olması gerekmediği için, kuyruk yöneticisi nesne adını gerektiğinde geçerli bir dosya adına dönüştürür.

İlgili başvurular

“IBM MQ nesnelerini adlandırma kuralları” sayfa 35
IBM MQ nesne adlarının uzunluk üst sınırı vardır ve bunlar büyük ve küçük harfe duyarlıdır. Her nesne tipi için tüm karakterler desteklenmez ve birçok nesnenin adların benzersizliğine ilişkin kuralları vardır.

IBM MQ nesnelerini adlandırma kuralları

IBM MQ nesne adlarının uzunluk üst sınırı vardır ve bunlar büyük ve küçük harfe duyarlıdır. Her nesne tipi için tüm karakterler desteklenmez ve birçok nesnenin adların benzersizliğine ilişkin kuralları vardır.

Birçok farklı IBM MQ nesnesi tipi vardır ve her tipteki nesneler ayrı nesne ad alanlarında bulundukları için aynı ada sahip olabilir: Örneğin, bir yerel kuyruk ve bir gönderen kanalı aynı ada sahip olabilir. Ancak, bir nesne aynı ad alanındaki başka bir nesneyle aynı ada sahip olamaz: Örneğin, bir yerel kuyruk model kuyruğuyla aynı ada sahip olamaz ve gönderen kanal, alıcı kanalla 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 adlarının 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 karakterden oluşmalı ve yalnızca büyük harfli karakterler ve sayısal karakterler içermelidir.
2. Kanal nesne adlarının ve istemci bağlantı kanalı adlarının uzunluk üst sınırı 20 karakterdir. Kanallar hakkında daha fazla bilgi için Kanalların tanımlanması başlıklı konuya bakın.
3. Konu dizgileri en çok 10240 byte olabilir. Tüm IBM MQ nesne adları büyük ve küçük harfe duyarlıdır.
4. Abonelik adları en çok 10240 bayt 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	Sınırlamalar
Büyük A-Z	• Yok

Karakterler	Sınırlamalar
Küçük harf a-z	• MQSC komut dosyalarında, küçük harfli adlar tek tırnak içine alınmalıdır. Bu, küçük harflerin büyük harfe katlanmasını önler. • EBCDIC Katakana kullanan sistemler, nesne adlarında küçük harfli a-z karakterleri kullanamazlar. • z/OS z/OS sistemlerinde küçük harfli karakterler kullanılırken kısıtlamalar olabilir; örneğin, kuyruk yöneticisi adları küçük harfli karakterler içeremez. • IBM i IBM i sistemlerinde CL komutlarını kullanırken, küçük harfli adlar tek tırnak içine alınmalıdır. Bu, küçük harflerin büyük harfe katlanmasını önler.
0-9 arası sayısal	• Yok
Nokta (.)	• Yok
Alt çizgi (_)	• Multi Yok • z/OS IBM MQ for z/OS işlemleri ve denetim panoları tarafından işlenemeyeceği için baştaki ya da sondaki alt çizgilere sahip adları kullanmaktan kaçının.
Eğik Çizgi (/)	• Windows Windows sistemlerinde, kuyruk yöneticisi adının ilk karakteri eğik çizgi olamaz. • IBM i IBM i sistemlerinde CL komutlarını kullanırken, eğik çizgi içeren adlar tek tırnak içine alınmalıdır. • z/OS Yok
Yüzde işareti (%)	• ALW Yok • z/OS IBM MQ for z/OSiçin dış güvenlik yöneticisi olarak RACF kullanıyorsanız, adlar RACF genel tanıtımları kullanıldığında güvenlik denetimlerine dahil edilmediğinden nesne adlarında% kullanmayın. • IBM i IBM i sistemlerinde CL komutlarını kullanırken, yüzde işareti içeren adlar tek tırnak işareti içine alınmalıdır.

Nesne adlarındaki karakterlerle ilgili bazı genel kurallar da vardır:

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

Kuyruk adları

Bir kuyruğun adı iki bölümden oluşur:

- Kuyruk yöneticisinin adı
- Kuyruk yöneticisinin tanıdığı şekilde, kuyruğun yerel adı

Kuyruk adının her bir bölümü 48 karakter uzunluğundadır.

Yerel bir kuyruğa gönderme yapmak için, kuyruk yöneticisinin adını atlayabilirsiniz (yerine boş karakterler koyarak ya da başında boş karakter kullanarak). Ancak, IBM MQ tarafından bir programa döndürülen tüm kuyruk adları, kuyruk yöneticisinin adını içerir.

z/OS Kuyruk paylaşım grubundaki herhangi bir kuyruk yöneticisinin erişebileceği paylaşılan bir kuyruk, aynı kuyruk paylaşım grubundaki paylaşılmayan herhangi bir yerel kuyrukla aynı ada sahip olamaz. Bu kısıtlama, bir uygulamanın yerel bir kuyruğu açmayı amaçladığında ya da tam tersi durumda, paylaşılan bir kuyruğu yanlışlıkla açma olasılığını ortadan kaldırır. Paylaşılan kuyruklar ve kuyruk paylaşım grupları yalnızca IBM MQ for z/OSüzerinde kullanılabilir.

Uzak bir kuyruğa gönderme yapmak için, programın kuyruk yöneticisinin adını tam kuyruk adında içermesi ya da uzak kuyruk için yerel bir tanımlama bulunması gerekir.

Bir uygulama bir kuyruk adı kullandığında, bu ad yerel bir kuyruğun adı (ya da bir kuyruğun diğer adı) ya da uzak bir kuyruğun yerel tanımının adı olabilir; ancak, kuyruktan ileti alması gerekmedikçe (kuyruğun yerel olması gerektiğinde), uygulamanın bunu bilmesi gerekmez. Uygulama kuyruk nesnesini açtığında, MQOPEN çağrısı sonraki işlemleri hangi kuyrukta gerçekleştireceğini saptamak için bir ad çözme işlevi gerçekleştirir. Bunun anlamı, uygulamanın kuyruk yöneticilerinden oluşan bir ağdaki belirli konumlarda tanımlanmakta olan belirli kuyruklara yerleşik bir bağımlılığı olmamasıdır. Bu nedenle, bir sistem denetimcisi ağdaki kuyrukların yerini değiştirir ve tanımlamalarını değiştirirse, bu kuyrukları kullanan uygulamaların değiştirilmesi gerekmez.

Ayrılmış nesne adları

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

z/OS IBM MQ for z/OSüzerinde, bağlaşım olanağı uygulama yapısı adı CSQSYSAPPL ayrılmıştır.

İlgili kavramlar

AIX, Linux, and Windows üzerindeki kuruluş adı

IBM MQ dosya adları

Her IBM MQ kuyruk yöneticisi, kuyruk, süreç tanımı, ad listesi, kanal, istemci bağlantısı kanalı, dinleyici, hizmet ve kimlik doğrulama bilgileri nesnesi bir dosya tarafından temsil edilir. Nesne adlarının geçerli olması gerekmediği için, kuyruk yöneticisi nesne adını gerektiğinde geçerli bir dosya adına dönüştürür.

Bir kuyruk yöneticisi dizininin varsayılan yolu şöyledir:

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

Varsa, önek IBM MQ Gezgini 'ndeki IBM MQ özellikler sayfası kullanılarak değiştirilebilir, tersi durumda `mqs.ini` yapılandırma dosyasını el ile düzenleyin.

- Kuyruk yöneticisi adı geçerli bir dizin adına dönüştürülür. Örneğin, kuyruk yöneticisi:

```
queue.manager
```

şu şekilde temsil edilir:

```
queue!manager
```

Bu sürece *ad dönüştürme*denir.

IBM MQiçinde, bir kuyruk yöneticisine en çok 48 karakter içeren bir ad verebilirsiniz.

Örneğin, bir kuyruk yöneticisine şu adı verebilirdiniz:

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

Ancak, her kuyruk yöneticisi bir dosya tarafından temsil edilir ve bir dosya adının uzunluk üst sınırına ve adda kullanılabilecek karakterlere ilişkin sınırlamalar vardır. Sonuç olarak, nesneleri temsil eden dosyaların adları, dosya sisteminin gereksinimlerini karşılayacak şekilde otomatik olarak dönüştürülür.

Bir kuyruk yöneticisi adının dönüştürülmesini düzenleyen kurallar şunlardır:

1. Tek tek karakterleri dönüştür:
   - -İlk. -Hadi!
   - Kaynak/& ye
2. Ad hala geçerli değilse:
   a. Sekiz karaktere kadar kısaltın
   b. Üç karakterli sayısal sonek ekle

Örneğin, varsayılan önek ve `queue.manager`adlı bir kuyruk yöneticisi varsayılarak:

- Windows NTFS ya da FAT32ile Windows ' de kuyruk yöneticisi adı şöyle olur:

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

- Windows Windows ' da FAT ile birlikte kuyruk yöneticisi adı şöyle olur:

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

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

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

Dönüştürme algoritması, yalnızca büyük ve küçük harfe duyarlı olmayan dosya sistemlerinde farklı olan adları da ayırır.

Nesne adı dönüşümü

Nesne adlarının geçerli dosya sistemi adları olması gerekmez. Nesne adlarınızı dönüştürmeniz gerekebilir. Kullanılan yöntem, kuyruk yöneticisi adları için kullanılandan 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 bilgileri, 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)

İlgili bilgiler

AllQueue mqs.ini dosyasının yöneticiler kısmı

IBM i IBM i üzerindeki nesne adları

Bir kuyruk yöneticisinin benzersiz adı olan ilişkili bir kuyruk yöneticisi kitaplığı vardır. IBM i Integrated File System (IFS) gereksinimlerini karşılamak için kuyruk yöneticisi adlarının ve nesne adlarının dönüştürülmesi gerekebilir.

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

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

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

IBM MQ IFS dizinleri ve dosyaları

IBM i Integrated File System (IFS), verileri depolamak için IBM MQ tarafından yaygın olarak kullanılır. IFS hakkında daha fazla bilgi için bkz. *Integrated File System Giriş*.

Her IBM MQ nesnesi (örneğin, bir kanal ya da kuyruk yöneticisi) bir dosya tarafından temsil edilir. Nesne adlarının geçerli olması gerekmediği için, kuyruk yöneticisi nesne adını gerektiğinde geçerli bir dosya adına dönüştürür.

Kuyruk yöneticisi dizininin yolu aşağıdakilerden oluşur:

- `qm.ini`kuyruk yöneticisi yapılanış dosyasında tanımlanan bir önek. Varsayılan önek `/QIBM/UserData/mqm`' dir.
- Bir hazır bilgi, `qmgrs`.
- Geçerli bir dizin adına dönüştürülen kuyruk yöneticisi adı olan kodlanmış kuyruk yöneticisi adı. Örneğin, `queue/manager` kuyruk yöneticisi `queue&managerile` gösterilir.

Bu işleme ad dönüşümü adı verilir.

IFS kuyruk yöneticisi adı dönüşümü

IBM MQiçinde, bir kuyruk yöneticisine en çok 48 karakter içeren bir ad verebilirsiniz.

Örneğin, bir kuyruk yöneticisine `QUEUE/MANAGER/ACCOUNTING/SERVICES`adını verebilirsiniz. Her kuyruk yöneticisi için bir kitaplık yaratılmasıyla aynı şekilde, her kuyruk yöneticisi de bir dosya ile gösterilir. EBCDIC ' deki değişken kod noktaları nedeniyle, adda kullanılabilecek karakterlere ilişkin sınırlamalar vardır. Sonuç olarak, nesneleri temsil eden IFS dosyalarının adları, dosya sisteminin gereksinimlerini karşılayacak şekilde otomatik olarak dönüştürülür.

queue/manageradlı bir kuyruk yöneticisi örneğini kullanarak, / karakterini &karakterine dönüştüren ve varsayılan öneki varsayarak IBM MQ for IBM i içindeki kuyruk yöneticisi adı /QIBM/UserData/mqm/qmgrs/queue&managerolur.

Nesne adı dönüşümü

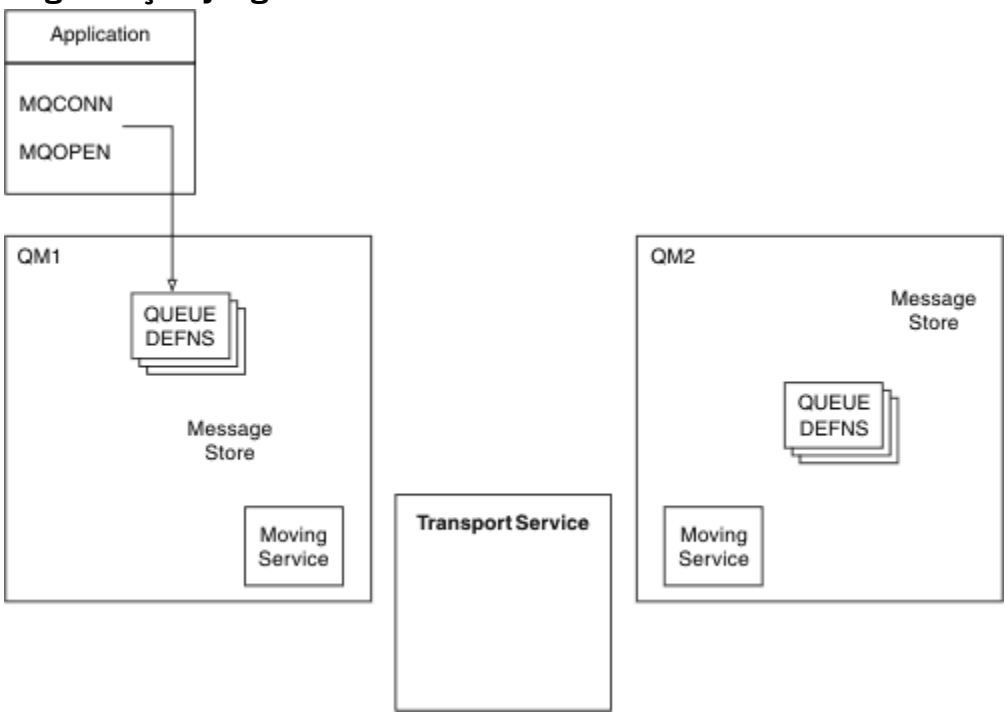
Nesne adlarının geçerli dosya sistemi adları olması gerekmediği için, nesne adlarının dönüştürülmesi gerekebilir. Kuyruk yöneticisi adları için kullanılan yöntem farklıdır; her makine için yalnızca birkaç kuyruk yöneticisi adı olsa da, her kuyruk yöneticisi için çok sayıda başka nesne olabilir. Dosya sisteminde yalnızca süreç tanımlamaları, kuyruklar ve ad alanları gösterilir; kanallar bu konulardan etkilenmez.

Dönüştürme işlemi tarafından yeni bir ad oluşturulduğunda, özgün nesne adıyla basit bir ilişki yoktur. IBM MQ nesnelerine ilişkin dönüştürülen adları görüntülemek için DSPMQMOBJN komutunu kullanabilirsiniz.

Dağıtılmış kuyruğa alma ve kümeler

Dağıtılmış kuyruğa alma, bir kuyruk yöneticisinden diğerine ileti gönderilmesi anlamına gelir. Alıcı kuyruk yöneticisi aynı makinede ya da başka bir makinede olabilir; yakınlarda ya da dünyanın diğer tarafında. Yerel kuyruk yöneticisiyle aynı altyapıda ya da IBM MQtarafından desteklenen herhangi bir altyapıda çalışabilir. Dağıtılmış bir kuyruğa alma ortamındaki tüm bağlantıları el ile tanımlayabilir ya da bir küme yaratabilir ve IBM MQ ' un bağlantı ayrıntılarının çoğunu sizin için tanımlamasına izin verebilirsiniz.

Dağıtılmış kuyruğa alma



Şekil 4. Dağıtılmış kuyruğa alma bileşenlerine genel bakış

Önceki şekilde:

- Bir uygulama, kuyruk yöneticisine bağlanmak için MQCONN çağrıyı kullanır. Daha sonra uygulama, kuyruğa ileti yerleştirebilmesi için bir kuyruğu açmak üzere MQOPEN çağrıyı kullanır.
- Her kuyruk yöneticisinin her bir kuyruğu için bir tanımlaması vardır. *Yerel kuyruk* tanımlamaları (bu kuyruk yöneticisi tarafından barındırılan) ve *uzak kuyruk* tanımlamaları (diğer kuyruk yöneticileri tarafından barındırılan) olabilir.
- İletiler uzak bir kuyruğa gönderilecekse, yerel kuyruk yöneticisi bunları uzak kuyruk yöneticisine iletilebilinceye kadar ileti deposunda saklayan bir *iletim kuyruğunda*tutar.

- Her kuyruk yöneticisi, kuyruk yöneticisinin diğer kuyruk yöneticileriyle iletişim kurmak için kullandığı *hareketli hizmet*olarak bilinen iletişim yazılımını içerir.
- *İletim hizmeti* kuyruk yöneticisinden bağımsızdır ve aşağıdakilerden biri olabilir (altyapıya bağlı olarak):
  - Sistem Ağ Mimarisi Gelişmiş Programı-Program İletişimi (SNA APPC)
  - İletim Denetimi İletişim Kuralı/Internet Protocol (TCP/IP)
  - Ağ Temel Giriş/Çıkış Sistemi (NetBIOS)
  - Sıralı Paket Değiş tokuşu (SPX)

İleti göndermek için gereken bileşenler

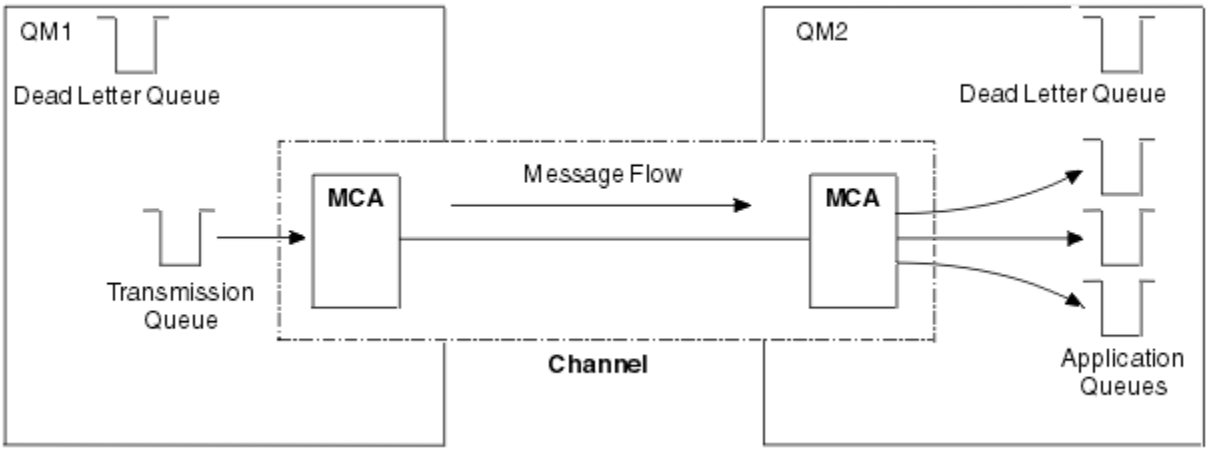
Uzak kuyruk yöneticisine bir ileti gönderilecek olursa, yerel kuyruk yöneticisi *iletim kuyruğu* ve *kanal*tanımlamalarına gereksinim duyar. Kanal, iki kuyruk yöneticisi arasındaki tek yönlü iletişim bağlantısıdır. Uzak kuyruk yöneticisinde istediğiniz sayıda kuyruğa gönderilecek iletileri taşıyabilir.

Bir kanalın her ucunun ayrı bir tanımı vardır; bu tanımlama, örneğin, gönderme sonu ya da alma sonu olarak tanımlanır. Basit bir kanal, yerel kuyruk yöneticisinde bir *gönderen* kanal tanımlaması ve uzak kuyruk yöneticisinde bir *alıcı* kanal tanımlamasından oluşur. Bu iki tanımlama aynı ada sahip olmalıdır ve birlikte bir kanal oluştururlar.

İletilerin gönderilmesini ve alınmasını işleyen yazılıma *Message Channel Agent* (MCA) adı verilir. Bir kanalın her ucunda bir *ileti kanalı aracısı* (MCA) vardır.

Her kuyruk yöneticisinin bir *gitmeyen ileti kuyruğu* olmalıdır (*teslim edilmeyen ileti kuyruğu* olarak da bilinir). İletiler, hedeflerine teslim edilemezlerse bu kuyruğa yerleştirilir.

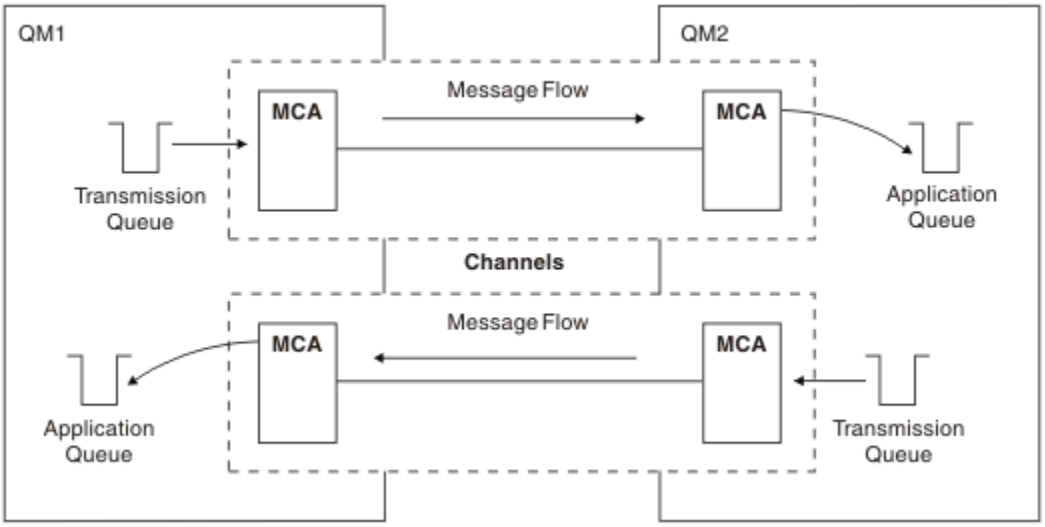
Aşağıdaki şekilde kuyruk yöneticileri, iletim kuyrukları, kanallar ve MCA ' lar arasındaki ilişki gösterilmektedir:



Şekil 5. İleti gönderilmesi

İleti döndürmek için gereken bileşenler

Uygulamanız uzak kuyruk yöneticisinden ileti döndürülmesini gerektiriyorsa, aşağıdaki şekilde gösterildiği gibi, kuyruk yöneticileri arasında ters yönde çalışmak için başka bir kanal tanımlamanız gerekir:



Şekil 6. Her iki yönde de ileti gönderilmesi

Kümeler

Dağıtılmış bir kuyruğa alma ortamındaki tüm bağlantıları el ile tanımlamak yerine, bir kümedeki kuyruk yöneticileri kümesini gruplayabilirsiniz. Bunu yaptığınızda, kuyruk yöneticileri, her hedef için belirtik kanal tanımlarına, uzak kuyruk tanımlarına ya da iletim kuyruklarına gerek kalmadan, anasistem kuyruklarını kümedeki diğer kuyruk yöneticilerinin kullanımına sunabilir. Bir kümedeki her kuyruk yöneticisinin, kümedeki başka bir kuyruk yöneticisine ileti ileten tek bir iletim kuyruğu vardır. Her kuyruk yöneticisi için yalnızca bir küme alıcı kanalı ve bir küme gönderen kanalı tanımlamanız gerekir; ek kanallar küme tarafından otomatik olarak yönetilir.

IBM MQ istemcisi, bir kümenin parçası olan bir kuyruk yöneticisine, diğer herhangi bir kuyruk yöneticisine bağlanabileceği gibi bağlanabilir. El ile yapılandırılmış dağıtılmış kuyruğa alma sırasında olduğu gibi, MQPUT çağrısı, herhangi bir kuyruk yöneticisindeki bir kuyruğa ileti koymak için kullanılır. Yerel bir kuyruktan ileti almak için MQGET çağrısı kullanılır.

Kümeleri destekleyen platformlardaki kuyruk yöneticileri bir kümenin parçası olmak zorunda değildir. Kümeleri kullanarak dağıtılmış kuyruğa alma işlemini el ile yapılandırmaya devam edebilir ya da bunun yerine el ile yapılandırmaya devam edebilirsiniz.

Kümeleri kullanmanın avantajları

Kümeleme iki temel avantaj sağlar:

- Kümeler, genellikle kanallar, iletim kuyrukları ve uzak kuyruklar için birçok nesne tanımlaması gerektiren IBM MQ ağlarının yönetimini basitleştirir. Bu durum özellikle, birçok kuyruk yöneticisinin birbirine bağlanması gereken büyük, potansiyel olarak değişen ağlarda geçerlidir. Bu mimarinin yapılandırılması ve etkin olarak sürdürülmesi özellikle zordur.
- Kümeler, ileti trafiğinin iş yükünü kümedeki kuyruklara ve kuyruk yöneticilerine dağıtmak için kullanılabilir. Bu dağıtım, tek bir kuyruğun ileti iş yükünün, birden çok kuyruk yöneticisindeki bu kuyruğun eşdeğer eşgörünümlerine dağıtılmasını sağlar. İş yükünün bu dağılımı, sistem hatalarına daha fazla dayanıklılık sağlamak ve bir sistemdeki özellikle etkin ileti akışlarının ölçekleme performansını artırmak için kullanılabilir. Böyle bir ortamda, dağıtılmış kuyrukların her bir örneğinin iletileri işleyen uygulamaları vardır. Daha fazla bilgi için İş yükü yönetimi için kümeleri kullanmabaşlıklı konuya bakın.

İletilerin bir kümede nasıl yönlendirildiği

Bir kümeyi, vicdani sistem yöneticisi tarafından tutulan kuyruk yöneticilerinin ağı olarak düşünebilirsiniz. Bir küme kuyruğu tanımladığınızda, sistem denetimcisi diğer kuyruk yöneticilerine ilişkin uzak kuyruk tanımlamalarını otomatik olarak yaratır.

IBM MQ , kümedeki her bir kuyruk yöneticisinde bir iletim kuyruğu sağladığından, iletim kuyruğu tanımları yapmanıza gerek yoktur. Bu tek iletim kuyruğu, iletileri kümedeki diğer herhangi bir kuyruk yöneticisine taşımak için kullanılabilir. Tek bir iletim kuyruğu kullanmakla sınırlı değilsiniz. Bir kuyruk yöneticisi, bir kümedeki her kuyruk yöneticisine giden iletileri ayırmak için birden çok iletim kuyruğu kullanabilir. Genellikle, bir kuyruk yöneticisi tek bir küme iletim kuyruğu kullanır. Bir kuyruk yöneticisinin kümedeki her kuyruk yöneticisi için farklı bir küme iletim kuyruğu kullanması için DEFCLXQkuyruk yöneticisi özniteliğini değiştirebilirsiniz. Küme iletim kuyruklarını el ile de tanımlayabilirsiniz.

Bir kümeye katılan tüm kuyruk yöneticileri bu şekilde çalışmayı kabul eder. Kendileri ve barındıkları kuyruklar hakkında bilgi gönderirler ve kümenin diğer üyeleri hakkında bilgi alırlar.

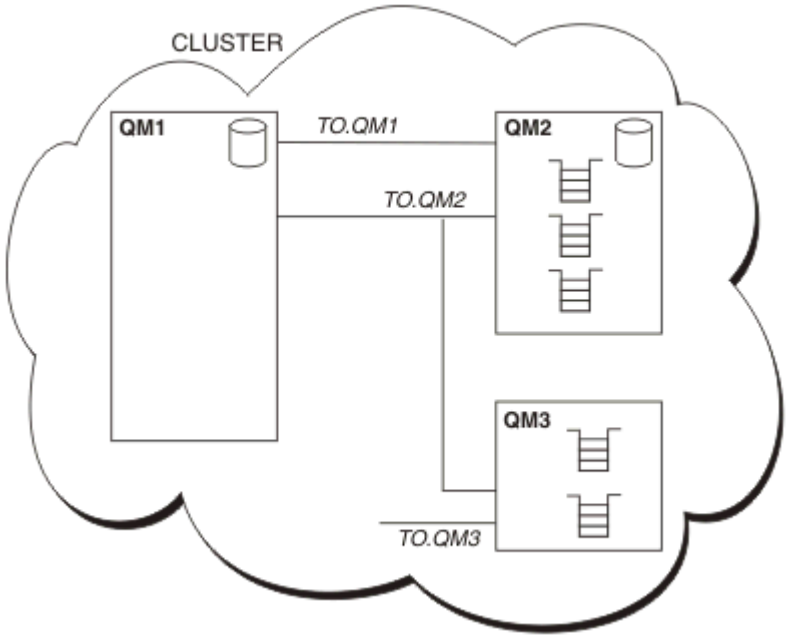
Bir kuyruk yöneticisi kullanılamaz duruma geldiğinde hiçbir bilginin kaybolmadığından emin olmak için, kümede *tam havuz*olarak işlev görecek iki kuyruk yöneticisi belirtirsiniz. Bu kuyruk yöneticileri, kümedeki tüm kuyruk yöneticileri ve kuyrukları hakkında tam bir bilgi kümesi saklar. Kümedeki diğer tüm kuyruk yöneticileri yalnızca, ileti değiş tokuşunda bulundukları kuyruk yöneticileri ve kuyruklarıyla ilgili bilgileri saklar. Bu kuyruk yöneticileri *kısmi havuzlar*olarak bilinir. Daha fazla bilgi için “Küme havuzu” sayfa 54 başlıklı konuya bakın.

Bir kümenin parçası olmak için kuyruk yöneticisinin iki kanalı olmalıdır; küme gönderen kanal ve küme alıcı kanal:

- Küme gönderen kanal, gönderen kanal gibi bir iletişim kanalıdır. Zaten kümenin üyesi olan tam bir havuza bağlanmak için bir kuyruk yöneticisinde bir küme gönderen kanalı el ile oluşturmanız gerekir.
- Bir küme-alıcı kanalı, alıcı kanalı gibi bir iletişim kanalıdır. El ile bir küme alıcı kanalı yaratmanız gerekir. Kanal, kuyruk yöneticisinin küme iletişimini alması için bir mekanizma görevi görür.

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

Aşağıdaki şekil, CLUSTER adı verilen bir kümenin bileşenlerini göstermektedir:



Şekil 7. Kuyruk yöneticileri kümesi

- CLUSTER üç kuyruk yöneticisi içerir: QM1, QM2ve QM3.
- QM1 ve QM2 anasistem, kümedeki kuyruk yöneticileri ve kuyruklarıyla ilgili bilgilerin tam havuzlarını içerir.
- QM2 ve QM3 anasistemleri, kümedeki diğer herhangi bir kuyruk yöneticisinin erişebileceği bazı küme kuyrukları.
- Her kuyruk yöneticisinin, ileti alabileceği TO.qmgr adlı bir küme alıcı kanalı vardır.

- Her kuyruk yöneticisinin, havuz kuyruğu yöneticilerinden birine bilgi gönderebildiği bir küme gönderen kanalı da vardır.
- QM1 ve QM3 , QM2 ve QM2 ' deki havuza QM1adresinden gönderir.

Dağıtılmış kuyruğa alma bileşenleri

Dağıtılmış kuyruğa alma bileşenleri, ileti kanalları, ileti kanalı aracıları, iletim kuyrukları, kanal başlatıcıları ve dinleyiciler ve kanal çıkış programlarıdır. Bir ileti kanalının her ucunun tanımı birkaç tipte olabilir.

İleti kanalları, bir kuyruk yöneticisinden diğerine ileti taşıyan kanallardır. İleti kanallarını MQI kanallarıyla karıştırmayın. İki tip MQI kanalı vardır: sunucu bağlantısı (SVRCONN) ve istemci bağlantısı (CLNTCONN). Daha fazla bilgi için bkz. Kanallar.

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

- Gönderen (SDR)
- Alıcı (RCVR)
- Sunucu (SVR)
- İstekte Bulunan (RQSTR)
- Küme gönderen (CLUSSDR)
- Küme alıcısı (CLUSRCVR)

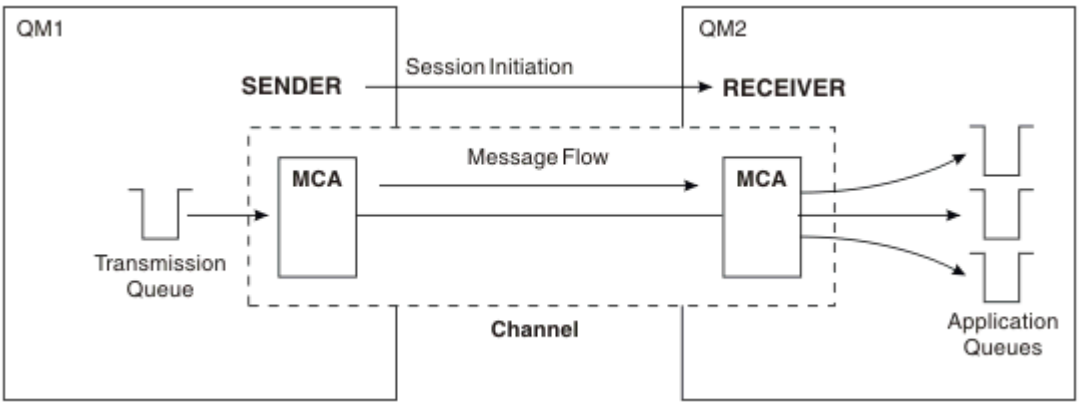
Bir ileti kanalı, bir uçta tanımlanan bu tiplerden biri ve diğer uçta uyumlu bir tip kullanılarak tanımlanır. Olası birleşimler şunlardır:

- Gönderen-alıcı
- İstekte bulunan-sunucu
- İstekçi-gönderen (geri arama)
- Sunucu-günlük nesnesi
- Küme gönderen-küme alıcısı

Gönderici-alıcı kanalı yaratılmasına ilişkin ayrıntılı yönergeler Kanalların tanımlanmasıbaşlığı altında yer alır. Gönderen alıcı kanallarını ayarlamak için gereken parametrelere ilişkin örnekler için, altyapınız için geçerli olan Örnek yapılandırma bilgileri konusuna bakın. Herhangi bir tipte kanal tanımlamak için gereken parametreler için bkz. DEFINE CHANNEL.

Gönderen-alıcı kanalları

Bir sistemdeki bir gönderen, diğer sisteme ileti gönderebilmesi için kanalı başlatır. Gönderen, kanalın diğer ucundaki alıcının başlamasını ister. Gönderen, iletim kuyruğundan alıcıya ileti gönderir. Alıcı iletileri hedef kuyruğa koyar. Şekil 8 sayfa 45 içinde bu gösterilmektedir.

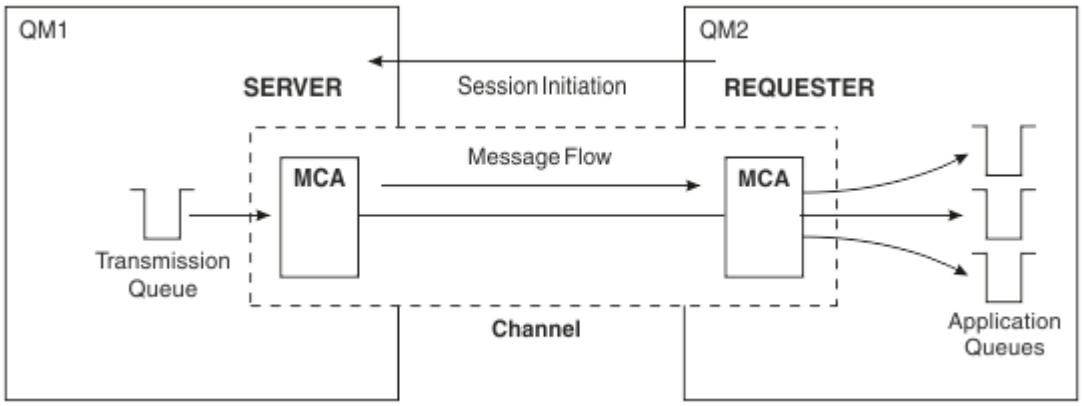


Şekil 8. Bir gönderici-alıcı kanalı

İstekte bulunan sunucu kanalları

Bir sistemdeki istekçi, diğer sistemden ileti alabilecek şekilde kanalı başlatır. İstekte bulunan, sunucunun kanalın diğer ucundan başlatımesini ister. Sunucu, kanal tanımında tanımlanan iletim kuyruğundan istekte bulunana ileti gönderir.

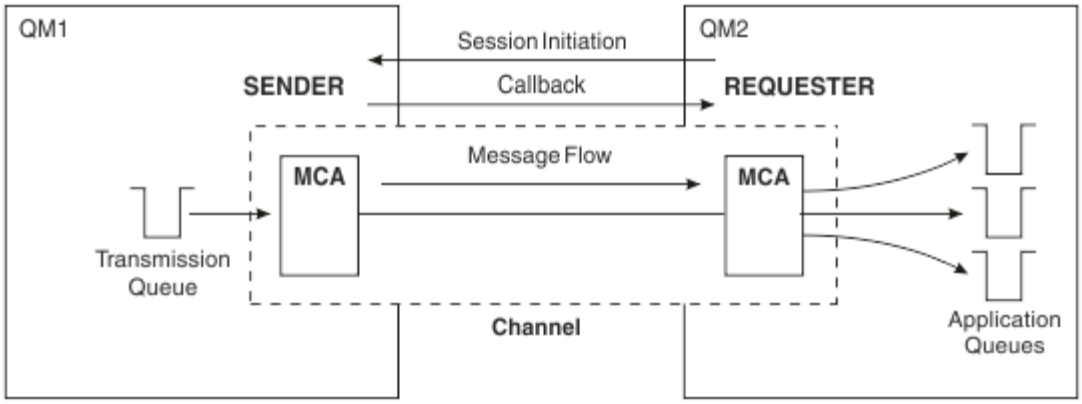
Sunucu kanalı, iletişimi başlatabilir ve istekte bulunana ileti gönderebilir. Bu, yalnızca kanal tanımında belirtilen ortak bağlantı adına sahip sunucu kanalları olan *nitelenmiş açık* sunucular için geçerlidir. Tam olarak nitelenmiş bir sunucu, istekte bulunan tarafından başlatılabilir ya da istekte bulunan ile iletişim başlatabilir.



Şekil 9. İstekçi sunucu kanalı

İstekte bulunan-gönderen kanalları

İstekte bulunan, kanalı başlatır ve gönderen, aramayı sonlandırır. Daha sonra gönderen, kanal tanımındaki bilgilere göre iletişimi yeniden başlatır (*geri arama* olarak bilinir). İletim kuyruğundan istekte bulunana ileti gönderir.



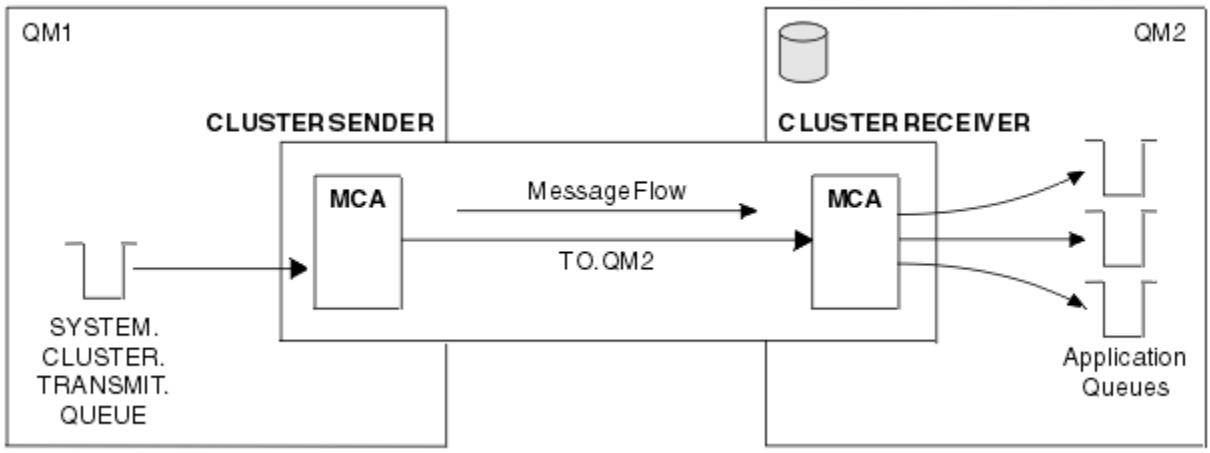
Şekil 10. İstekçi-gönderen kanalı

Sunucu alıcı kanalları

Bu, gönderen-alıcıya benzer, ancak yalnızca kanal tanımında belirtilen ortak bağlantı adına sahip sunucu kanalları olan *nitelenmiş açık* sunucular için geçerlidir. Bağlantının sunucu ucunda kanal başlatma başlatılmalıdır. Bunun şekli, Şekil 8 sayfa 45içindeki şekle benzer.

Küme-gönderen kanalları

Bir kümede, her kuyruk yöneticisinin, küme bilgilerini tam havuz kuyruğu yöneticilerinden birine gönderebildiği bir küme gönderen kanalı vardır. Kuyruk yöneticileri, küme gönderen kanallarında diğer kuyruk yöneticilerine de ileti gönderebilir.



Şekil 11. Küme gönderen kanal

Küme-alıcı kanalları

Bir kümede, her kuyruk yöneticisinin kümeyle ilgili iletileri ve bilgileri alabileceği bir küme alıcı kanalı vardır. Bunun şekli, Şekil 11 sayfa 47içindeki şekle benzer.

Gönderilmeyen ileti kuyrukları

Teslim edilmeyen iletiler kuyruğu (ya da teslim edilmeyen iletiler kuyruğu), iletilerin doğru hedeflerine yöneltilemeyecekleri durumlarda gönderildikleri kuyruktur. Her kuyruk yöneticisinin genellikle bir gönderilmeyen ileti kuyruğu vardır.

Teslim edilmeyen ileti kuyruğu (DLQ), bazen *teslim edilmeyen ileti kuyruğu*olarak da adlandırılır; örneğin, kuyruk var olmadığı için ya da dolu olduğu için hedef kuyruklarına teslim edilemeyen iletiler için bir bekleme kuyruğudur. Veri dönüştürme hataları için, bir kanalın gönderme ucunda, gönderilmeyen harf kuyrukları da kullanılır. Bir ağdaki her kuyruk yöneticisinin, genellikle, daha sonra alınmak üzere doğru hedeflerine teslim edilemeyen iletilerin depolanması için, gönderilmeyen iletiler kuyruğu olarak kullanılacak yerel bir kuyruğu vardır.

İletiler, kuyruk yöneticileri, ileti kanalı aracıları (MCA 'lar) ve uygulamalar tarafından DLQ' ya konabilir. DLQ üzerindeki tüm iletilerin başına *alınmeyen harf üstbilgisi* yapısı (MQDLH) eklenmelidir. MQDLH yapısının *Neden* alanı, iletinin neden DLQ üzerinde olduğunu belirten bir neden kodu içeriyor.

Genellikle her kuyruk yöneticisi için bir gitmeyen ileti kuyruğu tanımlamanız gerekir. Bunu yapmazsanız ve MCA bir ileti koyamazsa, ileti iletim kuyruğuna bırakılır ve kanal durdurulur. Ayrıca, hızlı, kalıcı olmayan iletiler (bkz. Hızlı, kalıcı olmayan iletiler) teslim edilemiyor ve hedef sistemde bir teslim edilmeyen ileti kuyruğu yok, bu iletiler atılıyor.

Ancak, gitmeyen harf kuyruklarının kullanılması iletilerin teslim edilme sırasını etkileyebilir ve bu nedenle bunları kullanmamayı seçebilirsiniz.

İlgili görevler

Gönderilmeyen ileti kuyruklarıyla çalışılması

Teslim edilmemiş iletilerde sorun giderme

İlgili başvurular

runmqdlq (teslim harfli kuyruk işleyicisini çalıştır)

Uzak kuyruk tanımlamaları

Uzak kuyruk tanımlamaları, başka bir kuyruk yöneticisinin iyeliğindeki kuyruklara ilişkin tanımlamalardır.

Uygulamalar iletileri yalnızca yerel kuyruklardan alabilirken, iletileri yerel kuyruklara ya da uzak kuyruklara yerleştirebilirler. Bu nedenle, her bir yerel kuyruğuna ilişkin bir tanımlamanın yanı sıra, bir kuyruk

yöneticisinin *uzak kuyruk tanımlamaları*olabilir. Uzak kuyruk tanımlarının yararı, uzak kuyruğun ya da uzak kuyruk yöneticisinin adını ya da iletim kuyruğunun adını belirlemek zorunda kalmadan bir uygulamanın uzak kuyruğa ileti göndermesini sağlamasıdır. Uzak kuyruk tanımlamaları size konum bağımsızlığı verir.

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

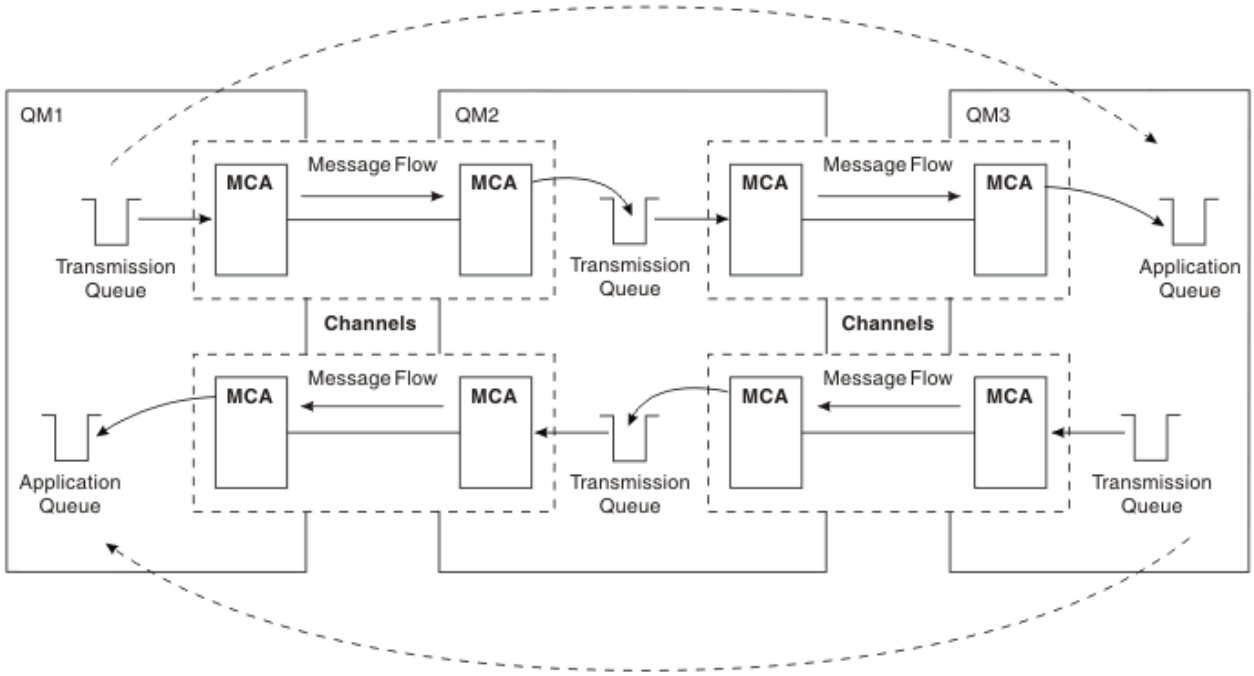
Uzak kuyruk yöneticisine nasıl ulaşacağınızı

Her kaynak ve hedef kuyruk yöneticisi arasında her zaman bir kanal olmayabilir. Çoklu sıçrama, kanalları paylaşma, farklı kanallar kullanma ve kümeleme gibi ikisi arasında bağlantı kurmanın birçok başka yolu vardır.

Çoklu atlama

Kaynak kuyruk yöneticisi ile hedef kuyruk yöneticisi arasında doğrudan iletişim bağlantısı yoksa, hedef kuyruk yöneticisine giderken bir ya da daha çok *ara kuyruk yöneticisinden* geçebilirsiniz. Bu, *çoklu sekme*olarak bilinir.

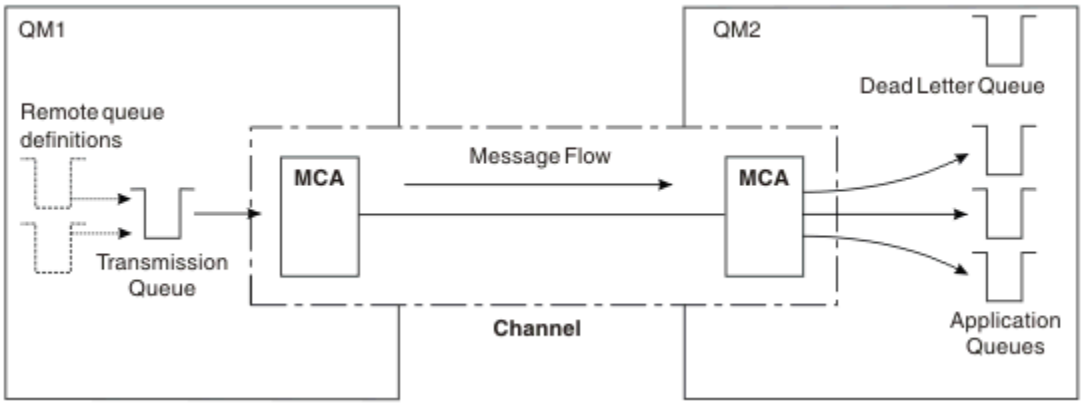
Tüm kuyruk yöneticileri ve ara kuyruk yöneticilerindeki iletim kuyrukları arasında kanal tanımlamanız gerekir. Bu, Şekil 12 sayfa 48içinde gösterilir.



Şekil 12. Ara kuyruk yöneticilerinden geçme

Kanalları paylaşma

Uygulama tasarımcısı olarak, uygulamalarınızı kuyruk adıyla birlikte uzak kuyruk yöneticisi adını belirtmeye ya da her uzak kuyruk için bir *uzak kuyruk tanımlaması* yaratmaya zorlama seçeneğiniz vardır. Bu tanımlama, uzak kuyruk yöneticisi adını, kuyruk adını ve iletim kuyruğunun adını içerir. Her iki yöntemle de, aynı uzak konumdaki kuyrukları adreslemekte olan tüm uygulamalardan gelen tüm iletilerin iletileri aynı iletim kuyruğundan gönderilir. Bu, Şekil 13 sayfa 49içinde gösterilir.



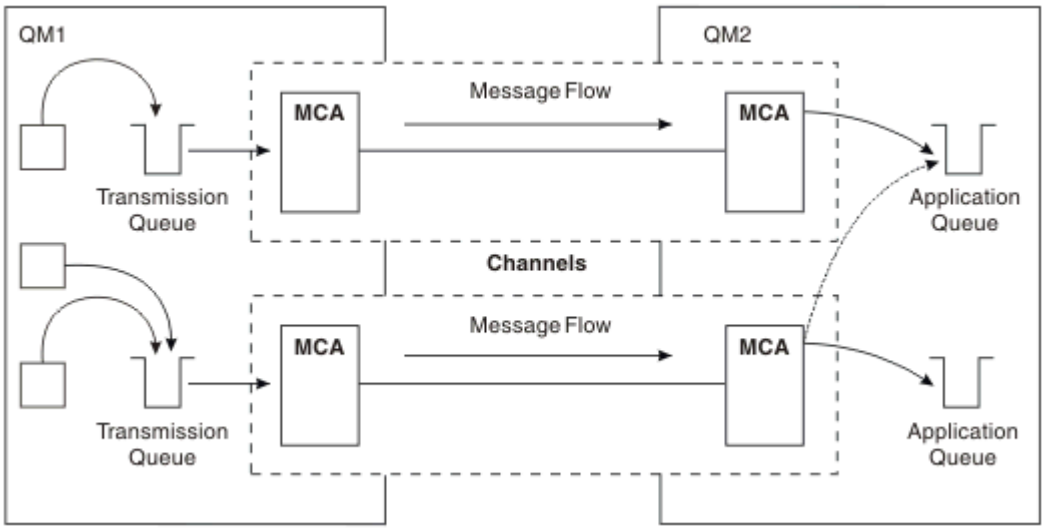
Şekil 13. İletim kuyruğunun paylaşılması

Şekil 13 sayfa 49 , birden çok uygulamadan birden çok uzak kuyruğa gelen iletilerin aynı kanalı kullanabildiğini gösterir.

Farklı kanallar kullanma

İki kuyruk yöneticisi arasında gönderilecek farklı tipte iletileriniz varsa, ikisi arasında birden çok kanal tanımlayabilirsiniz. Bazen alternatif kanallara ihtiyaç duyarsınız, belki de güvenlik amacıyla, ya da teslimat hızını mesaj trafiğinin büyük bir kısmı ile takas etmek için.

İkinci bir kanal kurmak için, başka bir kanal ve başka bir iletim kuyruğu tanımlamanız ve yeri ve iletim kuyruğu adını belirleyen bir uzak kuyruk tanımlaması yaratmanız gerekir. Uygulamalarınız daha sonra her iki kanalı da kullanabilir, ancak iletiler aynı hedef kuyruklara teslim edilir. Bu, Şekil 14 sayfa 49içinde gösterilir.



Şekil 14. Birden çok kanal kullanma

Bir iletim kuyruğu belirlemek için uzak kuyruk tanımlarını kullandığınızda, uygulamalarınız yeri (hedef kuyruk yöneticisi) **belirtmemelidir** . Varsa, kuyruk yöneticisi uzak kuyruk tanımlamalarını kullanmaz. Uzak kuyruk tanımlamaları size konum bağımsızlığı verir. Uygulamalar, kuyruğun yerini bilmeden iletileri *mantıksal* kuyruğa koyabilir ve uygulamalarınızı değiştirmek zorunda kalmadan *fiziksel* kuyruğu değiştirebilirsiniz.

Kümelemenin kullanılması

Bir kümedeki her kuyruk yöneticisi bir küme-alıcı kanalı tanımlar. Başka bir kuyruk yöneticisi bu kuyruk yöneticisine ileti göndermek istediğinde, ilgili küme gönderen kanalını otomatik olarak tanımlar.

Örneğin, bir kümede birden çok kuyruk eşgörünümü varsa, küme gönderen kanal, kuyruğu barındıran kuyruk yöneticilerinden herhangi birine tanımlanabilir. IBM MQ , bir iletiyi yöneltmek üzere kullanılabilir bir kuyruk yöneticisini seçmek için çevrimsel sıralı bir bin yordamını kullanan bir iş yükü yönetimi algoritmasını kullanır. Daha fazla bilgi için bkz. Kümeler.

Adresleme bilgileri

Bir uygulama uzak kuyruk yöneticisine gönderilecek iletileri yerleştirdiğinde, yerel kuyruk yöneticisi iletimi kuyruğuna yerleştirmeden önce bunlara bir iletim üstbilgisi ekler. Bu üstbilgi, hedef kuyruğun ve kuyruk yöneticisinin adını, yani *adresleme bilgilerini*içerir.

Tek bir kuyruk yöneticisi ortamında, bir uygulama iletileri koymak için bir kuyruk açtığında hedef kuyruğun adresi oluşturulur. Hedef kuyruk aynı kuyruk yöneticisinde olduğu için, herhangi bir adresleme bilgisine gerek yoktur.

Dağıtılmış bir kuyruğa alma ortamında, kuyruk yöneticisinin yalnızca hedef kuyruk adını değil, aynı zamanda bu kuyruğun yerini (kuyruk yöneticisi adı) ve o uzak yere (iletim kuyruğu) giden rotayı da bilmesi gerekir. Bu adresleme bilgileri iletim üstbilgisinde bulunur. Alıcı kanal, iletim üstbilgisini kaldırır ve hedef kuyruğu bulmak için içindeki bilgileri kullanır.

Uzak kuyruk tanımlaması kullanıyorsanız, uygulamalarınızın hedef kuyruk yöneticisinin adını belirtmesi gerekmesini önleyebilirsiniz. Bu tanımlama, uzak kuyruğun adını, iletilerin gönderileceği uzak kuyruk yöneticisinin adını ve iletileri iletmek için kullanılan iletim kuyruğunun adını belirler.

Takma adlar nedir?

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

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

Diğer ad tanımlamaları boş bir RNAME ile karakterize edilir.

Kuyruk adı çözümlemesi

Her kuyruk açıldığında kuyruk adı çözünürlüğü her kuyruk yöneticisinde ortaya çıkar. Amacı, hedef kuyruğu, hedef kuyruk yöneticisini (yerel olabilir) ve o kuyruk yöneticisine giden rotayı (boş olabilir) tanımlamaktır. Çözülen adın üç bölümü vardır: kuyruk yöneticisi adı, kuyruk adı ve kuyruk yöneticisi uzaksa, iletim kuyruğu.

Bir uzak kuyruk tanımlaması varsa, diğer ad tanımlamasına gönderme yapılmaz. Uygulama tarafından sağlanan kuyruk adı, hedef kuyruğun, uzak kuyruk yöneticisinin ve uzak kuyruk tanımında belirlenen iletim kuyruğunun adına çözülür. Kuyruk adı çözümlemesi hakkında daha ayrıntılı bilgi için bkz. Kuyruk adı çözümlemesi.

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

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

Bir uzak kuyruk tanımlaması ya da kuyruk yöneticisi diğer adı tanımı tarafından değiştirilmedikçe, kullanılan iletim kuyruğu genellikle çözülen kuyruk yöneticisiyle aynı ada sahiptir. Böyle bir iletim kuyruğu tanımlamadıysanız, ancak varsayılan bir 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 ad tanımlamaları, bir iletiyi koymak için bir kuyruğu açan bir uygulama, kuyruk adını **ve** kuyruk yöneticisi adını belirttiğinde geçerlidir.

Kuyruk yöneticisi diğer ad tanımlamalarının üç kullanımı vardır:

- İleti gönderirken, kuyruk yöneticisi adını yeniden eşleme
- İleti gönderirken, iletim kuyruğunu değiştirme ya da belirleme
- İletiler alınırken, yerel kuyruk yöneticisinin bu iletiler için hedeflenen hedef olup olmadığını belirleme

Giden iletiler-kuyruk yöneticisi adı yeniden eşleniyor

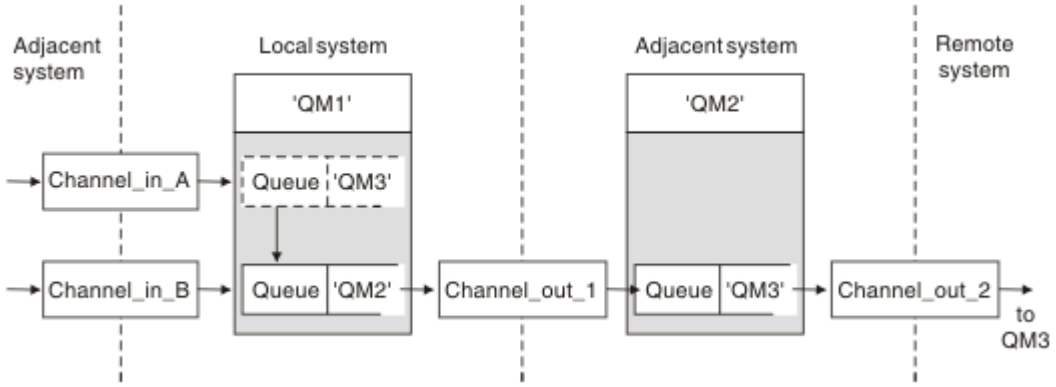
Kuyruk yöneticisi diğer ad tanımlamaları, bir MQOPEN çağrısında belirtilen kuyruk yöneticisi adını yeniden eşlemek için kullanılabilir. Örneğin, bir MQOPEN çağrısı `THISQ` kuyruk adını ve `YOURQM`kuyruk yöneticisi adını belirtir. Yerel kuyruk yöneticisinde, aşağıdaki örnek gibi bir kuyruk yöneticisi diğer adı tanımlaması vardır:

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

Bu, bir uygulama iletileri `YOURQM`kuyruk yöneticisine koyduğunda kullanılacak gerçek kuyruk yöneticisinin `REALQM`olduğunu gösterir. Yerel kuyruk yöneticisi `REALQM`ise, iletileri yerel bir kuyruk olan `THISQ`kuyruğuna koyar. Yerel kuyruk yöneticisi `REALQM`olarak adlandırılmazsa, iletiyi `REALQM`adlı bir iletim kuyruğuna yönlendirir. Kuyruk yöneticisi, iletim üstbilgisini `YOURQM`yerine `REALQM` olarak değiştirir.

Giden iletiler-iletim kuyruğunu değiştirme ya da belirleme

Şekil 15 sayfa 51 içinde iletilerin `QM1` kuyruk yöneticisine geldiği ve iletim üstbilgilerinin kuyruk adlarını kuyruk yöneticisinde gösterdiği bir senaryo gösterilmektedir `QM3`. Bu senaryoda `QM3` , `QM2`üzerinden çoklu sıçrama ile erişilebilir.



Şekil 15. Kuyruk yöneticisi diğer adı

`QM3` ile ilgili tüm iletiler, `QM1` kuyruk yöneticisi diğer adıyla yakalanır. Kuyruk yöneticisi diğer adı `QM3` ve `QM3` iletim kuyruğu yoluyla `QM2`tanımını içerir. Tanımlama aşağıdaki örneğe benzer:

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

Kuyruk yöneticisi iletileri `QM2` iletim kuyruğuna koyar, ancak `QM3`hedef kuyruk yöneticisinin adı değişmediği için iletim kuyruğu üstbilgisini değiştirmez.

QM1 adresine gelen ve QM2 adresinde bir kuyruk adı içeren bir iletim üstbilgisini gösteren tüm iletiler de QM2 iletim kuyruğuna yerleştirilir. Bu şekilde, farklı hedeflere sahip iletiler, hedeflerine doğru iletim için uygun bir bitişik sisteme ortak bir iletim kuyruğunda toplanır.

Gelen iletiler-hedefin belirlenmesi

Alıcı bir MCA, iletim üstbilgisinde başvurulan kuyruğu açar. Gönderme yapılan kuyruk yöneticisiyle aynı adı taşıyan bir kuyruk yöneticisi diğer adı tanımlaması varsa, iletim üstbilgisinde alınan kuyruk yöneticisi adı, o tanımdaki RQMNAME ile değiştirilir.

Bu sürecin iki kullanımı vardır:

- İletileri başka bir kuyruk yöneticisine yönlendirme
- Kuyruk yöneticisi adı, yerel kuyruk yöneticisiyle aynı olacak şekilde değiştiriliyor

Yanıtın gönderileceği kuyruk diğer adı tanımlamaları

Yanıtın gönderileceği kuyruk diğer adı tanımlaması, ileti tanımlayıcıdaki yanıt bilgilerine ilişkin diğer adları belirler. Bunun yararı, uygulamalarınızı değiştirmek zorunda kalmadan bir kuyruk ya da kuyruk yöneticisinin adını değiştirebilmenizdir.

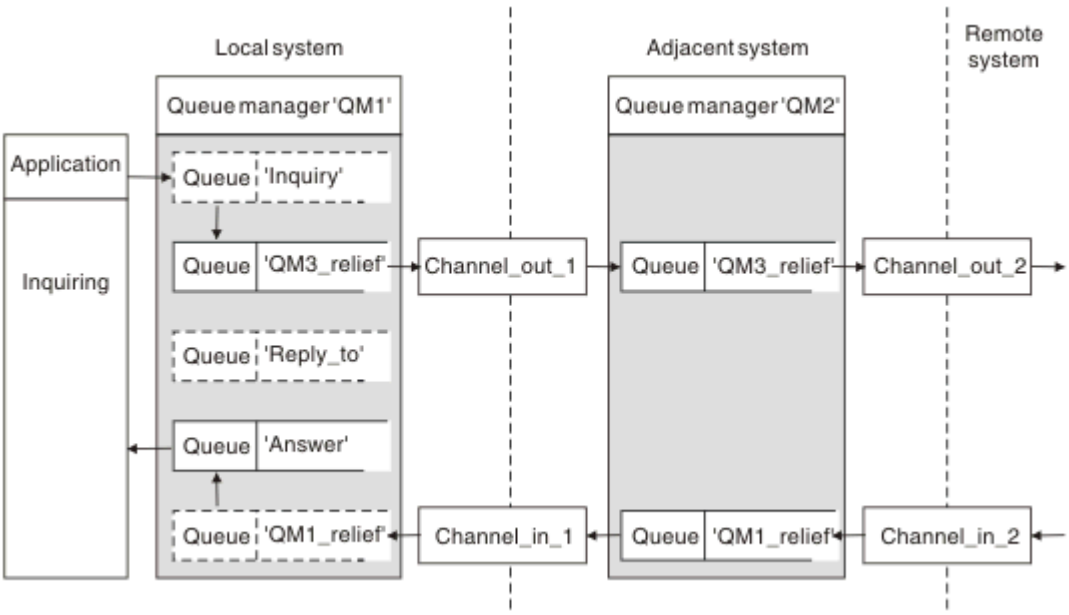
Kuyruk adı çözümlemesi

Bir uygulama bir iletiyi yanıtladığında, yanıtlanacak kuyruğun adını bulmak için aldığı iletinin *ileti tanımlayıcısındaki* verileri kullanır. Gönderen uygulama, yanıtların nereye gönderildiğini gösterir ve bu bilgileri iletilerine ekler. Bu kavram, uygulama tasarımınızın bir parçası olarak koordine edilmelidir.

Kuyruk adı çözülmesi, ileti bir kuyruğa konmadan önce, uygulamanızın gönderme sonunda gerçekleşir. Bu nedenle, kuyruk adı çözülmesi, iletinin gönderildiği uzak uygulamayla etkileşimden önce gerçekleşir. Bu, bir kuyruk açılmadığında ad çözümlenmesinin gerçekleştiği tek durumdur.

Kuyruk yöneticisi diğer adı kullanılarak kuyruk adı çözülmesi

Olağan durumda bir uygulama, bir yanıt kuyruğunu belirtir ve yanıtın gönderileceği kuyruk yöneticisi adını boş bırakır. Kuyruk yöneticisi, koyma sırasında kendi adını tamamlar. Bu yöntem, örneğin, QM1iletim kuyruğunu kullanan varsayılan dönüş kanalı yerine `QM1_relief` iletim kuyruğunu kullanan bir kanal gibi yanıtlar için alternatif bir kanal kullanılmasını istemediğiniz durumlar dışında iyi çalışır. Bu durumda, iletim kuyruğu üstbilgilerinde belirtilen kuyruk yöneticisi adları "gerçek" kuyruk yöneticisi adlarıyla eşleşmiyor, ancak kuyruk yöneticisi diğer ad tanımları kullanılarak yeniden belirtildi. Yanıtları alternatif rotalar boyunca döndürmek için, yanıtlanacak kuyruk diğer ad tanımlarını kullanarak yanıtlanacak kuyruk verilerini de eşlemek gerekir.



Şekil 16. Yanıt konumunu değiştirmek için kullanılan yanıt kuyruğu diğer adı

Şekil 16 sayfa 53örneğinde:

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

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

   Yanıtlanacak kuyruk diğer adının kullanılabilmesi için ReplyToQMgr boş olmalıdır.
2. `Answer`adını ve `QM1_relief`kuyruk yöneticisi adını içeren `Reply_to`adlı bir yanıt kuyruğu diğer adı tanımı yaratırsınız.

   ```
   DEFINE QREMOTE ('Reply_to') RNAME ('Answer')
   RQMNAME ('QM1_relief')
   ```
3. İletiler, `ReplyToQ='Answer'` ve `ReplyToQMgr='QM1_relief''`yi gösteren bir ileti tanımlayıcısıyla gönderilir.
4. Uygulama belirtimi, `Reply_to`yerine `Answer` kuyruğunda bulunacak yanıtlara ilişkin bilgileri içermelidir.

Yanıtlara hazırlanmak için koşut dönüş kanalını yaratmanız gerekir, tanımlama:

- QM2' de, `QM1_relief` adlı iletim kuyruğu

  ```
  DEFINE QLOCAL ('QM1_relief') USAGE(XMITQ)
  ```
- QM1' de, kuyruk yöneticisi diğer adı `QM1_relief`

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

  Bu kuyruk yöneticisi diğer adı, koşut dönüş kanalları zincirini sonlandırır ve QM1ile ilgili iletileri yakalar.

Bunu gelecekte bir zamanda yapmak isteyebileceğinizi düşünüyorsanız, uygulamaların başlangıçtaki diğer adı kullandığından emin olun. Şu an için bu, yanıt kuyruğunun olağan kuyruk diğer adıdır, ancak daha sonra bir kuyruk yöneticisi diğer adıyla değiştirilebilir.

Yanıtın gönderileceği kuyruk adı

Yanıt kuyruklarının adlandırılması için dikkatli olmak gerekir. Bir uygulamanın iletiye yanıt kuyruğu adı koymasının nedeni, yanıtlarının gönderileceği kuyruğu belirleyebilmektir. Bu adla bir yanıt kuyruğu diğer adı tanımı yarattığınızda, aynı ada sahip gerçek bir yanıt kuyruğuna (yerel kuyruk tanımı) sahip olamazsınız. Bu nedenle, yanıtlanacak kuyruk diğer adı tanımlamasının kuyruk yöneticisi adının yanı sıra yeni bir kuyruk adı da içermesi ve uygulama belirtiminin yanıtlarının diğer kuyrukta bulunduğu bilgileri içermesi gerekir.

Uygulamaların artık, özgün iletiyi yerleştirdiklerinde, yanıt kuyruğu olarak adlandırdıkları kuyruktan farklı bir kuyruktan iletileri alması gerekir.

Küme bileşenleri

Kümeler, kuyruk yöneticileri, küme havuzları, küme kanalları ve küme kuyruklarından oluşur.

Küme bileşenlerinin her birine ilişkin bilgi için aşağıdaki alt başlıklara bakın:

İlgili kavramlar
Kümeleme ve dağıtılmış kuyruğa alma karşılaştırması
İlgili görevler
Kuyruk yöneticisi kümesinin yapılandırılması
Yeni bir küme ayarlama

Küme havuzu

Havuz, bir kümenin üyesi olan kuyruk yöneticilerine ilişkin bilgilerin bir toplamidir.

Havuz bilgileri, kuyruk yöneticisi adlarını, yerlerini, kanallarını, barındıkları kuyrukları ve diğer bilgileri içerir. Bilgiler, `SYSTEM.CLUSTER.REPOSITORY.QUEUE`adlı bir kuyrukta ileti biçiminde saklanır. Bu kuyruk, varsayılan nesnelerden biridir. Multi Açık Çoklu platformlar, bir IBM MQ kuyruk yöneticisi yarattığınızda tanımlanır. z/OS Açık IBM MQ for z/OS, kuyruk yöneticisi uyarlamasının bir parçası olarak tanımlanır.

Tam havuz ve kısmi havuz

Genellikle, bir kümedeki iki kuyruk yöneticisi tam bir havuzu tutar. Geri kalan kuyruk yöneticilerinin tümü kısmi bir havuz bulundurmaktadır.

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

Kısmi bir havuz, yalnızca kuyruk yöneticisinin ileti değiş tokuşu yapması gereken kuyruk yöneticileriyle ilgili bilgileri içerir. Kuyruk yöneticileri, gereksinim duydukları bilgilerde güncelleme yapılmasını istemektedir; böylece, bu bilgiler değişirse, tam havuz kuyruk yöneticisi yeni bilgileri onlara gönderir. Çoğu zaman, kısmi bir havuz, bir kuyruk yöneticisinin küme içinde gerçekleştirmesi gereken tüm bilgileri içerir. Bir kuyruk yöneticisi bazı ek bilgilere gereksinim duyduğunda, tam havuz sorguları yapar ve kısmi havuzunu günceller. Kuyruk yöneticileri, havuzlara ilişkin güncellemeleri istemek ve almak için `SYSTEM.CLUSTER.COMMAND.QUEUE` kuyruğunu kullanır.

Bir kümenin üyesi olan kuyruk yöneticilerini geçirirken , kısmi havuzlardan önce tam havuzları geçirin. Bunun nedeni, daha eski bir havuzun daha yeni bir yayın düzeyinde tanıtılan yeni öznitelikleri saklayamıyor olmasıdır. Onlara tahammül eder ama onları saklamaz.

Küme kuyruk yöneticisi

Küme kuyruk yöneticisi, bir kümenin üyesi olan bir kuyruk yöneticisidir.

Bir kuyruk yöneticisi birden çok kümenin üyesi olabilir. Her küme kuyruk yöneticisinin, üyesi olduğu tüm kümelerde benzersiz bir adı olmalıdır.

Bir küme kuyruğu yöneticisi, kümedeki diğer kuyruk yöneticilerine duyurduğu kuyrukları barındıracak. Ama bunu yapmak zorunda değil. Bunun yerine, kümedeki başka bir yerde bulunan kuyruklara iletileri besleyebilir ve yalnızca belirtik olarak yönlendirilen yanıtları alabilir.

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

Küme kuyruğu yöneticileri özerk. Tanımladıkları kuyruklar ve kanallar üzerinde tam denetime sahipler. Bunların tanımlamaları diğer kuyruk yöneticileri (aynı kuyruk paylaşım grubundaki kuyruk yöneticileri dışında) tarafından değiştirilemez. Havuz kuyruğu yöneticileri, kümedeki diğer kuyruk yöneticilerindeki tanımlamaları denetleyemez. Bunlar, gerektiğinde kullanılmak üzere tüm tanımlardan oluşan tam bir kümeye sahip olur. Küme, kuyruk yöneticilerinin bir birleştiricidir.

Bir küme kuyruğu yöneticisinde tanımlama yarattıktan ya da değiştirdikten sonra, bilgiler tam havuz kuyruğu yöneticisine gönderilir. Kümedeki diğer havuzlar daha sonra güncellenir.

Tam Havuz kuyruk yöneticisi

Tam havuz kuyruğu yöneticisi, küme kaynaklarının tam gösterimini tutan bir küme kuyruğu yöneticisidir. Kullanılabilirliği sağlamak için, her bir kümede iki ya da daha fazla tam havuz kuyruğu yöneticisi ayarlayın. Tam havuz kuyruğu yöneticileri, kümedeki diğer kuyruk yöneticileri tarafından gönderilen bilgileri alır ve havuzlarını günceller. Her ikisinin de kümeyle ilgili yeni bilgilerle güncel tutulduğundan emin olmak için birbirlerine ileti gönderirler.

Kuyruk yöneticileri ve havuzlar

Her kümede, bir kümedeki kuyruk yöneticileri, kuyrukları ve kanallar hakkında tam havuz bulunduran en az bir (tercihen iki) kuyruk yöneticisi bulunur. 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 altkümesiyle ilgili bilgileri içeren kısmi bir havuzu tutar. Kuyruk yöneticileri, başka bir kuyruğa ya da kuyruk yöneticisine ilk erişmeleri gerektiğinde sorguları yaparak kısmi havuzlarını oluşturur. Bu kuyruk ya da kuyruk yöneticisine ilişkin yeni bilgilerin kendilerine bildirilmesini isterler.

Her kuyruk yöneticisi havuz bilgilerini `SYSTEM.CLUSTER.REPOSITORY.QUEUE`adlı bir kuyruktaki iletilerde saklar. Kuyruk yöneticileri, `SYSTEM.CLUSTER.COMMAND.QUEUE`adlı bir kuyruktaki iletilerde havuz bilgilerini değiştirir.

Bir kümeyi birleşen her kuyruk yöneticisi, bir kümeyi gönderen `CLUSSDR`, havuzlardan birine kanal tanımlar. Kümedeki diğer kuyruk yöneticilerinin tam havuzları tuttuğunu hemen öğrenir. Bundan sonra, kuyruk yöneticisi herhangi bir havuzdan bilgi isteyebilir. Kuyruk yöneticisi seçilen havuza bilgi gönderdiğinde, bilgileri başka bir havuza da gönderir (varsa).

Tam havuz, kuyruk yöneticisi kendisine bağlı kuyruk yöneticilerinden birinden yeni bilgiler aldığında güncellenir. Yeni bilgiler, bir havuz kuyruk yöneticisi hizmet dışı kaldıysa erteleme riskini azaltmak için başka bir havuza da gönderilir. Tüm bilgiler iki kez gönderildiği için, havuzların yinelemeleri atması gerekir. Her bilgi öğesi, havuzların yinelemeleri tanımlamak için kullandığı bir sıra numarası taşır. Tüm havuzlar, ileti değiş tokuşuyla birbiriyle adım adım 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 hangilerinin sisteminiz için daha uygun olduğuna karar vermenize yardımcı olmak üzere tasarlanmıştır.

Kanal Başlatıcı Maliyetleri

Küme kuyruklarında, iletiler kanallar tarafından gönderilir, bu nedenle uygulama maliyetlerine ek olarak kanal başlatıcı maliyetlerine izin verin. Kanallar mesajları alıp koyduğundan ağda maliyetler var. Bu maliyetler paylaşılan kuyruklarda bulunmaz; bu nedenle, bir kuyruk paylaşım grubundaki kuyruk yöneticileri arasında ileti taşırken küme kuyruklarına göre daha az işleme gücü kullanılır.

İletilerin kullanılabilirliği

Bir kuyruğa koyarken, küme kuyrukları iletiyi, kuyruk yöneticinize bağlı etkin kanalları olan kuyruk yöneticilerinden birine gönderir. Uzak kuyruk yöneticisinde, iletileri işlemek için kullanılan uygulamalar çalışmıyorsa, iletiler işlenmez ve uygulamalar başlayıncaya kadar beklenir. Benzer şekilde, bir kuyruk yöneticisi kapatılırsa, kuyruk yöneticisi yeniden başlatılıncaya kadar kuyruk yöneticisindeki iletiler kullanılamaz. Bu eşgörünümler, paylaşılan kuyrukları kullanmaktan daha düşük ileti kullanılabilirliğini gösterir.

Paylaşılan kuyrukları kullanırken, kuyruk paylaşım grubundaki herhangi bir uygulama gönderilen iletileri alabilir. Kuyruk paylaşım grubundaki bir kuyruk yöneticisini kapatırsanız, iletiler diğer kuyruk yöneticilerinin kullanımına sunulur ve küme kuyruklarının kullanımından daha yüksek ileti kullanılabilirliği sağlar.

Kapasite

Bir bağlaşım olanağı bir diskten daha pahalıdır; bu nedenle, yerel bir kuyrukta 1.000.000 ileti saklama maliyeti, aynı sayıda iletiyi depolamak için yeterli kapasiteye sahip bir bağlaşım olanağına sahip olandan daha düşüktür.

Diğer kuyruk yöneticilerine gönderme

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

İş yükü dengeleme

Hangi kanalların ve kuyruk yöneticilerinin gönderilen iletilerin oranını aldığı ağırlığı vermek için kümelemeyi kullanabilirsiniz. Örneğin, iletilerin %60 'ını bir kuyruk yöneticisine, %40 'ını başka bir kuyruk yöneticisine gönderebilirsiniz. Bu eşgörünüm, uzak kuyruk yöneticisinin işi işleme yeteneğine bağlı değildir. İlk kuyruk yöneticisine sahip sistem aşırı yüklenmiş olabilir ve ikinci kuyruk yöneticisine sahip sistem boşta olabilir, ancak iletilerin çoğu yine de ilk kuyruk yöneticisine gider.

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

Küme kanalları

Her tam havuzda, kümedeki diğer tüm havuzlara bağlanmak için bir küme alıcı kanalı ve bir küme gönderen kanal kümesi tanımlarsınız. Kısmi bir havuz eklediğinizde, el ile bir küme alıcı kanalı ve tam havuzlardan birine bağlanan tek bir küme gönderen kanalı tanımlarsınız. Diğer küme-gönderen kanalları gerektiğinde küme tarafından otomatik olarak tanımlanır. Otomatik olarak tanımlanan küme gönderen kanalları, özniteliklerini alan kuyruk yöneticisindeki karşılık gelen küme alıcı kanal tanımından alır.

Küme-alıcı kanalı: CLUSRCVR

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

Her küme kuyruğu yöneticisi için en az bir CLUSRCVR kanalı tanımlamanız gerekir. Kuyruk yöneticisi, CLUSRCVR kanalını tanımlayarak, iletileri alabileceği diğer küme kuyruğu yöneticilerini gösterir.

CLUSRCVR kanal tanımlaması, diğer kuyruk yöneticilerinin karşılık gelen küme gönderen kanal tanımlarını otomatik olarak tanımlamalarını da sağlar. Bu makalenin “Otomatik tanımlı küme gönderen kanalları” sayfa 57 bölümüne bakın.

Küme-gönderen kanalı: CLUSSDR

Her tam havuz kuyruğu yöneticisinden kümedeki diğer tüm havuz kuyruğu yöneticisine bir CLUSSDR kanalını el ile tanımlayabilirsiniz. Tüm havuzlar tarafından değiş tokuş edilen tüm güncellemeler yalnızca bu kanallarda akmaktadır. Bu kanalları el ile tanımlayarak, tam havuzların ağını belirtik olarak denetleyebilirsiniz.

Bir kümeye kısmi bir havuz kuyruğu yöneticisi eklediğinizde, tam havuzlardan birine bağlanmak için el ile tek bir CLUSSDR kanalı tanımlarsınız. İlk kişi yapıldıktan sonra CLUSSDR kanalları da dahil olmak üzere kuyruk yöneticiniz için diğer küme kuyruğu yöneticisi nesneleri otomatik olarak gerektiği şekilde tanımlandığından, seçtiğiniz tam havuz çok az fark yaratır. Bu, kuyruk yöneticinizin küme bilgilerini herhangi bir tam havuza göndermesini ve kümedeki herhangi bir kuyruk yöneticisine ileti göndermesini sağlar.

Bu makalenin bölümünde açıklandığı gibi, otomatik olarak tanımlanan gönderen kanalları, küme alıcı kanalının konfigürasyonuna dayanmaktadır. Bu nedenle, küme kanallarında ayarladığınız kanal özellikleri, eşleşen CLUSSDR ve küme alıcı kanallarında ya da yalnızca küme alıcı kanallarında ayarlanmalıdır.

CLUSSDR kanallarını yalnızca daha önce açıklanan nedenlerle el ile tanımlamanız gerekir. Diğer bir deyişle, kısmi bir havuzu başlangıçta tam bir havuza bağlamak ya da iki tam havuzu birbirine bağlamak. Kısmi bir havuza ya da kümede olmayan bir kuyruk yöneticisine bağlanan bir CLUSSDR kanalının el ile yapılandırılması, AMQ9427 ve AMQ9428 gibi hata iletilerinin yayınlanmasına neden olur. Bu bazen geçici bir durum olarak önlenemeyen bir durum olabilir; örneğin, tam bir havuzun konumunu değiştirirken, el ile tanımlama en kısa zamanda silinmelidir.

Otomatik tanımlı küme gönderen kanalları

Genellikle bir kümeye kısmi bir havuz kuyruğu yöneticisi eklediğinizde, kuyruk yöneticisinde yalnızca iki küme kanalını el ile tanımlarsınız:

- Küme gönderen (CLUSSDR) Kümeye ilişkin tam havuz kuyruğu yöneticisine yönlendirin.
- Küme alıcısı (CLUSRCVR) Kanal.

Tanımladığınız CLUSSDR kanalı, kuyruk yöneticisinin kümeyle ilk kez iletişim kurmasını sağlar. İlk iletişim kurulduktan sonra, CLUSSDR kanalları gerektiğinde küme tarafından otomatik olarak tanımlanır.

Otomatik tanımlı bir CLUSSDR kanalı, özniteliklerini alan kuyruk yöneticisindeki ilgili CLUSRCVR kanal tanımından alır. El ile tanımlanmış bir CLUSSDR kanalı olsa bile, otomatik olarak tanımlanan CLUSSDR kanalındaki öznitelikler kullanılır. Örneğin, **CONNAME** parametresinde bir kapı numarası belirtmeden

CLUSRCVR kanalı tanımladığınızı ve kapı numarası belirten bir CLUSSDR kanalını el ile tanımladığınızı varsayın. Otomatik olarak tanımlanan CLUSSDR kanalı el ile tanımlanan kanalın yerini aldığında, kapı numarası (CLUSRCVR kanalından alınan) boş olur. Varsayılan kapı numarası kullanılır ve kanal başarısız olur.

El ile tanımlanan CLUSSDR kanalı ile ilgili CLUSRCVR kanal tanımlaması arasında yapılandırma farklılıkları varsa, bazı farklılıklar hemen yürürlüğe girer (örneğin, iş yükü dengeleme parametreleri) ve bazıları yalnızca kanal yeniden başlatmada (örneğin, TLS yapılandırması) etkili olur.

Karışıklığı önlemek için, mümkün olduğunca aşağıdaki yönergeleri izleyin:

- CLUSSDR kanallarını yalnızca tam havuzları gösterecek şekilde el ile tanımlayın.
- CLUSSDR kanallarını el ile tanımladığınız yerlerde, bunları alıcı kuyruk yöneticisindeki ilgili CLUSRCVR kanal tanımlamasıyla aynı şekilde eşleşecek şekilde yapılandırın.

Ayrıca bkz. Otomatik tanımlı kanallarla çalışma.

İlgili kavramlar
Otomatik tanımlı kanallarla çalışma
Küme iletim kuyrukları ve küme gönderen kanallarıyla çalışma
İlgili görevler
Yeni bir küme ayarlama
Kümeye 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, bu konunun tanımlandığı (anasistem) kuyruk yöneticisinde yapılmalıdır; bu noktada değişiklik, tam havuzlar aracılığıyla kümenin tüm üyelerine yayılır.

Küme konularının *doğrudan yöneltme* ya da *konu anasistem yöneltmesi*kullanacak şekilde yapılandırılması ve kümelenmiş konu kalıtımı ve genel arama karakteri abonelikleri hakkında bilgi 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
DISPLAYTOPIC
DISPLAYTPSTATUS
DISPLAYSUB

Varsayılan küme nesneleri

Multi Multiplatforms üzerinde, varsayılan küme nesneleri, bir kuyruk yöneticisi tanımladığınızda otomatik olarak yaratılan varsayılan nesneler kümesine eklenir. z/OS z/OSüzerinde, varsayılan küme nesnesi tanımlamaları uyarlama 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 yerel bir kuyruğu vardır. İleti, kuyruk yöneticisiyle ilgili yeni ya da değiştirilmiş bilgileri ya da diğer kuyruk yöneticileriyle ilgili bilgi isteklerini içerir. SYSTEM.CLUSTER.COMMAND.QUEUE olağan durumda boştur.

SYSTEM.CLUSTER.HISTORY.QUEUE
Bir kümedeki her kuyruk yöneticisinin SYSTEM.CLUSTER.HISTORY.QUEUEadlı yerel bir 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 ayarlanır. Geçmiş derlemini engellemek için ayarı PUT (DISABLED) olarak değiştirin.

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

SYSTEM.CLUSTER.TRANSMIT.QUEUE
Her kuyruk yöneticisinin SYSTEM.CLUSTER.TRANSMIT.QUEUEadlı yerel bir kuyruk için bir tanımlaması vardır. SYSTEM.CLUSTER.TRANSMIT.QUEUE , kümeler içindeki tüm kuyruklara ve kuyruk yöneticilerine gönderilen tüm iletilere ilişkin varsayılan iletim kuyruğudur. DEFCLXQ kuyruk yöneticisi özniteliğini değiştirerek, her küme gönderen kanalı için varsayılan iletim kuyruğunu SYSTEM.CLUSTER.TRANSMIT. *ChannelName*olarak değiştirebilirsiniz. SYSTEM.CLUSTER.TRANSMIT.QUEUEöğesini silemezsiniz. Kullanılan varsayılan iletim kuyruğunun SYSTEM.CLUSTER.TRANSMIT.QUEUE ya da SYSTEM.CLUSTER.TRANSMIT. *ChannelName*olup olmadığını belirlemek için de kullanılır.

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

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

İlgili kavramlar
Varsayılan küme nesneleriyle çalışma

İleti alışverişi yayınla/abone ol

Yayınlama/abone olma ileti sistemi, bilgi sağlayıcısını bu bilgilerin kullanıcılarından ayıklamanızı sağlar. Gönderilen ve alınan bilgilerin gönderilmesi ve alınması için gönderen uygulamanın birbirleri hakkında hiçbir şey bilmesine gerek yoktur.

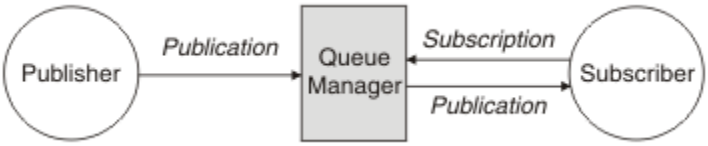
Bir noktadan noktaya iletişim IBM MQ uygulaması başka bir uygulamaya ileti göndermeden önce, o uygulama hakkında bir şeyler bilmesi gerekir. Örneğin, bilgilerin gönderileceği kuyruğun adını bilmesi ve bir kuyruk yöneticisi adı belirtmesi gerekir.

IBM MQ yayınlama/abone olma, uygulamanızın hedef uygulamayla ilgili her şeyi bilmesini sağlar. Göndericinin yapması gereken tek şey:

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

Benzer şekilde, hedef uygulamanın aldığı bilgilerin kaynağı hakkında hiçbir şey bilmesine gerek yoktur.

Aşağıdaki şekil, en basit yayınlama/abone olma sistemini göstermektedir. Bir yayınlayıcı, bir kuyruk yöneticisi ve bir abone vardır. Kuyruk yöneticisindeki abone tarafından abonelik yapılır, yayınlayıcıdan kuyruk yöneticisine bir yayın gönderilir ve daha sonra yayın kuyruk yöneticisi tarafından aboneye iletilir.



Şekil 17. Basit yayınlama/abone olma yapılandırması

Tipik bir yayınlama/abone olma sistemi birden çok farklı konuda birden çok yayınlayıcı ve birden çok abone içerir ve genellikle birden çok kuyruk yöneticisi vardır. Bir uygulama hem yayınlayıcı hem de abone olabilir.

Yayınlama/abone olma ileti sistemi ile noktadan noktaya iletişim arasındaki diğer bir önemli fark, noktadan noktaya iletişim kuyruğuna gönderilen bir iletinin yalnızca tek bir tüketen uygulama tarafından işlendiğidir. Birden çok abonenin ilgi alanı kaydettiği bir yayınlama/abone olma konusuna yayınlanan bir ileti, ilgili her abone tarafından işlenir.

Bileşenleri yayınla/abone ol

Yayınlama/abone olma, abonelerin yayıncılardan ileti şeklinde bilgi alabileceği bir mekanizmadır. Yayıncılar ve aboneler arasındaki etkileşimler, standart IBM MQ olanaklarını kullanarak kuyruk yöneticileri tarafından denetlenir.

Tipik bir yayınlama/abone olma sistemi birden çok farklı konuda birden çok yayınlayıcı ve birden çok abone içerir ve genellikle birden çok kuyruk yöneticisi vardır. Bir uygulama hem yayınlayıcı hem de abone olabilir.

Bilgi sağlayıcısına *yayınlayıcı*adı verilir. Yayıncılar, bu bilgilerle ilgilenen uygulamalar hakkında bilgi edinmeye gerek kalmadan, bir konu hakkında bilgi sağlar. Yayıncılar bu bilgileri yayınlamak istedikleri *yayınlar* adı verilen iletiler biçiminde oluşturur ve bu iletilerin konusunu tanımlar.

Bilgilerin tüketicisine *abone*denir. Aboneler, abonenin ilgilendiği konuyu açıklayan *abonelikler* oluşturur. Bu nedenle abonelik, hangi yayınların aboneye iletileceğini belirler. Aboneler birden çok abonelik yapabilir ve birçok farklı yayıncıdan bilgi alabilir.

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

Her bir iletiye, noktadan noktaya mesajlaşmada gerektiği gibi belirli bir hedef ekleme gereksinimi ortadan kaldırılarak, bilgi sağlayıcıların ve tüketicilerin yayınlama/abone olma mesajında ayrışmasına olanak sağlayan konuların varlığıdır.

Yayıncılar ve aboneler arasındaki etkileşimlerin tümü bir kuyruk yöneticisi tarafından denetlenir. Kuyruk yöneticisi, yayıncılardan ve abonelerden (bir dizi konuya) ileti alır. Kuyruk yöneticisinin işi, yayınlanan iletileri, iletilerin konusuna ilgi duyan abonelere yönlendirmektir.

Standart IBM MQ olanakları iletileri dağıtmak için kullanılır; böylece, uygulamalarınız var olan IBM MQ uygulamalarının kullanabileceği tüm özellikleri kullanabilir. Bu, yalnızca bir kez güvenli teslim almak için kalıcı iletileri kullanabileceğiniz ve iletilerinizin yalnızca yayıncı tarafından kesinleştirildiğinde aboneye teslim edilmelerini sağlamak için bir işlem biriminin parçası olacağı anlamına gelir.

Yayıncılar ve yayınlar

IBM MQ yayınlama/abone olma içinde bir yayınlayıcı, belirli bir konuya ilişkin bilgileri, yayın adı verilen standart bir IBM MQ iletisi biçiminde kuyruk yöneticisine sağlayan bir uygulamadır. Bir yayınlayıcı, birden çok konuya ilişkin bilgileri yayınlayabilir.

Yayıncılar, daha önce açılmış bir konuya ileti koymak için MQPUT fiilini kullanır; bu ileti bir yayındır. Daha sonra, yerel kuyruk yöneticisi yayını, yayının konusuna abonelikleri olan tüm abonelere yönlendirir. Yayınlanan bir ileti birden çok abone tarafından tüketilebilir.

Bir kuyruk yöneticisi, yayınları uygun abonelikleri olan tüm yerel abonelere dağıtmanın yanı sıra, yayını doğrudan ya da konuya aboneleri olan kuyruk yöneticilerinden oluşan bir ağ aracılığıyla da bu yayına bağlı diğer kuyruk yöneticilerine dağıtabilir.

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

Syncpoint altındaki yayınlar

Yayınlayıcılar, bir iş birimindeki abonelere teslim edilen tüm iletileri dahil etmek için eşitleme noktasında MQPUT ya da MQPUT1 çağrıları yayınlayabilir. MQPMO_RETAIN seçeneği ya da ALL ya da ALLDUR değerlerini içeren NPMSGDLV ya da PMSGDLV konu teslim seçenekleri belirtilirse, kuyruk yöneticisi, yayınlayıcı MQPUT ya da MQPUT1 çağrısı kapsamında, syncpoint içinde iç MQPUT ya da MQPUT1 çağrılarını kullanır.

Durum ve Olay Bilgileri

Yayınlar, hisse senetlerinin güncel fiyatı ya da hisse senetlerindeki ticaret gibi olay yayınları gibi devlet yayınları olarak kategorilere ayrılabilir.

Eyalet yayınları

Eyalet yayınları , stok fiyatı ya da bir futbol maçındaki geçerli puan gibi bir şeyin geçerli durumu hakkında bilgi içerir. Bir şey olduğunda (örneğin, hisse senedi fiyatı değiştiğinde veya futbol puanı değiştiğinde), yeni bilgiler yerini aldığından önceki durum bilgisi artık gerekli değildir.

Bir abone, başladığında durum bilgilerinin geçerli sürümünü almak ve durum değiştiğinde yeni bilgiler gönderilmesini ister.

Bir yayın durum bilgisi içeriyorsa, genellikle korunan bir yayın olarak yayınlanır. Yeni bir abone genellikle yürürlükteki durum bilgilerini hemen ister; abone, bilgilerin yeniden yayınlanmasına neden olan bir olayı beklemek istemez. Abone MQSO_PUBLICATIONS_ON_REQUEST ya da MQSO_NEW_PUBLICATIONS_ONLY seçeneklerini kullanmadığı sürece, aboneler abone olduğunda bir konunun alıkonan yayınını otomatik olarak alır.

Etkinlik yayınları

Etkinlik yayınları , belirli bir stoktaki ticaret ya da belirli bir hedefin puanlaması gibi gerçekleşen tek tek olaylarla ilgili bilgileri içerir. Her olay diğer olaylardan bağımsızdır.

Bir abone, olaylar hakkında gerçekleştikçe bilgi almak isteyecektir.

Yayınları alıkoyma

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

Tüm ilgili abonelere gönderildikten sonra yayınların silinmesi olay bilgileri için uygundur, ancak durum bilgileri için her zaman uygun değildir. Bir iletiyi alıkoyarak, yeni abonelerin ilk durum bilgilerini almadan önce bilgilerin yeniden yayınlanmasını beklemesi gerekmez. Örneğin, hisse senedi fiyatına aboneliği olan bir abone, hisse senedi fiyatının değişmesini beklemeden geçerli fiyatı hemen alır (ve bu nedenle yeniden yayınlanır).

Kuyruk yöneticisi her konu için yalnızca bir yayın tutabilir; bu nedenle, alıkonan yeni bir yayın kuyruk yöneticisine geldiğinde, bir konunun alıkonan yayını silinir. Ancak, var olan yayının silinmesi, yeni alıkonan yayının gelmesiyle eşzamanlı olarak gerçekleşmeyebilir. Bu nedenle, mümkünse, herhangi bir konuda alıkonan yayınları gönderen birden fazla yayıncı yoktur.

Aboneler, MQSO_NEW_PUBLICATIONS_ONLY abonelik seçeneğini kullanarak alıkonan yayınları almak istemediklerini belirtebilir. Var olan aboneler, alıkonan yayınların kopyalarının kendilerine gönderilmesini isteyebilir.

Durum bilgileri için bile yayınları saklamak istemeyebileceğiniz zamanlar vardır:

- Bir konuda herhangi bir yayın yapılmadan önce bir konuya ilişkin tüm abonelikler yapılırsa ve yeni aboneliklere izin vermezseniz, yayınları ilk yayınlandığında tam abone kümesine teslim edildikleri için saklamanıza gerek yoktur.
- Yayınlar her saniye gibi sık sık ortaya çıkarsa, yeni bir abone (ya da bir hatadan kurtaran bir abone), ilk aboneliğinden hemen sonra geçerli durumu alır, bu nedenle bu yayınları saklamaya gerek yoktur.
- Yayınlar büyükse, her konu için alıkonan yayını saklamak için önemli miktarda depolama alanına ihtiyacınız olabilir. Birden çok kuyruk yöneticisi ortamında, alıkonan yayınlar, ağdaki eşleşen bir aboneliği olan tüm kuyruk yöneticileri tarafından saklanır.

Alıkonan yayınların kullanılıp kullanılmayacağına karar verirken, abone olan uygulamaların bir hatadan nasıl kurtulduğunu göz önünde bulundurun. Yayınlayıcı alıkonan yayınları kullanmıyorsa, abone uygulamasının yürürlükteki durumunu yerel olarak saklaması gerekebilir.

Bir yayının alıkonduğundan emin olmak için MQPMO_RETAIN koyma iletisi seçeneğini kullanın. Bu seçenek kullanılırsa ve yayın alıkonamazsa, ileti yayınlanmaz ve çağrı MQRC_PUT_NOT_ALLOWED ile başarısız olur.

Bir ileti alıkonan bir yayınsa, bu, MQIsRetained ileti özelliğiyle gösterilir. Bir iletinin sürekliliği, ilk yayınlandığı zamanki gibi.

İlgili kavramlar

Yayınlama/abone olma kümelerinde tutulan yayınlar için tasarımla ilgili dikkat edilecek noktalar

Syncpoint altındaki yayınlar

IBM MQ yayınlama/abone olma içinde, eşitleme noktası yayıncılar tarafından ya da kuyruk yöneticisi tarafından dahili olarak kullanılabilir.

Yayıncılar, MQPMO_SYNCPOINT seçeneğiyle MQPUT/MQPUT1 çağrılarını yayınlarken eşitleme noktasını kullanır. Abonelere teslim edilen tüm iletiler, work.The MAXUMSGS kuyruk yöneticisi özniteliği bu sınırı belirler. Sınıra ulaşılırsa, yayınlayıcı 2024 (07E8) (RC2024): MQRC_SYNCPOINT_LIMIT_UY neden kodunu alır.

Bir yayınlayıcı MQPMO_NO_SYNCPOINT ile MQPMO_RETAIN seçeneğiyle MQPUT/MQPUT1 çağrılarını ya da ALL ya da ALLDUR değerleriyle NPMSGDLV/PMSGDLV konu teslim seçeneklerini kullanarak MQPMO_NO_SYNCPOINT ' i açtığında, kuyruk yöneticisi iletilerin istendiği şekilde teslim edilmesini sağlamak için iç eşitleme noktalarını kullanır. Yayınlayıcı, yayınlayıcı MQPUT/MQPUT1 çağrısı kapsamında sınıra ulaşılırsa, 2024 (07E8) (RC2024): MQRC_SYNCPOINT_LIMIT_UY neden kodunu alabilir.

Aboneler ve abonelikler

IBM MQ yayınlama/abone olma içinde, abone olma, yayınlama/abone olma ağındaki bir kuyruk yöneticisinden belirli bir konu hakkında bilgi isteyen bir uygulamadır. Bir abone, birden çok yayıncıdan aynı ya da farklı konularda ileti alabilir.

Abonelikler, bir MQSC komutu kullanılarak ya da uygulamalar tarafından el ile yaratılabilir. Bu abonelikler yerel kuyruk yöneticisine yayınlanır ve abonenin almak istediği yayınlara ilişkin bilgileri içerir:

- Abonenin ilgilendiği konu; genel arama karakterleri kullanılırsa, bu birden çok konuya çözülebilir.
- Yayınlanan iletilere uygulanacak isteğe bağlı bir seçim dizgisi.
- Seçilen yayınların yerleştirilmesi gereken bir kuyruğa ilişkin tanıtıcı (*abone kuyruğu* olarak bilinir) ve isteğe bağlı CorrelId.

Yerel kuyruk yöneticisi abonelik bilgilerini saklar ve bir yayın aldığında, yayının konusu ve seçim dizgisiyle eşleşen bir abonelik olup olmadığını belirlemek için bilgileri tarar. Eşleşen her abonelik için kuyruk yöneticisi, yayını abonenin abone kuyruğuna yönlendirir. Bir kuyruk yöneticisinin aboneliklerle ilgili olarak sakladığı bilgiler, DIS SUB ve DIS SBSTATUS komutları kullanılarak görüntülenebilir.

Abonelik yalnızca aşağıdaki olaylardan biri gerçekleştiğinde silinir:

- Abone, MQCLOSE çağrısı kullanılarak abonelikten çıkarılır (abonelik sürekli yapılmamışsa).
- Abonelik süresi sona erer.
- Abonelik, DELETE SUB komutu kullanılarak sistem yöneticisi tarafından silinir.
- Abone uygulaması sona erer (abonelik sürekli değilse).
- Kuyruk yöneticisi durdurulur ya da yeniden başlatılır (abonelik sürekli değilse).

İleti alınırken, MQGET çağrısında uygun seçenekleri kullanın. Uygulamanız iletileri yalnızca tek bir abonelik için işlerse, en az C örnek programında `amqssbxa.c` ve yönetilmeyen MQ abonesindegösterildiği gibi `get-by-correlid`komutunu kullanmanız gerekir. Kullanılacak **`CorrelId`** , MQSD ' deki MQSUB tarafından döndürülür.**`SubCorrelId`** alanı.

İlgili kavramlar

Klonlanan ve paylaşılan abonelikler

İlgili başvurular

sharedSubscription özelliğinin nasıl tanımlanacağına ilişkin örnekler

Yönetilen kuyruklar ve yayınlama/abone olma

Bir abonelik oluşturduğunuzda, yönetilen kuyruğa almayı kullanmayı seçebilirsiniz. Yönetilen kuyruğa alma işlevini kullanırsanız, abonelik yarattığınızda otomatik olarak bir abonelik kuyruğu yaratılır. Yönetilen kuyruklar, aboneliğin dayanıklılığına uygun olarak otomatik olarak toplanır. Yönetilen kuyrukların kullanılması, yayınları almak için kuyruk oluşturma konusunda endişelenmenize gerek olmadığı anlamına gelir ve sürekli olmayan bir abonelik bağlantısı kapatılırsa, kullanılmayan yayınlar otomatik olarak abone kuyruklarından kaldırılır.

Bir uygulamanın belirli bir kuyruğu abone kuyruğu olarak kullanması gerekmezse, aldığı yayınların hedefi, MQSO_MANAGED abonelik seçeneğini kullanarak *yönetilen abonelikleri* kullanabilir. Yönetilen abonelik yaratırsanız, kuyruk yöneticisi, yayınların alınacağı bir abone kuyruğuna ilişkin aboneye bir nesne tanıtıcısı döndürür. Bunun nedeni, *yönetilen aboneliklerin* IBM MQ ' in aboneliği işlediği bir abonelik olmasıdır. Kuyruğun nesne tanıtıcısı, kuyruğa göz atmanızı, kuyrukta nesne tanıtıcısı almanızı ya da sorgu yapmanızı sağlar (geçici dinamik kuyruklara belirtik olarak erişim verilmedikçe, yönetilen bir kuyruğun özniteliklerine yerleştirilemez ya da öznitelikleri ayarlanamaz).

Aboneliğin dayanıklılığı, abone olan uygulamanın kuyruk yöneticisine olan bağlantısı kesildiğinde yönetilen kuyruğun devam edip etmediğini belirler.

Yönetilen abonelikler, özellikle sürekli olmayan aboneliklerle kullanıldığında kullanışlıdır; çünkü uygulamanın bağlantısı sona erdirildiğinde, tüketilmeyen iletiler, kuyruk yöneticinizde süresiz olarak yer kaplayan abone kuyruğunda kalır. Yönetilen bir abonelik kullanıyorsanız, yönetilen kuyruk geçici bir dinamik kuyruk olur ve bu nedenle, aşağıdaki nedenlerden herhangi biri nedeniyle bağlantı kesildiğinde kullanılmayan iletilerle birlikte silinir:

- MQCO_REMOVE_SUB ile MQCLOSE kullanıldı ve yönetilen Hobj kapatıldı.
- Kalıcı olmayan bir abonelik (MQSO_NON_DURABLE) kullanılarak bir uygulamayla bağlantı kaybedildi.
- bir abonelik, süresi dolduğu ve yönetilen Hobj kapatıldığı için kaldırılır.

Yönetilen abonelikler sürekli aboneliklerle de kullanılabilir, ancak bağlantı yeniden açıldığında alınabilmeleri için abone kuyruğunda kullanılmayan iletileri bırakmak isteyebilirsiniz. Bu nedenle, sürekli abonelikler için yönetilen kuyruklar kalıcı dinamik kuyruk şeklini alır ve abone olan uygulamanın kuyruk yöneticisiyle bağlantısı kesildiğinde kalır.

Kalıcı dinamik bir yönetilen kuyruk kullanmak istiyorsanız, aboneliğinizde bir süre bitimi ayarlayabilirsiniz; böylece, bağlantı kesildikten sonra kuyruk var olmaya devam eder.

Yönetilen kuyruğu silerseniz bir hata iletisi alırsınız.

Yaratılan yönetilen kuyruklar, her birinin benzersiz olması için sondaki 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 geri 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 kuyruğunda tüketilmeyen 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 abonelikle eşleşip eşleşmediğini belirlemek için bir yayına uygulanan bir ifadedir. Seçim dizgileri genel arama karakterleri içerebilir.

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

Seçim dizgisi, her aboneye teslim edilmek üzere değiştirilmeden önce yayıncı tarafından konduğu gibi iletiye göre değerlendirilir. Yayınlama işleminin bir parçası olarak değiştirilebilecek seçim dizgisindeki alanları kullanırken dikkatli olun. Örneğin, MQMD alanları `UserIdentifier`, `MsgId`ve `CorrelId`.

Seçim dizgileri, yayınlama işleminin bir parçası olarak kuyruk yöneticisi tarafından eklenen ileti özelliği alanlarından herhangi birine gönderme yapmamalıdır (yayına ilişkin konu dizgisini içeren `MQTopicString`ileti özelliği dışında, bkz. Yayınlama/abone olma iletisi özellikleri).

İlgili kavramlar

Seçim dizgisi kuralları ve kısıtlamaları

Konular

Konu, bir yayınlama/abone olma iletisinde yayınlanan bilgilerin konusudur.

Noktadan noktaya iletişim sistemlerindeki iletiler belirli bir hedef adrese gönderilir. Konu tabanlı yayınlama/abone olma sistemlerindeki iletiler, iletinin içeriğini açıklayan konuya dayalı olarak abonelere gönderilir. İçerik tabanlı sistemlerde, iletiler iletinin içeriğine dayalı olarak abonelere gönderilir.

IBM MQ yayınlama/abone olma sistemi, konu tabanlı bir yayınlama/abone olma sistemidir. Bir yayınlayıcı bir ileti oluşturur ve bunu, yayının konusuna en uygun bir konu dizgisiyle yayınlar. Yayınları almak için bir abone, yayın konularını seçmek üzere kalıp eşleştirme konu dizesi içeren bir abonelik oluşturur. Kuyruk yöneticisi, yayınları, yayın konusuyla eşleşen abonelikleri olan abonelere teslim eder ve yayınları alma yetkisine sahiptir. “Konu dizgileri” sayfa 66başlıklı makalede, bir yayının konusunu tanımlayan konu dizgilerinin sözdizimi açıklanmaktadır. Aboneler, alınacak konuları seçmek için konu dizgileri de oluşturur. Abonelerin yaratmış olduğu konu dizeleri, yayınlardaki konu dizgileriyle eşleşecek iki alternatif joker karakter şemasını içerebilir. Kalıp eşleştirme “Genel arama karakteri şemaları” sayfa 67içinde açıklanmıştır.

Konu tabanlı yayınlama/abone olma durumunda, yayıncılar veya yöneticiler konuları konular halinde sınıflandırmaktan sorumludur. Genellikle konular, konu dizgisinde alt konular oluşturmak için `'/'` karakteri kullanılarak hiyerarşik olarak konu ağaçları halinde düzenlenir. Konu ağaçlarına ilişkin örnekler için bkz. “Konu ağaçları” sayfa 72 . Konular, konu ağacındaki düğümlerdir. Konular, alt konuları olmayan yaprak düğümler ya da alt konuları olan ara düğümler olabilir.

Konuları sıradüzensel bir konu ağacında düzenlemeye paralel olarak, konuları yönetim konusu nesneleriyle ilişkilendirebilirsiniz. Bir konuya öznitelikleri (örneğin, konunun bir kümede dağıtılıp dağıtılmayacağı), bir denetim konusu nesnesiyle ilişkilendirerek atarsınız. İlişki, denetim konusu nesnesinin `TOPICSTR` özniteliği kullanılarak adlandırılarak yapılır. Bir denetim konusu nesnesini bir konuyla belirtik olarak ilişkilendirmezseniz, konu, yönetimle ilgili bir konu nesnesiyle *ilişkilendirdiğiniz* konu ağacındaki en yakın üst nesnesinin özniteliklerini devralır. Herhangi bir üst konu tanımlamadıysanız, üst konu `SYSTEM.BASE.TOPIC`. Yönetimle ilgili konu nesneleri “Yönetim konusu nesneleri” sayfa 73içinde açıklanmıştır.

Not: Bir konunun tüm özniteliklerini `SYSTEM.BASE.TOPIC`, `SYSTEM.BASE.TOPIC`. Örneğin, ABD durumlarının konu alanında, `USA/Alabama` `USA/Alaska`, vb., USA kök konudur. Kök konunun ana amacı, yanlış aboneliklerle eşleşen yayınları önlemek için ayrık, örtüşmeyen konu alanları oluşturmaktır. Bu, kök konunuzun özniteliklerini tüm konu alanınızı etkileyecek şekilde değiştirebileceğiniz anlamına da gelir. Örneğin, **CLUSTER** özniteliği için ad ayarlayabilirsiniz.

Bir konuya yayınlayıcı ya da abone olarak başvurduğunuzda, bir konu dizgisi belirtmeyi ya da bir konu nesnesine gönderme yapma seçeneğiniz vardır. Ya da her ikisini de yapabilirsiniz; bu durumda, sağladığınız konu dizgisi konu nesnesinin bir alt konusunu tanımlar. Kuyruk yöneticisi, konu dizgisini konu nesnesinde adı belirtilen konu dizgisi önekine ekleyerek konuyu tanıtır ve iki konu dizgisinin arasına ek bir `'/'` ekler; örneğin, *konu dizgisi/nesne dizgisi*. “Konu dizgilerini birleştirme” sayfa 71 içinde bu konuda

daha ayrıntılı bilgi vardır. Sonuçtaki konu dizgisi, konuyu tanımlamak ve bir yönetim konusu nesnesiyle ilişkilendirmek için kullanılır. Yönetim konusu nesnesinin, ana konuya karşılık gelen konu nesnesiyle aynı konu nesnesi olması gerekmez.

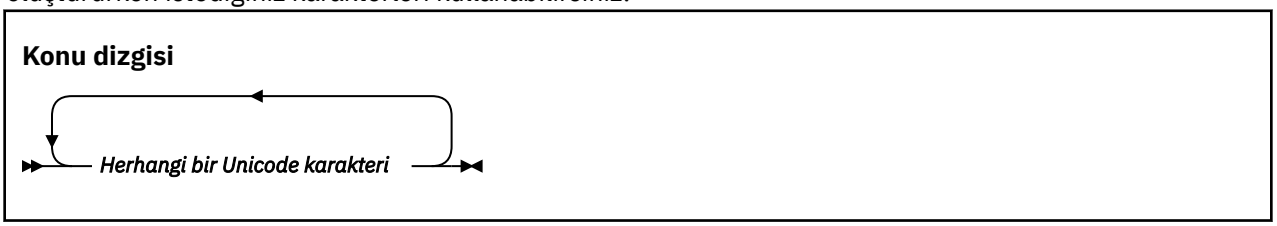
İçerik tabanlı yayınlama/abone olma içinde, her iletinin içeriğinde arama yapan seçim dizgileri sağlayarak hangi iletileri almak istediğinizi tanımlayabilirsiniz. IBM MQ , iletinin tüm içeriği yerine ileti özelliklerini tarayan ileti seçicileri kullanarak, içeriğe dayalı yayınlama/abone olma işlevlerinin ara biçimini sağlar (bkz. Seçiciler). İleti seçicilerin arketipik kullanımı, bir konuya abone olmak ve daha sonra, seçimi sayısal bir özellikle nitelemek için kullanılır. Seçici, yalnızca belirli bir aralıktaki değerlerle ilgilendiğinizi belirtmenizi sağlar; karakter ya da konu tabanlı genel arama karakterleri kullanarak yapamazsınız. İletinin tüm içeriğine dayalı olarak süzgeç uygulamanız gerekiyorsa, IBM Integration Busöğesini kullanmanız gerekir.

Konu dizgileri

Konu dizgisi kullanarak konu olarak yayınladığınız bilgileri etiketleyin. Karakter ya da konu tabanlı genel arama karakteri konu dizgilerini kullanarak konu gruplarına abone olun.

Konular

Konu dizgisi , yayınlama/abone olma iletisinin konusunu tanımlayan bir karakter dizgisidir. Bir konu dizgisi oluştururken istediğiniz karakterleri kullanabilirsiniz.



IBM WebSphere MQ 7 yayınlama/abone olma içinde üç karakterin özel anlamı vardır. Bir konu dizgisinin herhangi bir yerinde bunlara izin verilir, ancak bunları dikkatli kullanın. Özel karakterlerin kullanımı “Konuya dayalı genel arama karakteri şeması” sayfa 67içinde açıklanmıştır.

Eğik çizgi (/)

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

Yapabilirseniz, boş konu düzeylerinden ('//') kaçının. Bunlar, konu dizesi olmadan konu sıradüzenindeki düğümlere karşılık gelir. Bir konu dizgisindeki baştaki ya da sondaki '/' , baştaki ya da sondaki boş düğüme karşılık gelir ve bundan da kaçınılmalıdır.

Hash işareti (#)

Aboneliklerde çok düzeyli bir joker karakter oluşturmak için '/' ile birlikte kullanılır. Yayınlanan konuları adlamak için kullanılan konu dizelerindeki '#' bitişik '/' öğesini kullanarak dikkatli olun. “Konu dizgileri örnekleri” sayfa 67 içinde '#'kullanımı gösterilmektedir.

'.../#/...', '#/...' ve '.../#' dizgilerinin abonelik konusu dizgilerinde özel bir anlamı vardır. Dizgiler, konu sıradüzenindeki bir ya da daha fazla düzeydeki tüm konularla eşleşiyor. Bu nedenle, bu sıralardan biriyle bir konu oluşturduysanız, konu sıradüzenindeki birden çok düzeydeki tüm konulara abone olmadan konuya abone olamazdınız.

Artı işareti (+)

Aboneliklerde tek düzeyli genel arama karakteri oluşturmak için '/' ile birlikte kullanılır. Yayınlanan konuları adlamak için kullanılan konu dizelerindeki '+' bitişik '/' öğesini kullanarak dikkatli olun.

'.../+/...', '+/...' ve '.../+' dizgilerinin abonelik konusu dizgilerinde özel bir anlamı vardır. Dizgiler, konu sıradüzenindeki bir düzeydeki tüm konularla eşleşiyor. Bu nedenle, bu sıralardan biriyle bir konu oluşturduysanız, konu sıradüzeninde bir düzeydeki tüm konulara abone olmadan konuya abone olamazdınız.

Konu dizgileri örnekleri

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

İlgili başvurular

Konu

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

MQSO_WILDCARD_TOPIC
Konuya dayalı genel arama karakteri şemasını kullanarak abone olunacak konuları seçin.

Belirtik olarak bir genel arama karakteri şeması seçilmezse, bu varsayılan değerdir.

MQSO_WILDCARD_CHAR
Karakter tabanlı genel arama karakteri şemasını kullanarak abone olunacak konuları seçin.

DEFINE SUB komutunda **wschema** parametresini belirterek her iki şemayı da ayarlayın. Daha fazla bilgi için bkz. DEFINE SUB.

Not: IBM WebSphere MQ 7.0 ' den önce oluşturulan abonelikler her zaman karakter tabanlı genel arama karakteri şemasını kullanır.

Örnekler

```
IBM/+/Results
#/Results
IBM/Software/Results
IBM/*ware/Results
```

Konuya dayalı genel arama karakteri şeması
Konu tabanlı joker karakterler, abonelerin bir kerede birden çok konuya abone olmalarına izin verir.

Konu tabanlı joker karakterler, IBM MQ yayınlama/abone olma içinde konu sisteminin güçlü bir özelliğidir. Abonelikler için çok düzeyli joker karakter ve tek düzeyli joker karakter kullanılabilir, ancak bunlar bir iletinin yayıncısı tarafından bir konu içinde kullanılamaz.

Konu tabanlı genel arama karakteri şeması, konu düzeyine göre gruplanmış yayınları seçmenizi sağlar. Konu sıradüzenindeki her düzey için, o konu düzeyine ilişkin abonelikteki dizginin yayındaki dizgiyle tam olarak eşleşip eşleşmeyeceğini seçebilirsiniz. Örneğin, `IBM/+/Results` aboneliği tüm konuları,

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

İki tip joker karakter vardır.

Çok düzeyli genel arama karakteri

- Aboneliklerde çok düzeyli joker karakter kullanılır. Bir yayında kullanıldığında, hazır bilgi olarak kabul edilir.
- `'#'` çok düzeyli genel arama karakteri, bir konu içindeki herhangi bir sayıda düzeyle eşleşmek için kullanılır. Örneğin, örnek konu ağacını kullanarak `'USA/Alaska/#''` e abone olursanız, `'USA/Alaska'` ve `'USA/Alaska/Juneau'` konularındaki iletileri alırsınız.
- Çok düzeyli joker karakter sıfır ya da daha fazla düzeyi temsil edebilir. Bu nedenle `'USA/#'` , `'#'` 'un sıfır düzeylerini temsil ettiği tekil `'USA'` ile de eşleşebilir. Ayrılacak 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 geçerlidir. Bu nedenle, `'#'` ve `'USA/#'` , `'#'` karakterinin genel arama

karakteri olarak değerlendirildiği geçerli konulardır. Ancak, 'USA#' geçerli bir konu dizgisi olsa da, '#' karakteri genel arama karakteri olarak kabul edilmez ve özel bir anlamı yoktur. Ek bilgi için bkz. “Konuya dayalı genel arama karakterleri genel arama karakteri olmadığında” sayfa 69 .

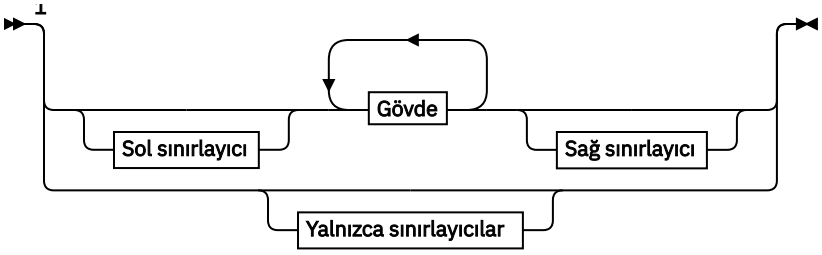
Tek düzeyli genel arama karakteri

- Aboneliklerde tek joker karakter kullanılır. Bir yayında kullanıldığında, hazır bilgi olarak kabul edilir.
- Tek düzeyli genel arama karakteri '+' , tek bir konu düzeyiyle eşleşir. Örneğin, 'USA/+' , 'USA/Alabama/Auburn'ile değil, 'USA/Alabama'ile eşleşir. Tek düzeyli joker karakter yalnızca tek bir düzeyle eşleştiği için 'USA/+' , 'USA'ile eşleşmez.
- Tek düzeyli joker karakter, konu ağacındaki herhangi bir düzeyde ve çok düzeyli joker karakterle birlikte kullanılabilir. Tek düzeyli genel arama karakteri, kendi başına belirtilmesi dışında, konu düzeyi ayırıcının yanında belirtilmelidir. Bu nedenle, '+' ve 'USA/+' , '+' karakterinin genel arama karakteri olarak değerlendirildiği geçerli konulardır. Ancak, 'USA+' geçerli bir konu dizgisi olsa da, '+' karakteri genel arama karakteri olarak kabul edilmez ve özel bir anlamı yoktur. Ek bilgi için bkz. “Konuya dayalı genel arama karakterleri genel arama karakteri olmadığında” sayfa 69 .

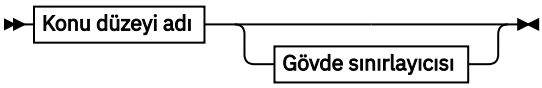
Konuya dayalı genel arama karakteri şemasının sözdiziminde çıkış karakteri yok. '#' ve '+' genel arama karakteri olarak değerlendirilip değerlendirilmeyeceği bağlamına bağlıdır. Ek bilgi için bkz. “Konuya dayalı genel arama karakterleri genel arama karakteri olmadığında” sayfa 69 .

Not: Bir konu dizgisinin başı ve sonu özel bir şekilde işlenir. Dizginin sonunu belirtmek için '$' kullanılması, '$#/...' çok düzeyli bir genel arama karakteridir ve '$/#/..'. çok düzeyli bir joker karakterin izlediği, kökteki boş bir düğümdür.

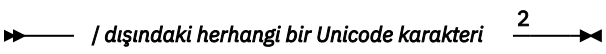
Konuya dayalı genel arama karakteri dizgisi



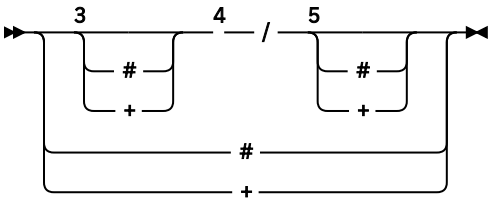
Gövde



Konu düzeyi adı



Yalnızca sınırlayıcılar



Notlar:

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

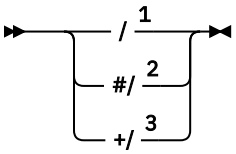
[2] Karakter tabanlı ve konu tabanlı genel arama karakteri şemaları arasındaki uyumluluk için düzey adı dizgilerinde *, ?, %'i kullanmamanız önerilir.

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

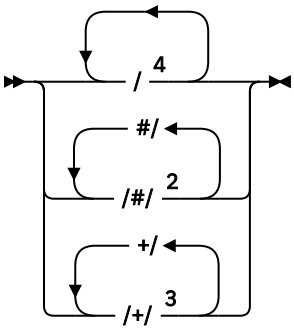
[4] Joker karakteri olmayan / , tek bir boş konuyla eşleşir.

[5] Bu durumlar, *sağ sınırlayıcı* örüntüsüne eşdeğerdir.
[6] Her konuyu eşleştirin.
[7] Yalnızca bir düzey olan her konuyu eşleştirin.

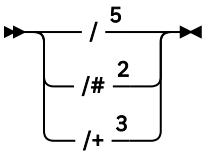
Sol sınırlayıcı



Gövde sınırlayıcı



Sağ sınırlayıcı



Notlar:

[1] Konu dizgisi boş bir konuyla başlıyor
[2] Sıfır ya da daha fazla düzeyle eşleşir. Birden çok çok çok düzeyli eşleşme dizgisi, bir çok düzeyli eşleşme dizgisiyle aynı etkiye sahiptir.
[3] Tam olarak bir düzeyle eşleşir.
[4] / / boş bir konudur-konu dizgisi olmayan bir konu nesnesidir.
[5] Konu dizgisi boş bir konuyla bitiyor

Konuya dayalı genel arama karakterleri genel arama karakteri olmadığında

Bir konu düzeyinde diğer karakterlerle (kendileri dahil) karıştırıldığında, '+' ve '#' genel arama karakterlerinin özel bir anlamı yoktur.

Bu, '+' ya da '#' dizgisini 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 genel arama karakteri olarak kabul edilir ve bu nedenle, yayınlanacak ancak bir abonelikte geçerli olan bir konu dizgisinde geçerli değildir.

İkinci örnekte, '+' ve '#' karakterleri genel arama karakteri olarak kabul edilmez ve bu nedenle 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 karakteri şeması, geleneksel karakter eşleştirmesine dayalı olarak konuları seçmenizi sağlar.

'*'dizgisini kullanarak bir konu sıradüzenindeki birden çok düzeydeki tüm konuları seçebilirsiniz. Karakter tabanlı genel arama karakteri şemasında '*' kullanılması, konuya dayalı genel arama karakteri dizgisinin kullanılmasıyla eşdeğerdir '#'

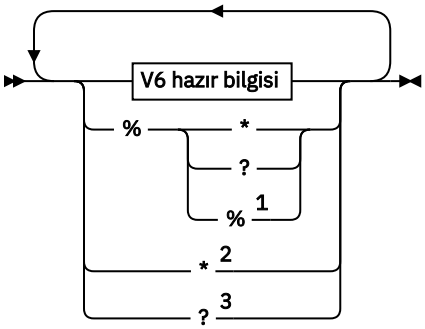
' *x*/*/*y* ' , konuya dayalı şemadaki ' *x*/#/*y* ' ile eşdeğerdir ve ' *x* ile *y* 'düzeyleri arasındaki konu sıradüzenindeki tüm konuları seçer; burada ' *x* ' ve ' *y* ' , genel arama karakterinin döndürdüğü düzeyler kümesinde olmayan konu adlarıdır.

Konu tabanlı şemada '/+/' , karaktere dayalı şemada tam olarak eşdeğer bir şemaya sahip değildir. 'IBM/*/Results' , 'IBM/Patents/Software/Results'seçeneğini de belirlerdi. Yalnızca sıradüzenin her düzeyindeki konu adları kümesi benzersizse, her zaman aynı eşleşmeleri veren iki şemayla sorgular oluşturabilirsiniz.

Karaktere dayalı şemada genel bir şekilde kullanılan '*' ve '?' , konuya dayalı şemada eşdeğerlere sahip değildir. Konuya dayalı şema, joker karakter kullanarak kısmi eşleştirme gerçekleştirmez. Karaktere dayalı genel arama karakteri aboneliği 'IBM/*ware/Results' konu tabanlı eşdeğerine sahip değil.

Not: Genel arama karakteri abonelikleri kullanan eşleşmeler, konu tabanlı abonelikler kullanan eşleşmelerden daha yavaştır.

Karaktere dayalı genel arama karakteri dizgisi



V6 hazır bilgisi

Notlar:

[1] Bir hazır bilgi olarak kabul edilmesi için aşağıdaki karakterin çıkış karakteri olarak kabul edilmesi anlamına gelir. '%' 'yi '*', '?' ya da '%'izlemelidir. Bkz. “Konu dizgileri örnekleri” sayfa 67.

[2] Bir abonelikte sıfır ya da daha fazla karakterin eşleşmesi anlamına gelir.

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

Örnekler

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

Konu dizgilerini birleştirme

Abonelikler oluştururken ya da ileti yayınlayabilmek için konuları açarken, konu dizgisi iki ayrı alt konu dizgisi ya da "alt konu"birleştirilerek oluşturulabilir. Bir alt konu, uygulama ya da yönetim komutu tarafından konu dizgisi olarak, diğeri ise bir konu nesnesiyle ilişkili konu dizesi olarak sağlanır. Alt konuyu kendi başına konu dizgisi olarak kullanabilir ya da yeni bir konu adı oluşturmak için bunları birleştirebilirsiniz.

Örneğin, **DEFINE SUB**MQSC komutunu kullanarak bir abonelik tanımladığınızda, komut **TOPICSTR** (konu dizgisi) ya da **TOPICOBJ** (konu nesnesi) öğesini öznitelik olarak ya da her ikisini birlikte alabilir. Yalnızca **TOPICOBJ** sağlanırsa, o konu nesnesiyle ilişkili konu dizgisi konu dizgisi olarak kullanılır. Yalnızca **TOPICSTR** sağlanırsa, konu dizgisi olarak kullanılır. Her ikisi de sağlanırsa, bunlar **TOPICOBJ** / **TOPICSTR**biçiminde tek bir konu dizgisi oluşturmak üzere birleştirilir; burada **TOPICOBJ** yapılandırılan konu dizgisi her zaman önce gelir ve dizginin iki bölümü her zaman bir "/" karakteriyle ayrılır.

Benzer şekilde, bir MQI programında tam konu adı MQOPENtarafından oluşturulur. Yayınlama/abone olma MQI çağrılarında, listelenen sırada kullanılan iki alandan oluşur:

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

Sonuçtaki konu dizgisi **ResObjectString** değiştirgesinde döndürülür.

Her alanın ilk karakteri boşluk ya da boş karakter değilse ve alan uzunluğu sıfırdan büyükse, bu alanların var olduğu varsayılır. Alanlardan yalnızca biri varsa, konu adı olarak değiştirilmeden kullanılır. Her iki alanda da değer yoksa, arama MQRC_UNKNOWN_OBJECT_NAMEneden koduyla başarısız olur ya da tam konu adı geçerli değilse MQRC_TOPIC_STRING_ERROR .

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

Aşağıdaki çizelge, konu dizgisi bitiştirme örneklerini göstermektedir:

Çizelge 2. Konu dizgi bitiştirme örnekleri

Konu nesnesinin TOPICSTR	**Uygulama ya da DEFINE SUB komutu tarafından sağlanan konu dizgisi**	**Tam konu adı**	**Yorum**
Futbol/Puanlar	''	Futbol/Puanlar	Konu nesnesinin TOPICSTR 'si tek başına kullanılır.
''	Futbol/Puanlar	Futbol/Puanlar	ObjectString/TOPICSTR tek başına kullanılır.
Futbol	Puanlar	Futbol/Puanlar	Birleştirme noktasına bir "/" karakteri eklenir.
Futbol	/Puanlar	Futbol//Puanlar	İki dizgi arasında bir 'boş düğüm' üretilir. Bu, "Football/Scores"(Futbol/Puan) seçeneğinden farklıdır.
/Futbol	Puanlar	/Futbol/Puanlar	Konu 'boş düğüm' ile başlıyor. Bu, "Football/Scores"(Futbol/Puan) seçeneğinden farklıdır.

"/" karakteri,“Konu ağaçları” sayfa 72içinde tam konu adına yapı sağlayan özel bir karakter olarak kabul edilir. Konu ağacının yapısı etkilendiği için "/" karakteri başka bir nedenle kullanılmamalıdır. "/Football" konusu, "Football"konusuyla aynı değil.

Not: Abonelik oluştururken bir konu nesnesi kullanırsanız, konu nesnesi konu dizgisinin değeri, abonelikte tanımlama sırasında düzeltilebilir. Konu nesnesinde yapılan sonraki değişiklikler, aboneliğin tanımlandığı konu dizgisini etkilemez.

Konu dizelerindeki genel arama karakterleri

Aşağıdaki genel arama karakterleri özel karakterlerdir:

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

Genel arama karakterleri yalnızca abonelik tarafından kullanıldığında özel anlam taşır. Bu karakterler başka bir yerde kullanıldığında geçersiz kabul edilmez, ancak bunların nasıl kullanıldığını anladığınızdan emin olmanız gerekir ve konu nesnelerini yayınlarken ya da tanımlarken konu dizgilerinizde bu karakterleri kullanmamayı tercih edersiniz.

ya da + ile bir konu dizgisinde, bir konu düzeyinde diğer karakterlerle (kendileri dahil) karıştırılmış olarak yayınlarsanız, konu dizgisine genel arama karakteri şemasından biriyle abone olunabilir.

İki / karakteri arasındaki tek karakter olarak # ya da + ile bir konu dizgisinde yayınlarsanız, konu dizgisi MQSO_WILDCARD_TOPIC joker şemasını kullanan bir uygulama tarafından belirtik olarak abone olunamaz. Bu durum, uygulamanın beklenenden daha fazla yayın almasıyla sonuçlanır.

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

Örnek kod parçacığı

Örnek 2: Bir değişken konuda yayınlayıcıörnek programından alınan bu kod parçacığı, bir konu nesnesini 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

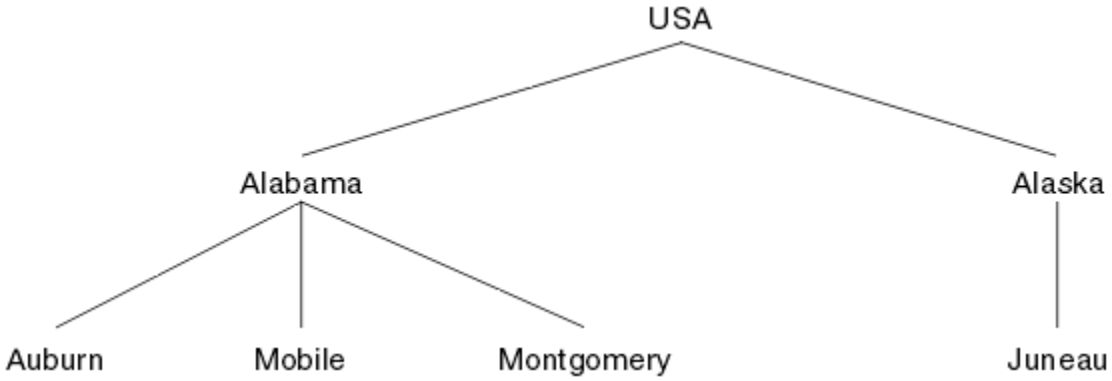
Tanımladığınız her konu, konu ağacındaki bir öğe ya da düğümdür. Konu ağacı, MQSC ya da PCF komutları kullanılarak daha önce tanımlanmış konularla başlamak için boş olabilir ya da konuları içerebilir. Konu yaratma komutlarını kullanarak ya da bir yayında ya da abonelikte ilk kez konuyu belirterek yeni bir konu tanımlayabilirsiniz.

Bir konunun konu dizgisini tanımlamak için herhangi bir karakter dizgisi kullanabilirsiniz, ancak sıradüzenli bir ağaç yapısına sığacak bir konu dizgisi seçmeniz önerilir. Konu sokmaları ve konu ağaçlarının düşünceli tasarımı, aşağıdaki işlemlerde size yardımcı olabilir:

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

Bir konu ağacını düz, doğrusal bir yapı olarak oluşturabilseniz de, bir veya daha fazla kök konu içeren hiyerarşik bir yapıda bir konu ağacı oluşturmak daha iyidir. Güvenlik planlaması ve konularla ilgili daha fazla bilgi için Güvenliğin yayınlanması/abone olunmasıbaşlıklı konuya bakın.

Şekil 18 sayfa 73 içinde tek bir kök konuyu içeren bir konu ağacı örneği gösterilmektedir.



Şekil 18. Konu ağacı örneği

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

Şekil 18 sayfa 73 içinde gösterilen konu ağacındaki geçerli konular şunlardır:

"ABD"
"USA/Alabama"
"ABD/Alaska"
"ABD/Alabama/Auburn"
"USA/Alabama/Mobil"
"ABD/Alabama/Montgomery"
"ABD/Alaska/Juneau"

Konu dizgileri ve konu ağaçları tasarladığınızda, kuyruk yöneticisinin konu dizgisini yorumlamadığını ya da konu dizgisinden anlam türetme girişiminde bulunmadığını unutmayın. Konu dizgisini, seçilen iletileri o konunun abonelerine göndermek için kullanır.

Aşağıdaki ilkeler bir konu ağacının inşası ve içeriği için geçerlidir:

- Bir konu ağacındaki düzey sayısı için sınır yoktur.
- Konu ağacındaki bir düzeyin adının uzunluğuna ilişkin bir sınır yoktur.
- Herhangi bir sayıda "kök" düğüm olabilir; başka bir ifadeyle, herhangi bir sayıda konu ağacı olabilir.

İlgili görevler
Konu ağacında istenmeyen konuların sayısını azaltma

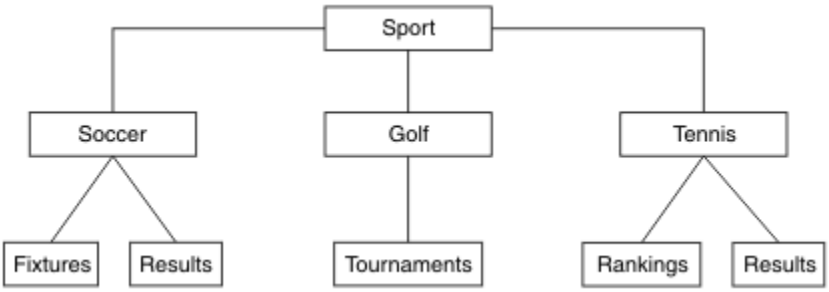
Yönetim konusu nesneleri

Bir denetim konusu nesnesini kullanarak, konulara belirli, varsayılan olmayan öznitelikler atayabilirsiniz.

Şekil 19 sayfa 73 , `Sport` üst düzey konusunun farklı sporları kapsayan ayrı konulara nasıl bölündüğünü konu ağacı olarak nasıl görselleştirildiğini gösterir:

Şekil 19. Konu ağacının görselleştirilmesi

Şekil 20 sayfa 74 içinde, her bir sporla ilgili farklı bilgi tiplerini ayırmak için konu ağacının nasıl daha fazla bölünebildiği gösterilmektedir:



Şekil 20. Genişletilmiş konu ağacı

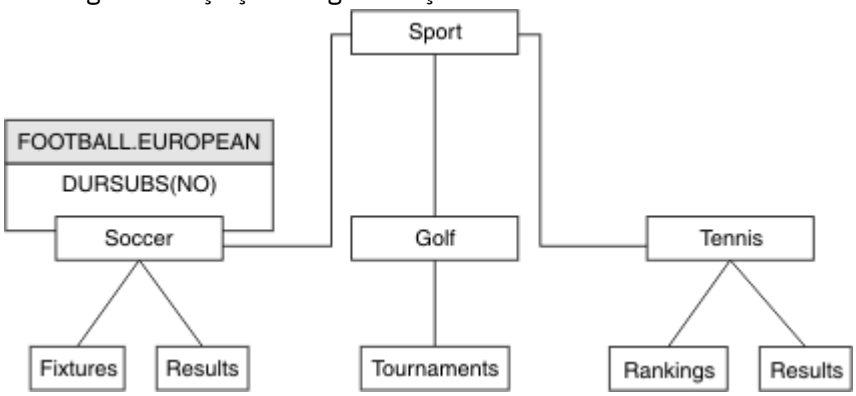
Gösterilen konu ağacını yaratmak için yönetim konusu nesneleri tanımlanmasına gerek yoktur. Bu ağaçtaki düğümlerin her biri, yayınlama ya da abone olma işleminde yaratılan bir konu dizgisi tarafından tanımlanır. Ağaçtaki her konu, özniteliklerini üst öğesinden devralır. Varsayılan olarak tüm öznitelikler ASPARENTolarak ayarlandığından, öznitelikler üst konu nesnesinden devralınır. Bu örnekte, her konu `Sport` konusuyla aynı özniteliklere sahiptir. `Sport` konusunun denetim konusu nesnesi yoktur ve özniteliklerini SYSTEM.BASE.TOPIC.

SYSTEM.BASE.TOPIC. Bu nedenle, yetkililere bu seviyede yetki vererek, tüm ağaca yetki veriyorsunuz. Sıradüzende alt konu düzeyinde yetki vermeniz gerekir.

Denetim konusu nesneleri, konu ağacındaki belirli düğümlere ilişkin belirli öznitelikleri tanımlamak için kullanılabilir. Aşağıdaki örnekte, futbol konusunun DURSUB sürekli abonelikler özelliğini NOdeğerine ayarlamak için yönetim konusu nesnesi tanımlanmıştır:

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

Konu ağacı artık şu şekilde görselleştirilebilir:



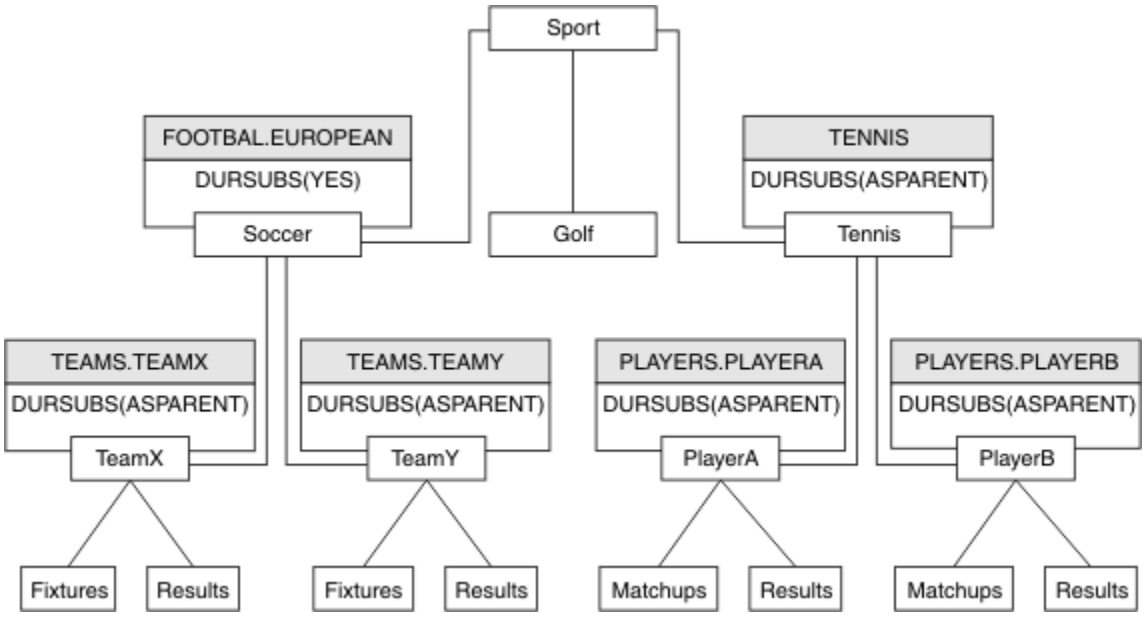
Şekil 21. Spor/Futbol konusuyla ilişkili bir yönetim konusu nesnesinin görselleştirilmesi

Ağaçta Futbol altındaki konulara abone olan uygulamalar, yönetim konusu nesnesi eklenmeden önce kullandıkları konu dizgilerini kullanmaya devam edebilir. Ancak artık bir uygulama, `/Sport/Soccer`dizgisi yerine `FOOTBALL.EUROPEAN`nesne adı kullanılarak abone olmak üzere yazılabilir. Örneğin, `/Sport/Soccer/Results`'a abone olmak için bir uygulama `MQSD.ObjectName` ' i `FOOTBALL.EUROPEAN` ve `MQSD.ObjectString` ' yi `Results`olarak belirtebilir.

Bu özellikle, konu ağacının bir kısmını uygulama geliştiricilerinden gizleyebilirsiniz. Konu ağacındaki belirli bir düğümde bir denetim konusu nesnesi tanımlayın; böylece, uygulama geliştiricileri kendi konularını düğümün alt öğeleri olarak tanımlayabilir. Geliştiriciler üst konu hakkında bilgi almalıdır, ancak üst ağaçtaki diğer düğümler hakkında bilgi almamalıdır.

Öznitelikleri devralma

Bir konu ağacının çok sayıda denetim konusu nesnesi varsa, her denetim konusu nesnesi varsayılan olarak, öznitelikleri en yakın üst yönetim konusundan devralır. Önceki örnek Şekil 22 sayfa 75içinde genişletilmiştir:



Şekil 22. Birkaç denetim konusu nesnesi içeren konu ağacı

Örneğin, `/Sport/Soccer` özelliğinin tüm alt konularına aboneliklerin kalıcı olmadığı alt konular vermek için kalıtımı kullanın. `FOOTBALL.EUROPEAN` özniteliğinin `DURSUB` özniteliğini `NO`olarak değiştirin.

Bu öznitelik aşağıdaki komut kullanılarak ayarlanabilir:

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

`Sport/Soccer` alt konularının tüm yönetim konusu nesnelerinin `DURSUB` özelliği, `ASPARENT`varsayılan değerine ayarlıdır. `FOOTBALL.EUROPEAN` özelliğinin `DURSUB` özellik değerini `NO`olarak değiştirdikten sonra, `Sport/Soccer` alt konuları `DURSUB` özellik değerini `NO`devralır. `Sport/Tennis` nesnesinin tüm alt konuları `DURSUB` değerini `SYSTEM.BASE.TOPIC` nesnesinden devralır. `SYSTEM.BASE.TOPIC` , `YES`değerine sahiptir.

`Sport/Soccer/TeamX/Results` konusuna sürekli abonelik yapmaya çalışılması şimdi başarısız olur; ancak, `Sport/Tennis/PlayerB/Results` ' a sürekli abonelik yapmaya çalışılması başarılı olur.

WILDCARD özelliğiyle genel arama karakteri kullanımını denetleme

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

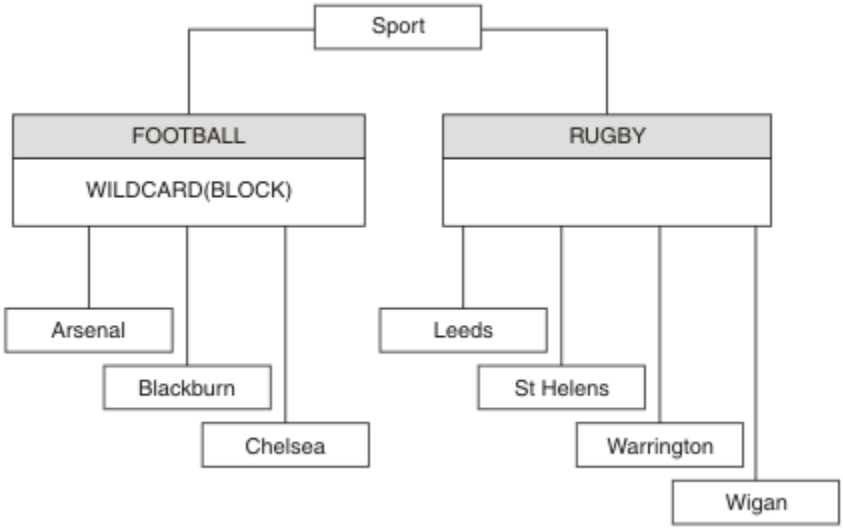
Genel arama karakteri
Bu konuya ilişkin genel arama karakteri aboneliklerinin davranışı.

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.

"Örnek: Sport yayınlama/abone olma kümesinin yaratılması" sayfa 79örneğinde, Şekil 23 sayfa 76içinde gösterilen konu ağacı yapısını oluşturmak için aşağıdaki adımları izleyebilirsiniz.



Şekil 23. WILDCARD özelliğini kullanan bir konu ağacı, BLOCK

joker konu dizgisini kullanan bir abone, `Sport` konusuna ve `Sport/Rugby` alt ağacına ilişkin tüm yayınları alır. `Sport/Football` konusunun `WILDCARD` özellik değeri `BLOCK`olduğundan, abone `Sport/Football` alt ağacına hiçbir yayın almıyor.

PASSTHRU varsayılan ayardır. `WILDCARD` özellik değerini PASSTHRU `Sport` ağacındaki düğümlere ayarlayabilirsiniz. Düğümlerde `WILDCARD` özellik değeri `BLOCK`yoksa, PASSTHRU ayarı, abonelerin `Sports` ağacındaki düğümlere gözlemlediği davranışı değiştirmez.

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

```
pub QMA
```

Şekil 24. QMA ' da Yayınla

Sonuçlar Çizelge 3 sayfa 76içinde gösterilir. `WILDCARD` özellik değerinin `BLOCK`ayarlanmasının, joker karakter içeren aboneliklerin genel arama karakteri kapsamındaki konularla ilgili yayınları almasını nasıl önlediğine dikkat edin.

Çizelge 3. QMA tarihinde alınan yayınlar

Abonelik	Konu dizgisi	Alınan yayınlar	Notlar

SPORTS	`Sports/#`	`Sports`
		`Sports/Rugby`
		`Sports/Rugby/Leeds`
			Futbol alt ağacına ilişkin tüm yayınlar `Sports/Football` üzerinde `WILDCARD(BLOCK)` tarafından engellendi

Çizelge 3. QMA tarihinde alınan yayınlar (devamı var)			
Abonelik	**Konu dizgisi**	**Alınan yayınlar**	**Notlar**
SARSENAL	Sports/#/Arsenal	-	WILDCARD(BLOCK) on Sports/Football , Arsenal üzerinde genel arama karakteri aboneliğini engeller
SLEEDS	Sports/#/Leeds	Sports/Rugby/Leeds	Sports/Rugby üzerinde varsayılan WILDCARD , Leedsüzerinde genel arama karakteri aboneliğini engellemez.

Not:

Bir aboneliğin, WILDCARD özellik değeriyle BLOCKbir konu nesnesiyle eşleşen bir genel arama karakteri olduğunu varsayın. Aboneliğin, eşleşen genel arama karakterinin sağında da bir konu dizgisi varsa, abonelik hiçbir zaman bir yayın almaz. Engellenmeyen yayınlar, engellenen joker karakterin üst karakterleri olan konulara ilişkin yayınlardır. BLOCK özellik değerine sahip konunun alt öğeleri olan konulara ilişkin yayınlar, genel arama karakteri tarafından engellenir. Bu nedenle, joker karakterin sağındaki bir konuyu içeren abonelik konusu dizgileri hiçbir zaman eşleşecek herhangi bir yayın almaz.

WILDCARD özellik değerinin BLOCK olarak ayarlanması, genel arama karakterleri içeren bir konu dizgisi kullanarak abone olamayacağınız anlamına gelmez. Böyle bir abonelik normal. Aboneliğin, WILDCARD özellik değerine BLOCKsahip bir konu nesnesiyle konuyu eşleyen belirtik bir konusu vardır. WILDCARD özellik değeri BLOCKile konunun üst ya da alt öğeleri olan konular için genel arama karakterleri kullanır. Şekil 23 sayfa 76örneğinde, Sports/Football/# gibi bir abonelik yayınları alabilir.

Genel arama karakterleri ve küme konuları

Küme konusu tanımlamaları, bir kümedeki her kuyruk yöneticisine yayılır. Kümede bir kuyruk yöneticisinde bir küme konusuna abonelik, kuyruk yöneticisinin yetkili sunucu abonelikleri yaratmasıyla sonuçlanır. Kümedeki diğer her kuyruk yöneticisinde bir yetkili sunucu aboneliği yaratılır. Genel arama karakterleri içeren konu dizelerini kullanan abonelikler, küme konularıyla birleştirildiğinde, davranışı tahmin etmek zor olabilir. Davranış aşağıdaki örnekte açıklanmıştır.

"Örnek: Sport yayınlama/abone olma kümesinin yaratılması" sayfa 79örneği için ayarlanan kümede QMB , QMAile aynı abonelik kümesine sahiptir, ancak QMB , yayıncı QMA' e yayınladıktan sonra herhangi bir yayın almadı, bkz. Şekil 24 sayfa 76. Sports/Football ve Sports/Rugby konuları küme konuları olsa da, fullsubs.tst içinde tanımlanan abonelikler bir küme konusuna başvurmaz. QMB 'den QMA' a herhangi bir yetkili sunucu aboneliği dağıtılmadı. Yetkili sunucu abonelikleri olmadan, QMA 'e ilişkin hiçbir yayın QMB' e iletilmez.

Sports/#/Leedsgibi aboneliklerin bazıları bir küme konusuna başvurmuş gibi görünebilir (bu durumda Sports/Rugby). Sports/#/Leeds aboneliği aslında SYSTEM.BASE.TOPICkonu nesnesine çözülür.

Sports/#/Leeds gibi bir abonelik tarafından başvurulan konu nesnesini çözme kuralı aşağıdaki gibidir. Konu dizgisini ilk genel arama karakterine kadar kısaltın. İlişkili bir yönetim konusu nesnesine sahip ilk konuyu aramak için konu dizesini tarayın. Konu nesnesi bir küme adı belirtebilir ya da yerel bir konu nesnesi tanımlayabilir. Sports/#/Leedsörneğinde, kesildikten sonra konu dizgisi Sportsolur; konu nesnesi yoktur ve Sports/#/Leeds , yerel konu nesnesi olan SYSTEM.BASE.TOPICnesnesinden devralır.

Kümelenmiş konulara abone olmanın joker karakter yayılımının çalışma şeklini nasıl değiştirebileceğini görmek için toplu iş komut dosyasını (upsubs.bat) çalıştırın. Komut dosyası abonelik kuyruklarını temizler ve fullsubs.tsticinde küme konusu aboneliklerini ekler. Bir yayın kümesi oluşturmak için puba.bat dosyasını yeniden çalıştırın; bkz. Şekil 24 sayfa 76.

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

Çizelge 4. QMA tarihinde alınan yayınlar			
Abonelik	**Konu dizgisi**	**Alınan yayınlar**	**Notlar**
SPORTS	Sports/#	Sports Sports/Rugby Sports/Rugby/Leeds	Futbol alt ağacına ilişkin tüm yayınlar Sports/Football üzerinde WILDCARD(BLOCK) tarafından engellendi
SARSENAL	Sports/#/Arsenal	-	WILDCARD(BLOCK) on Sports/Football , Arsenal üzerinde genel arama karakteri aboneliğini engeller
SLEEDS	Sports/#/Leeds	Sports/Rugby/Leeds	Sports/Rugby üzerinde varsayılan WILDCARD , Leedsüzerinde genel arama karakteri aboneliğini engellemez.
FARSENAL	Sports/Football/Arsenal	Sports/Football/Arsenal	Aboneliğin genel arama karakteri olmadığı için Arsenal bir yayın alır.
FLEEDS	Sports/Rugby/Leeds	Sports/Rugby/Leeds	Leeds , her durumda bir yayın alır.

Çizelge 5 sayfa 78 içinde, QMB 'a iki yeni abonelik eklenmesinin ve QMA' ta yayınlanmasının sonuçları gösterilir. QMB ' inin bu iki yeni abonelik olmadan hiçbir yayın almadığını hatırlayın. Beklendiği gibi, Sports/FootBall ve Sports/Rugby küme konuları olduğundan, iki yeni abonelik yayınları alır. QMB , Sports/Football/Arsenal ve Sports/Rugby/Leeds için yetkili sunucu aboneliklerini QMA'e iletti, daha sonra yayınları QMB' e gönderdi.

Beklenmeyen sonuç, daha önce hiçbir yayın almayan Sports/# ve Sports/#/Leeds aboneliklerinin artık yayınları almaktır. Bunun nedeni, diğer abonelikler için QMB 'e iletilen Sports/Football/Arsenal ve Sports/Rugby/Leeds yayınlarının artık QMB' e bağlı tüm aboneler için kullanılabilir olması olabilir. Sonuç olarak, Sports/# ve Sports/#/Leeds yerel konularına ilişkin abonelikler Sports/Rugby/Leeds yayınını alır. Sports/#/Arsenal , Sports/Football 'un WILDCARD özellik değeri BLOCKolarak ayarlandığı için yayın almaya devam eder.

Çizelge 5. QMB tarihinde alınan yayınlar			
Abonelik	**Konu dizgisi**	**Alınan yayınlar**	**Notlar**
SPORTS	*Sports/#*	*Sports/Rugby/Leeds*	*Futbol alt ağacına ilişkin tüm yayınlar Sports/Football üzerinde WILDCARD(BLOCK) tarafından engellendi*
SARSENAL	Sports/#/Arsenal	-	WILDCARD(BLOCK) on Sports/Football , Arsenal üzerinde genel arama karakteri aboneliğini engeller
SLEEDS	*Sports/#/Leeds*	*Sports/Rugby/Leeds*	*Sports/Rugby üzerinde varsayılan WILDCARD , Leedsüzerinde genel arama karakteri aboneliğini engellemez.*
FARSENAL	Sports/Football/Arsenal	Sports/Football/Arsenal	Aboneliğin genel arama karakteri olmadığı için Arsenal bir yayın alır.
FLEEDS	Sports/Rugby/Leeds	Sports/Rugby/Leeds	Leeds , her durumda bir yayın alır.

Çoğu uygulamada, bir aboneliğin başka bir aboneliğin davranışını etkilemesi istenmez. BLOCK değeriyle WILDCARD özelliğinin önemli bir kullanımı, joker karakter içeren aynı konu dizgisine aboneliklerin aynı

şekilde davranmasını sağlamaktır. Aboneliğin yayıncı ile aynı kuyruk yöneticisinde ya da farklı bir kuyruk yöneticisinde olması fark etmez, aboneliğin sonuçları aynıdır.

Joker karakterler ve akışlar

Yayınlama/abone olma API 'sine yazılan yeni bir uygulama için, `*` aboneliğinin hiçbir yayın almamasıdır. Tüm Sports yayınlarını almak için `Sports/*`, ya da `Sports/#`ve benzer şekilde `Business` yayınlarına abone olmanız gerekir.

Yayınlama/abone olma aracısı daha sonraki bir IBM MQsürümüne geçirildiğinde, kuyruğa alınmış bir yayınlama/abone olma uygulamasının davranışı değişmez. **Publish**, **Register Publisher**ya da **Subscriber** komutlarındaki **StreamName** özelliği, akışın geçirildiği konunun adıyla eşlenir.

Genel arama karakterleri ve abonelik noktaları

Yayınlama/abone olma API 'sine yazılan yeni bir uygulama için geçişin etkisi, `*` aboneliğinin hiçbir yayın almamasıdır. Tüm Sports yayınlarını almak için `Sports/*`, ya da `Sports/#`ve benzer şekilde `Business` yayınlarına abone olmanız gerekir.

Yayınlama/abone olma aracısı daha sonraki bir IBM MQsürümüne geçirildiğinde, kuyruğa alınmış bir yayınlama/abone olma uygulamasının davranışı değişmez. **Publish**, **Register Publisher**ya da **Subscriber** komutlarındaki **SubPoint** özelliği, aboneliğin geçirildiği konunun adıyla eşlenir.

Örnek: `Sport` yayınlama/abone olma kümesinin yaratılması

Aşağıdaki adımlar, dört kuyruk yöneticisine sahip bir küme (`CL1`) oluşturur: iki tam havuz, `CL1A` ve `CL1B`ve iki kısmi havuz, `QMA` ve `QMB`. Tam havuzlar yalnızca küme tanımlamalarını tutmak için kullanılır. `QMA` , küme konusu anasistemi olarak atanmıştır. Sürekli abonelikler hem `QMA` hem de `QMB`üzerinde tanımlanır.

Not: Örnek, Windowsiçin kodlanmıştır. Örneği diğer platformlarda yapılandırmak ve test etmek için Create qmgrs.bat ve create pub.bat kodlarını yeniden oluşturmanız gerekir.

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

```
qmgrs
```

Şekil 23 sayfa 76içinde konuları oluşturun. Şekil 5 'teki komut dosyası, `Sports/Football` ve `Sports/Rugby`küme konularını oluşturur.

Not: `REPLACE` seçeneği, bir konunun `TOPICSTR` özelliklerinin yerine geçmez. `TOPICSTR` , farklı konu ağaçlarını test etmek için örnekte kullanılan bir özelliktir. 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 oluşturun: topics.tst

Not: Konu dizgilerini REPLACE değiştirmediği için konuları silin.

Genel arama karakterleriyle abonelikler oluşturun. Şekil 23 sayfa 76içindeki konu nesnelerine karşılık gelen genel arama karakterleri. Her abonelik için bir kuyruk oluşturun. Komut dosyası çalıştırıldığında ya da yeniden çalıştırıldığında kuyruklar temizlenir ve abonelikler silinir.

Not: REPLACE seçeneği, bir aboneliğin TOPICOBJ ya da TOPICSTR özelliklerinin yerine geçmez. TOPICOBJ ya da TOPICSTR , farklı abonelikleri test etmek için örnekte kullanışlı olarak kullanılan özelliklerdir. 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 başvuran abonelikler oluşturun.

Not:

/sınırlayıcısı, TOPICOBJtarafından başvurulan konu dizgisi ile TOPICSTRtarafından tanımlanan konu dizgisi arasına otomatik olarak eklenir.

DEFINE SUB(FARSENAL) TOPICSTR('Sports/Football/Arsenal') DEST(QFARSENAL) tanımlaması aynı aboneliği yaratır. TOPICOBJ , önceden tanımladığınız konu dizgisine gönderme yapmanın hızlı bir yolu olarak kullanılır. Abonelik, oluşturulduğunda, artık konu nesnesine başvurmaz.

```
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 silin ve oluşturun: fullsubs.tst

İki havuzlu bir küme oluşturun. Yayınlama ve abone olmak için iki kısmi havuz oluşturun. Her şeyi silmek ve yeniden başlamak için komut dosyasını yeniden çalıştırın. Komut dosyası, konu sıradüzenini ve ilk genel arama karakteri aboneliklerini de oluşturur.

Not:

Diğer platformlarda benzer bir komut dosyası yazın ya da tüm komutları yazın. Bir komut dosyası kullanılması, her şeyi silmeyi ve aynı yapılandırmayla yeniden başlamayı hızlı hale getirir.

```
@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 oluştur: qmgrs.bat

Küme konularına abonelikler 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 güncelle: upsubs.bat

Yayın konusu dizgisini içeren iletileri yayınlamak için, parametre olarak bir kuyruk yöneticisiyle pub.batkomutunu çalıştırın. Pub.bat örnek programı kullanır **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, tümleşik yayınlama/abone olma modelinde bulunmayan bir yayın akışı kavramına sahiptir. Kuyruğa alınmış yayınlama/abone olma durumunda akışlar, farklı konular için bilgi akışını ayırmanın bir yolunu sağlar. Akış, yönetimsel olarak farklı bir konu tanıtıcısıyla eşlenebilen üst düzey bir konu olarak uygulanır.

Varsayılan akış SYSTEM.BROKER.DEFAULT.STREAM , bir ağdaki tüm aracılar ve kuyruk yöneticileri için otomatik olarak ayarlanır ve varsayılan akışı kullanmak için ek yapılandırma gerekmez. Varsayılan akışı adlandırılmamış bir varsayılan konu alanı olarak düşünün. Varsayılan akışa yayınlanan konular, kuyruğa alınmış yayınlama/abone olma özelliği etkinleştirilmiş olarak, bağlı tüm kuyruk yöneticileri tarafından anında kullanılabilir. Adlandırılmış akışlar ayrı, adlandırılmış konu alanları gibidir. Adı belirtilen akış, kullanıldığı her aracıda tanımlanmalıdır.

Yayınlayıcılar ve aboneler farklı kuyruk yöneticilerdeyse, aracılar aynı aracı sıradüzenine bağlandıktan sonra, yayınlar ve bunlar arasında akacak abonelikler için başka yapılandırma gerekmez. Aynı birlikte çalışabilirlik de ters yönde çalışıyor.

Adlandırılmış akışlar

Kuyruğa alınmış yayınlama/abone olma programlama modeliyle çalışan bir çözüm tasarımcısı, tüm spor yayınlarını Sportadlı adlandırılmış bir akışa yerleştirmeye karar verebilir. Akışın, kuyruğa alınmış yayınlama/abone olma özelliği etkinleştirilmiş olarak IBM MQ üzerinde çalışan bir kuyruk yöneticisi tarafından kullanılabilmesi için akışın el ile eklenmesi gerekir.

Değişiklik olmadan Soccer/Results akışta Sport işe abone olan kuyruğa alınmış yayınlama/abone olma uygulamaları. Sport using MQSUBbaşlıklı konuya abone olan ve Soccer/Results konu dizgisini sağlayan tümleşik yayınlama/abone olma uygulamaları da aynı yayınları alır.

Akış ekleme görevi, Akış eklemekonusunda açıklanır. İki nedenden ötürü akışları el ile eklemeniz gerekebilir.

1. Uygulamaları tümleşik yayınlama/abone olma MQI arabirimine geçirmek yerine, daha sonraki sürüm kuyruğu yöneticileriyle çalışan kuyruğa alınmış yayınlama/abone olma uygulamalarınızı geliştirmeye devam edersiniz.
2. Akışların konularla varsayılan eşlemesi, konu alanında bir "çakışmaya" neden olur ve bir akıştaki yayınlar, başka bir yerdeki yayınlarla aynı konu dizesine sahiptir.

Yetkiler

Varsayılan olarak, konu ağacının kökünde birden çok konu nesnesi vardır: SYSTEM.BASE.TOPIC, SYSTEM.BROKER.DEFAULT.STREAMve SYSTEM.BROKER.DEFAULT.SUBPOINT. Yetkiler (örneğin, yayınlama ya da abone olma), SYSTEM.BASE.TOPIC üzerindeki yetkiler tarafından belirlenir; SYSTEM.BROKER.DEFAULT.STREAM ya da SYSTEM.BROKER.DEFAULT.SUBPOINT üzerindeki tüm yetkiler yoksayılır. SYSTEM.BROKER.DEFAULT.STREAM ya da SYSTEM.BROKER.DEFAULT.SUBPOINT silinip boş olmayan bir konu dizgisiyle yeniden oluşturulursa, bu nesneler üzerinde tanımlanan yetkiler normal bir konu nesnesiyle aynı şekilde kullanılır.

Akışlar ve konular arasında eşleme

Kuyruğa alınmış bir yayınlama/abone olma akışı, bir kuyruk yaratılarak ve akımla aynı ad verilerek IBM MQ içinde taklit edilir. Bazen kuyruğa alınan yayınlama/abone olma uygulamaları için kuyruğun

adı akış kuyruğu olarak adlandırılır. Kuyruk, SYSTEM.QPUBSUB.QUEUE.NAMELISTadlı özel ad listesine eklenerek yayınlama/abone olma motoruna tanıtılır. Ad listesine ek özel kuyruklar ekleyerek, gereksinim duyduğunuz kadar akış ekleyebilirsiniz. Son olarak, akışlarla aynı adlara ve akış adıyla aynı konu dizgilerine sahip konular eklemeniz gerekir; böylece konuları yayınlayabilir ve konulara abone olabilirsiniz.

Ancak, istisnai durumlarda, konuları tanımlarken seçtiğiniz konu dizgilerini akışlara karşılık gelen konulara verebilirsiniz. Konu dizgisinin amacı, konuya konu alanında benzersiz bir ad vermektir. Genellikle akış adı bu amaca mükemmel bir şekilde hizmet eder. Bazen bir akış adı ve var olan bir konu adı çakışıyor. Sorunu çözmek için akışla ilişkili konu için başka bir konu dizgisi seçin. Benzersiz olduğundan emin olarak herhangi bir konu dizesini seçin.

Konu tanımlamasında tanımlanan konu dizgisinin başına, MQOPEN ya da MQSUB MQI çağrılarını kullanan yayıncılar ve aboneler tarafından sağlanan konu dizgisinin başına olağan şekilde önek konmuştur. Konu nesnelerini kullanan konulara başvuran uygulamalar, önek konu dizgisinin seçiminden etkilenmez; bu nedenle, yayınları konu alanında benzersiz tutan herhangi bir konu dizesini seçebilirsiniz.

Farklı akışların farklı konularla yeniden eşlenmesi, bir konu kümesini diğerinden tamamen ayırmak için konu dizgilerinin benzersiz olması için kullanılan öneklere dayanır. Eşlemenin çalışması için kesin olarak uyulacak evrensel bir konu adlandırma kuralı tanımlamanız gerekir.

IBM MQ' de, konu dizgisini konu alanındaki başka bir yerle yeniden eşlemek için ön düzeltme mekanizmasını kullanırsınız.

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

Abonelik noktaları ve konuları

Adlandırılmış abonelik noktaları, konular ve konu nesneleri tarafından öykünür.

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

IBM MQ içindeki abonelik puanları

IBM MQ , aboneliğin IBM MQ konu ağacındaki farklı konu alanlarını göstermesini sağlar. Abonelik noktası olmayan komut iletilerindeki konular, IBM MQ konu ağacının köküne değiştirilmeden eşlenir ve özellikleri SYSTEM.BASE.TOPIC' den edinilir.

Abonelik noktası olan komut iletileri, SYSTEM.QPUBSUB.SUBPOINT.NAMELISTiçindeki konu nesneleri listesi kullanılarak işlenir. Komut iletisindeki abonelik noktası adı, listedeki her konu nesnesi için konu dizgisiyle eşleştirilir. Bir eşleşme bulunursa, konu dizgisi için abonelik noktası adının başına konu düğümü olarak önek konmuştur. Konu, özelliklerini SYSTEM.QPUBSUB.SUBPOINT.NAMELISTiçinde bulunan ilişkili konu nesnesinden devralır.

Abonelik noktalarını kullanmanın etkisi, her abonelik noktası için ayrı bir konu alanı yaratmaktır. Konu alanı, abonelik noktasıyla aynı adı taşıyan bir konuda kök salmıştır. Her konu alanındaki konular, abonelik noktasıyla aynı ada sahip konu nesnesinden özelliklerini devralır.

Eşleşen konu nesnesinde ayarlanmayan özellikler, SYSTEM.BASE.TOPIC' den normal şekilde devralınır.

MQRFH2 ileti üstbilgilerini kullanarak var olan kuyruğa alınmış yayınlama/abone olma uygulamaları, Publish ya da Register subscriber komut iletilerinde **SubPoint** özelliğini ayarlayarak çalışmaya devam eder. Abonelik noktası, komut iletisindeki konu dizgisiyle birleştirilir ve sonuçtaki konu diğerleri gibi işlenir.

IBM MQ uygulamaları, abonelik puanlarından etkilenmez. Bir uygulama, bilgileri eşleşen konu nesnelerinden birinden devralan bir konu kullanıyorsa, o uygulama, eşleşen abonelik noktasını kullanarak kuyruğa alınmış bir uygulamayla birlikte çalışır.

Örnek

Var olan bir WebSphere Message Broker (şimdi IBM Integration Bus olarak bilinir) IBM MQ ' e geçirilen yayınlama/abone olma uygulaması, ilgili 'GBP' ve 'USD' konu dizgileriyle GBP ve USDolmak üzere iki konu nesnesi oluşturdu.

NYSE/IBM/SPOTbaşlıklı konuya ilişkin var olan yayınlayıcılar, USD bu konuda yayınlar yarat USD/NYSE/IBM/SPOTabonelik noktasını kullanan IBM MQüzerinde çalıştırılmak üzere geçirildi. Benzer şekilde var olan NYSE/IBM/SPOTaboneleri, USD abonelik noktasını kullanarak USD/NYSE/IBM/SPOTabonelikleri oluşturun.

MQSUB' i arayarak bir IBM MQ yayınlama/abone olma programında dolar noktası fiyatına abone olun. ' C' kod parçasında gösterildiği gibi USD konu nesnesini ve konu dizgisini 'NYSE/IBM/SPOT' kullanarak bir abonelik oluşturun.

```
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 anasisteminde USD ve GBP konu nesnelerinin CLUSTER özniteliğini ayarlayın.
2. Kümedeki diğer kuyruk yöneticilerindeki USD ve GBP konu nesnelerinin tüm kopyalarını silin.
3. USD ve GBP öğesinin kümedeki her kuyruk yöneticisinde SYSTEM.QPUBSUB.SUBPOINT.NAMELIST içinde tanımlandığından emin olun.

Tek bir kuyruk yöneticisi yayınlama/abone olma yapılandırması örneği

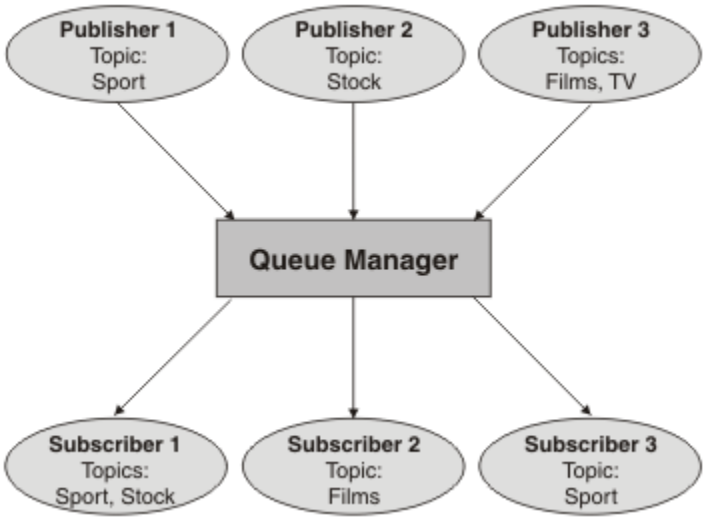
Şekil 31 sayfa 84 içinde temel bir tek kuyruk yöneticisi yayınlama/abone olma yapılandırması gösterilir. Bu örnekte, yayıncılardan birkaç konuya ilişkin bilgilerin bulunduğu bir haber hizmetine ilişkin yapılandırma gösterilmektedir:

- Yayıncı 1, Spor konusu kullanılarak spor sonuçları hakkında bilgi yayınlıyor
- Yayıncı 2 hisse senedi konusunu kullanarak hisse senedi fiyatları hakkında bilgi yayınlıyor
- Yayıncı 3, Films konusunu kullanarak film incelemeleri ve TV ' nin bir konusunu kullanarak televizyon listeleri hakkında bilgi yayınlıyor

Üç abone farklı konulara ilgi duydukları için kuyruk yöneticisi onlara ilgilendikleri bilgileri gönderir:

- Abone 1 spor sonuçlarını ve hisse senedi fiyatlarını alır
- Abone 2 film eleştirilerini alıyor
- Abone 3 spor sonuçlarını aldı

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



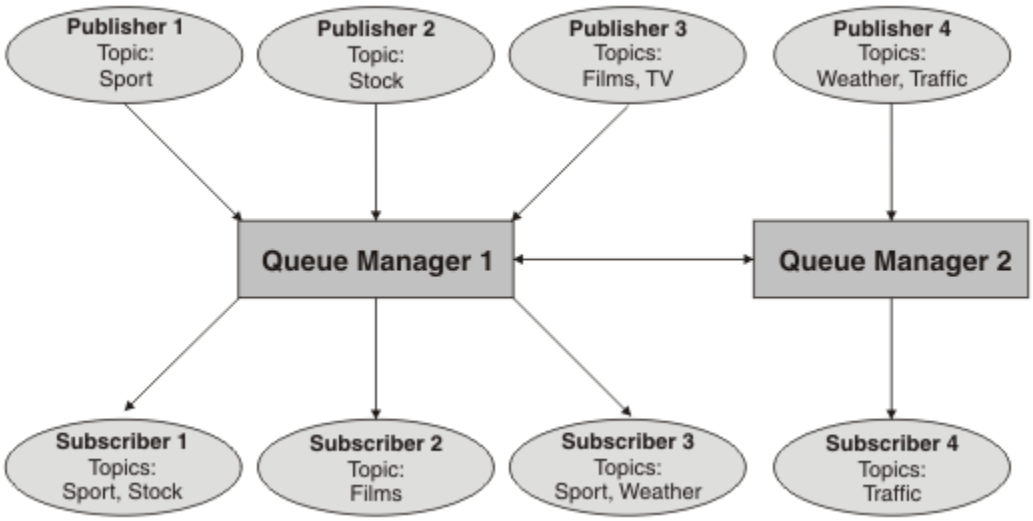
Şekil 31. Tek kuyruk yöneticisi yayınlama/abone olma örneği

Dağıtılmış yayınlama/abone olma ağları

Her kuyruk yöneticisi, bir konuda yayınlanan iletileri, o konuya abone olan yerel olarak oluşturulan aboneliklerle eşleştirir. Kuyruk yöneticilerinden oluşan bir ağı, bir kuyruk yöneticisine bağlı bir uygulama tarafından yayınlanan iletilerin, ağdaki diğer kuyruk yöneticilerinden yaratılan eşleşen aboneliklere teslim edilmesi için yapılandırabilirsiniz. Bu, kuyruk yöneticileri arasındaki basit kanallar üzerinden ek yapılandırma gerektirir.

Dağıtılmış yayınlama/abone olma yapılandırması, birbirine bağlı kuyruk yöneticileri kümesidir. Kuyruk yöneticilerinin tümü aynı fiziksel sistemde olabilir ya da birden çok fiziksel sisteme dağıtılabilir. Kuyruk yöneticilerini birbirine bağladığınızda, aboneler bir kuyruk yöneticisine abone olabilir ve başlangıçta başka bir kuyruk yöneticisine yayınlanan iletileri alabilir. Bunu göstermek için, aşağıdaki şekil "Tek bir kuyruk yöneticisi yayınlama/abone olma yapılandırması örneği" sayfa 84içinde açıklanan yapılandırmaya ikinci bir kuyruk yöneticisi ekler.

- Kuyruk yöneticisi 2, Yayıncı 4 tarafından hava durumu tahmini bilgilerini yayınlamak, Hava Durumu konusunu kullanmak ve Trafik konusunu kullanarak ana yollardaki trafik koşullarıyla ilgili bilgileri yayınlamak için kullanılır.
- Abone 4 ayrıca bu kuyruk yöneticisini kullanır ve Trafik konusunu kullanarak trafik koşullarıyla ilgili bilgilere abone olur.
- Abone 3, yayıncıdan farklı bir kuyruk yöneticisi kullanmasına rağmen hava durumu hakkında bilgilere de abone olur. Kuyruk yöneticileri birbirine bağlı olduğu için bu mümkün olabilir.

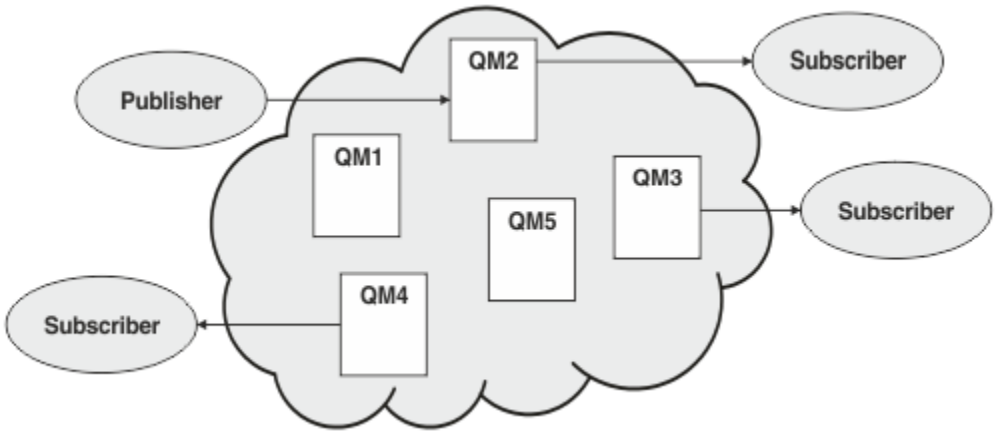


Şekil 32. İki kuyruk yöneticisiyle yayınlama/abone olma örneği

Bir üst ve alt sıradüzendeki kuyruk yöneticilerini el ile bağlayabilir ya da bir yayınlama/abone olma kümesi yaratabilir ve IBM MQ ' un sizin için bağlantı ayrıntılarının çoğunu tanımlamasına izin verebilirsiniz. Ayrıca, örneğin bir sıradüzende birkaç kümeyi birleştirerek, her iki topolojiyi de birlikte kullanabilirsiniz.

Yayınlama/abone olma kümelerine genel bakış

Yayınlama/abone olma kümesi, kümeye bir ya da daha fazla konu nesnesi eklenmiş standart bir kümedir. Bir kümedeki herhangi bir kuyruk yöneticisinde yönetimle ilgili konu nesnesi tanımladığınızda ve bir küme adı belirterek bu konu nesnesini kümelenmiş hale getirdiğinizde, bu konuya ilişkin yayınlayıcılar ve aboneler kümedeki herhangi bir kuyruk yöneticisine bağlanabilir ve yayınlanan iletiler, kuyruk yöneticileri arasındaki küme kanalları üzerinden abonelere yöneltilir.



Şekil 33. Kümeyi yayınla/abone ol

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

- doğrudan yönlendirme
- konu anasistem yöneltmesi

Doğrudan yöneltilen kümelenmiş bir konu yapılandırdığınızda, bir kuyruk yöneticisinde yayınlanan iletiler doğrudan o kuyruk yöneticisinden kümedeki diğer herhangi bir kuyruk yöneticisindeki her aboneliğe gönderilir. Bu, yayınlar için en doğrudan yolu sağlayabilir, ancak bir kümedeki tüm kuyruk yöneticilerinin diğer tüm kuyruk yöneticilerinden haberdar olmasına neden olur ve her biri arasında küme kanalları oluşturmuş olabilir.

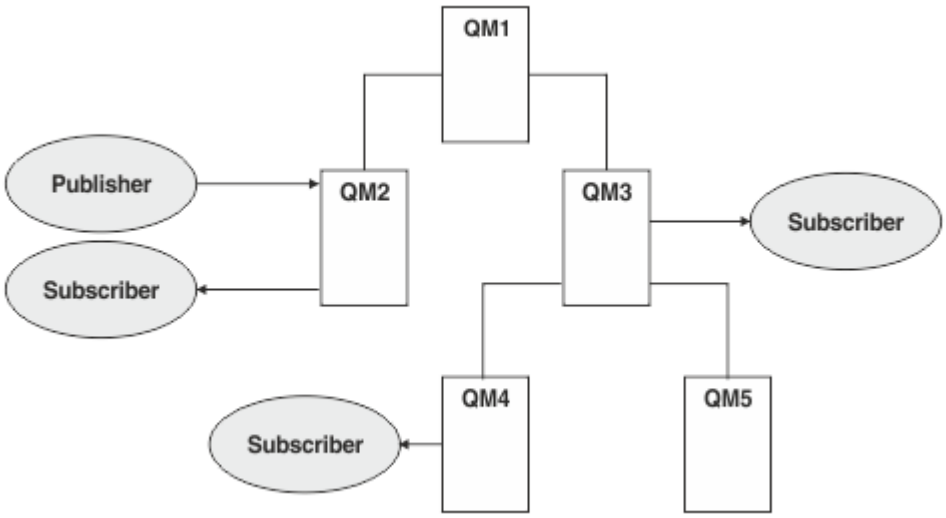
Konu anasistemi yöneltme özelliğini kullandığınızda, bir kuyruk yöneticisinde yayınlanan iletiler, denetlenen konu nesnesinin tanımlamasını barındıran bir kuyruk yöneticisine gönderilir. Bu *konu anasistem kuyruk yöneticisi* , iletiyi kümedeki diğer herhangi bir kuyruk yöneticisindeki her aboneliğe yönlendirir. Yayınlayıcılar ya da aboneler konu anasistem kuyruk yöneticilerine yerleştirilmemişse, bu durum yayınlar için daha uzun bir rotayla sonuçlanır. Ancak, yalnızca konu anasistem kuyruk yöneticilerinin kümedeki diğer tüm kuyruk yöneticilerini tanıması ve bu yöneticilerle birlikte küme kanalları kurulmuş olması yararlıdır.

Daha fazla bilgi için “Kümeleri yayınla/abone ol” sayfa 88 başlıklı konuya bakın.

Yayınlama/abone olma sıradüzenlerine genel bakış

Yayınlama/abone olma sıradüzeni, kanallar tarafından sıradüzensel bir yapıya bağlanan bir kuyruk yöneticileri kümedir. Her kuyruk yöneticisi, *üst* kuyruk yöneticisini tanımlar; açıklamalar için bkz. Kuyruk yöneticisinin yayınlama/abone olma sıradüzenine bağlanması.

Bir konuya ilişkin yayınlayıcılar ve aboneler, sıradüzendeki herhangi bir kuyruk yöneticisine bağlanabilir ve sıradüzensel kuyruk yöneticisi bağlanırlığını kullanarak bunlar arasında ileti akışı olabilir.



Şekil 34. Sıradüzeni yayınla/abone ol

Önceki şekilde, QM3 ve QM4 abonelerine gönderilen yayınlar QM2 'den QM1 ' e, ardından QM3 'e ve son olarak QM4' e yönlendirilmiştir.

Sıradüzenler, sıradüzendeki her kuyruk yöneticisi arasındaki ilişkiler üzerinde doğrudan denetim sağlar. Bu, yayıncılardan abonelere iletilerin yönlendirilmesi üzerinde ayrıntılı denetim sağlar ve özellikle kuyruk yöneticisi ağları arasında sınırlı bağlanırlık ile yönlendirme sırasında kullanışlıdır. Bir iletinin yayıncıdan abonelere yöneltildiği her kuyruk yöneticisinin kullanılabilirliğini ve yeteneğini dikkatli bir şekilde göz önünde bulundurmalısınız.

Daha fazla bilgi için "Sıradüzenlerini yayınla/abone ol" sayfa 90 başlıklı konuya bakın.

Kuyruk yöneticileri arasında yayın dağıtımı

Yöneltme seçeneklerine ek olarak, yayınları kuyruk yöneticilerinden oluşan bir ağa dağıtmaya ilişkin iki yaklaşım vardır:

- Yayınları yalnızca bir kuyruk yöneticisinden, o yayın için abonelik barındıran kuyruk yöneticilerine gönderin.
- Her yayını tüm kuyruk yöneticilerine gönderin ve abonelikleriyle eşleştirmelerine izin verin.

Önceki, yayın iletilerinin yalnızca gerektiğinde gönderilmesine neden olur, ancak kuyruk yöneticileri arasında bir abonelik bilgisi düzeyinin paylaşılmasını gerektirir. İkincisi, abonelik bilgisinin paylaşılmasını gerektirmez, ancak kuyruk yöneticileri arasında gereksiz yayın iletilerinin gönderilmesine neden olabilir.

Varsayılan olarak IBM MQ , yayınların yalnızca kendileri için abonelikleri olan kuyruk yöneticilerine gönderildiği eski yöntemi kullanır. Abonelik bilgisi, kuyruk yöneticileri arasında *yetkili sunucu abonelikleri*biçiminde yayılır. Aboneliklerin dağıtımı ve kullanım ömrüne ve dağıtılmış bir yayınlama/abone olma topolojisinde kullanmak için en verimli olan yayınların sıklığına bağlıdır. Bkz. Yayınlama/abone olma ağlarında abonelik performansı.

İlgili kavramlar

"Konu ağaçları" sayfa 72
Tanımladığınız her konu, konu ağacındaki bir öğe ya da düğümdür. Konu ağacı, MQSC ya da PCF komutları kullanılarak daha önce tanımlanmış konularla başlamak için boş olabilir ya da konuları içerebilir. Konu yaratma komutlarını kullanarak ya da bir yayında ya da abonelikte ilk kez konuyu belirterek yeni bir konu tanımlayabilirsiniz.

Sıradüzen senaryolarını yayınla/abone ol

İlgili görevler

Yayınlama/abone olma kümeleri tasarlanması

Kümeleri yayınla/abone ol

Yayınlama/abone olma kümesi, yayınların otomatik olarak yayınlama uygulamalarından kümedeki kuyruk yöneticilerinden herhangi birinde bulunan aboneliklere taşındığı, birbirine bağlı kuyruk yöneticilerinin standart bir kümesidir. Yayınları bir yayınlama/abone olma kümesinde yönlendirmek için iki seçenek vardır: *doğrudan yöneltme* ve *konu anasistem yönlendirmesi*. Seçtiğiniz yönlendirme, kümenizin boyutuna 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 kümesindeki kuyruk yöneticileri fiziksel olarak ayrı bilgisayarlarda bulunabilir ve her bir kuyruk yöneticisi çifti gerektiğinde küme kanalları tarafından otomatik olarak birbirine bağlanır. Daha fazla bilgi için bkz. Kümeler.

Yayınlama/abone olma ileti sistemi için standart bir kuyruk yöneticisi kümesi yapılandırmak üzere, kümedeki bir kuyruk yöneticisinde denetlenen bir ya da daha çok konu nesnesi tanımlarsınız. Konuyu bir küme konusu yapmak için, **CLUSTER** özelliğini kümenin adıyla yapılandırırsınız. Bunu yaptığınızda, konu ağacında bir yayıncı ya da abone tarafından kullanılan her konu, kümedeki tüm kuyruk yöneticileri tarafından paylaşılır ve konu ağacının kümelenmiş bir dalına yayınlanan iletiler, kümedeki diğer kuyruk yöneticilerine otomatik olarak yönlendirilir.

Hedef kuyruk yöneticisindeki iletiye ilişkin abone sayısından bağımsız olarak, yayınlayıcı kuyruk yöneticisi ile diğer kuyruk yöneticilerinin her biri arasında her iletinin yalnızca bir kopyası gönderilir. Bir ya da daha fazla aboneliğe sahip bir kuyruk yöneticisine vardığınızda, ileti tüm aboneliklerde yinelenir.

Kümeye katılan herhangi bir kuyruk yöneticisi otomatik olarak kümelenmiş konuları anlar ve o kuyruk yöneticisindeki yayıncılar ve aboneler kümeye otomatik olarak katılır.

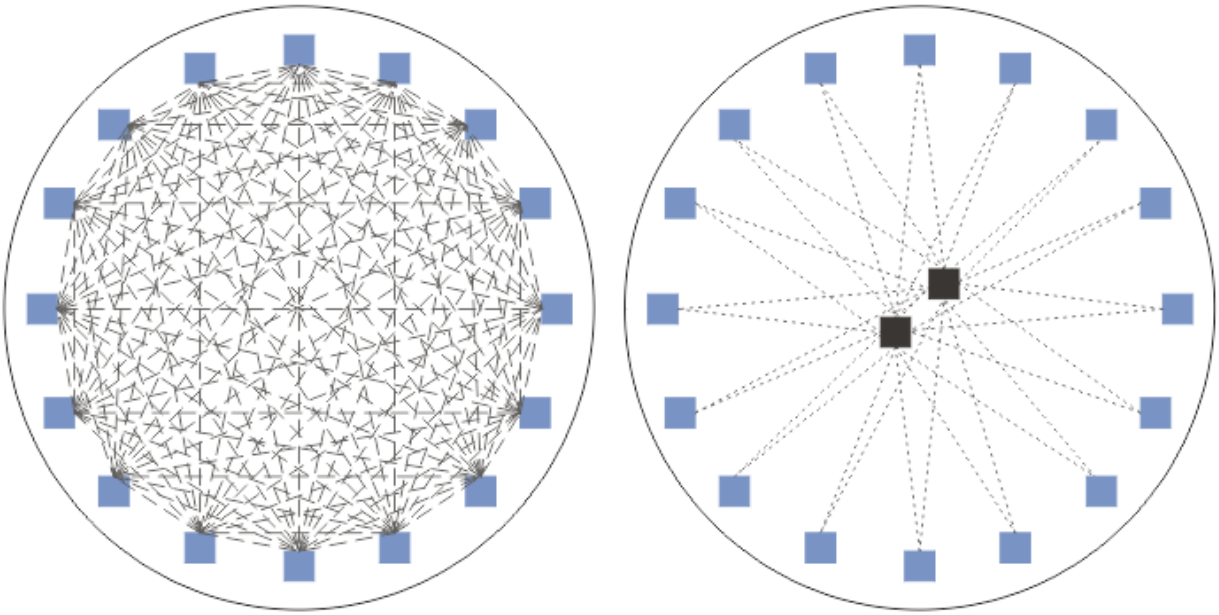
Kümelenmemiş yayınlama/abone olma etkinliği, kümelenmiş bir konu nesnesinin altına düşmeyen konu dizgileriyle çalışarak bir yayınlama/abone olma kümesinde de gerçekleşebilir.

Yayınları bir yayınlama/abone olma kümesinde yönlendirmek için iki seçenek vardır: *doğrudan yöneltme* ve *konu anasistem yönlendirmesi*. Küme içinde kullanılacak ileti yöneltmesini seçmek için, yönetilen konu nesnesindeki **CLROUTE** özelliğini aşağıdaki değerlerden birine ayarlayın:

- **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 35. Doğrudan yönlendirme ve konu anasistemi yönlendirmesi

Doğrudan yönlendirmenin genel bakışı

Denetlenen bir konu nesnesi doğrudan yöneltme için yapılandırıldığında, tüm kuyruk yöneticilerinin bunu öğrenmeleri için konu nesnesinin yalnızca kümedeki kuyruk yöneticilerinden birinde tanımlanması gerekir. Konunun tanımlandığı kuyruk yöneticisinin seçimi, konuya ilişkin yayınlama/abone olma ileti alışverişi davranışını etkilemez.

Her ileti, doğrudan yayınlayıcı kuyruk yöneticisinden kümedeki diğer kuyruk yöneticilerindeki her aboneliğe doğru akar, herhangi bir ara kuyruk yöneticisinden geçmez.

Varsayılan olarak, iletiler yalnızca bir ya da daha fazla abonelik barındıran kümedeki diğer kuyruk yöneticilerine gönderilir.

- Bu, her kuyruk yöneticisinin, şu anda bir ya da daha fazla aboneliği olan tüm konular kümesindeki diğer tüm kuyruk yöneticilerini doğrudan bilgilendirmesine dayanır. Bu, kümedeki tüm kuyruk yöneticilerinin abone olunan tüm konuları ve diğer kuyruk yöneticilerine kanal kuran bir aboneliği barındıran herhangi bir kuyruk yöneticisini bilmesine neden olur. Bu, her kuyruk yöneticisinin bir yayınlayıcısı olup olmadığına göre bağımsızdır.
- Tüm kuyruk yöneticilerindeki abone olunan her bir konunun bilgisi, abonelikleri olup olmadığına bakılmaksızın, kümedeki tüm kuyruk yöneticilerine tüm yayınların gönderileceği bir modele geçilerek kaldırılabilir. Bu, abonelik bilgi trafiğini azaltır, ancak büyük olasılıkla yayın trafiğini ve her kuyruk yöneticisinin oluşturduğu kanal sayısını artırır. Bkz. Yayınlama/abone olma ağlarında abonelik performansı.

Doğrudan yöneltilen kümelenmiş konuları kullanarak ileti akışlarını yayınlama/abone olma, her bir kümeden yayınlama/abone olma sıradüzenine bir kuyruk yöneticisi ekleyerek birden çok yayınlama/abone olma kümesini kapsayabilir. Bkz. Birden çok kümenin konu alanlarını birleştirme.

Doğrudan yönlendirmenin daha ayrıntılı bir keşfi için bkz. Yayınlama/abone olma kümelerinde doğrudan yönlendirme.

Konu anasistemi yönlendirmesine genel bakış

Denetlenen bir konu nesnesi konu anasistem yöneltmesi için yapılandırıldığında, kümedeki bir kuyruk yöneticisinden gelen yayınlar, konu nesnesinin yapılandırıldığı bir kuyruk yöneticisi aracılığıyla ("konu anasistemi") ve bundan sonra aboneliklerin bulunduğu kuyruk yöneticilerine yönlendirilir.

- Bu, şu anda bir ya da daha fazla aboneliği olan her konuya ilişkin tüm konu anasistemlerini bilgilendiren her kuyruk yöneticisine dayanır. Abonelik barındıran herhangi bir kuyruk yöneticisi, aboneliğin ilgili olduğu konu için her konu anasistemine bir kanal oluşturur.
- Konu dışı barındırma kuyruk yöneticileri, yayınlama/abone olma amacıyla kümedeki diğer konu dışı barındırma kuyruk yöneticilerinden haberdar edilmez ve bu amaçla bunlar arasında kanallar oluşturulmaz.
- Yayınlama uygulaması konuyu barındıran bir kuyruk yöneticisine bağlıysa, yayınlanan iletiler, ek bir 'hop' gerektirmeden, eşleşen aboneliklerin yaratıldığı kuyruk yöneticilerine doğrudan yöneltilir. Benzer şekilde, eşleşen abonelikler konuyu barındıran tek kuyruk yöneticisinde oluşturulursa, bu konuda yayınlanan iletiler, ek bir sekme gerektirmeden doğrudan o kuyruk yöneticisine yönlendirilir.
- Yayınlayıcıyla aynı kuyruk yöneticisine ilişkin abonelikler, ilk olarak yayınları konu nesnesinin anasistemlerine yönlendirmeden karşılanmıştır.

Kümelenmiş kuyruklara gelince, birden çok kuyruk yöneticisi aynı denetim konusu nesnesini yapılandırabilir. Bu, ileti yönlendirmesinin daha yüksek düzeyde kullanılabilirliği ve iş yükü dengeleme yoluyla yatay ölçeklendirme sağlar. Konu anasistem tarafından yönlendirilen konu nesneleri için, birden çok kuyruk yöneticisi konu ağacının aynı dalı için aynı adlandırılmış konuyu yapılandırdığında, her konu anasistemi abonelik barındıran her kuyruk yöneticisi tarafından abone olunan konuları tanır.

- Bir ileti yayınlandığında, abonelik barındırma kuyruk yöneticilerine iletilmesi için konu anasistem kuyruk yöneticilerinden birine gönderilir. Konu anasistem kuyruk yöneticisinin seçimi, kümelenmiş noktadan noktaya kuyruklarla aynı varsayılan iş yükü dengeleme kurallarını izler.
- Bir yayınlama kuyruğu yöneticisi bir ya da daha çok konu anasistem kuyruk yöneticisiyle iletişim kuramazsa, iletiler diğer kullanılabilir konu barındırma kuyruk yöneticilerine yönlendirilir.

Konu ağacının yönlendirilmiş bir dalındaki bir konuya ilişkin her yayın, kümenin herhangi bir yerinde bu konuya abonelik olmasa da, konu anasistemlerinden birine iletilir. Varsayılan olarak, iletiler buradan yalnızca bir ya da daha fazla abonelik barındıran kümedeki diğer kuyruk yöneticilerine gönderilir.

- Bu, kümedeki her kuyruk yöneticisinde abone olunan tüm konu dizgilerinden haberdar olunan her konu anasistem kuyruk yöneticisine dayanır.
- Abone olunan her bir konunun bilgisi, abonelikleri olup olmadığına bakılmaksızın, kümedeki tüm kuyruk yöneticilerine bir konu anasistemine yönlendirilen tüm yayınların gönderileceği bir modele geçilerek kaldırılabilir. Bu, abonelik bilgi trafiğini azaltır, ancak büyük olasılıkla yayın trafiğini ve her konu barındırma kuyruk yöneticisiyle oluşturulan kanalların sayısını artırır. Bkz. Yayınlama/abone olma ağlarında abonelik performansı.

Konu anasistemi tarafından yönlendirilen kümelenmiş konuları kullanarak ileti akışlarını yayınlama/abone olma sıradüzenini kullanarak birden çok yayınlama/abone olma kümesine **yayılamaz** .

Konu anasistemi yönlendirmesinin daha ayrıntılı bir keşfi için Yayınlama/abone olma kümelerinde konu anasistem yönlendirmesibaşlıklı konuya bakın.

Sıradüzenlerini yayınla/abone ol

Kanalları kullanarak kuyruk yöneticilerini birbirine bağlayarak ve ardından kuyruk yöneticisi çiftleri arasında bir alt-üst ilişkisi tanımlayarak bir yayınlama/abone olma sıradüzeni oluşturuyorsunuz. Bir ileti, bir sıradüzendeki doğrudan ilişkiler aracılığıyla bir yayınlayıcıdan aboneliklere akar. Bunun, birden çok "atlamak" anlamına gelebileceğini unutmayın.

Hedef kuyruk yöneticisindeki iletiye ilişkin abone sayısından bağımsız olarak, herhangi bir kuyruk yöneticisi çifti arasında iletinin yalnızca bir kopyası gönderilir. Bir ya da daha fazla aboneliğe sahip bir kuyruk yöneticisine vardığınızda, ileti tüm aboneliklerde yinelenir.

Varsayılan olarak, iletiler yalnızca başka bir kuyruk yöneticisindeki bir aboneliğe giden rotadaki sıradüzendeki diğer kuyruk yöneticilerine gönderilir:

- Bu, her kuyruk yöneticisinin, şu anda bu kuyruk yöneticisinde ya da diğer ilişkilerinden birinde bir ya da daha fazla aboneliği olan tüm konularla doğrudan ilişkisini bildirmesine dayanır. Bu, sıradüzendeki tüm kuyruk yöneticilerinin abone olunan tüm konuları bilmesine neden olur.

- Bu davranış, var olan aboneliklerden bağımsız olarak, sıradüzendeki tüm kuyruk yöneticilerine her zaman yayın gönderecek şekilde değiştirilebilir. Bu, abonelik bilgilerinin sıradüzeni boyunca yayılması gereksinmesini ortadan kaldırır, ancak yayın trafiğini artırabilir.

Bir küme yarattığınızda, iletilerin ağ içinde sonsuza kadar dönmesine neden olacak bir döngü oluşturmamaya dikkat etmelisiniz. 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, her bir kümeden bir yayınlama/abone olma sıradüzenine bir kuyruk yöneticisi ekleyin.

Daha ayrıntılı bir keşif için Yayınlama/abone olma sıradüzenlerinde yönlendirmebaşlıklı konuya bakın.

Bir yayınlama/abone olma ağındaki yetkili sunucu abonelikleri

Yetkili sunucu aboneliği, bir kuyruk yöneticisi tarafından başka bir kuyruk yöneticisinde yayınlanan konular için yapılan aboneliktir. Yetkili sunucu aboneliği, bir abonelik tarafından abone olunan her bir konu dizgisi için kuyruk yöneticileri arasında akar. Yetkili sunucu abonelikleri belirtik olarak oluşturulmaz, kuyruk yöneticisi sizin adınıza bunu yapar.

Kuyruk yöneticilerini bir yayınlama/abone olma kümesine ya da bir yayınlama/abone olma sıradüzenine bağlayabilirsiniz. Bağlı kuyruk yöneticileri arasında yetkili sunucu abonelikleri akışı. Yetkili sunucu abonelikleri, bir kuyruk yöneticisine bağlı bir yayıncı tarafından yaratılan bir konuya ilişkin yayınların, diğer kuyruk yöneticilerine bağlı bu konuya aboneler tarafından alınmasına neden olur. Bkz. “Dağıtılmış yayınlama/abone olma ağları” sayfa 85.

Tek tek konu dizgilerine binlerce abonelik içeren yayınlama/abone olma topolojilerinde ya da bu aboneliklerin varlığının hızla değiştiği durumlarda, yetkili sunucu aboneliği yayılımının ek yükü dikkate alınmalıdır. Bu konunun geri kalanında açıklanan otomatik toplamaya ek olarak, bağlı kuyruk yöneticileri arasındaki yetkili sunucu aboneliklerinin ve yayınların akışını daha da kısıtlayan ve bir yetkili sunucu aboneliğinin tüm bağlı kuyruk yöneticilerine yayılmasını bekleme gecikmesini azaltan el ile yapılandırma değişiklikleri yapabilirsiniz. Bkz. Yayınlama/abone olma ağlarında abonelik performansı.

Yetkili sunucu abonelikleri, yerel abonelikler tarafından kullanılan seçicileri içermez ve genel arama karakterleri içeren abonelik konusu dizgileri basitleştirilebilir. Bu, gerçek aboneliklerin eşleşmediği yerlerde yetkili sunucu abonelikleriyle eşleşen yayınlarla sonuçlanabilir ve kuyruk yöneticileri arasında ek yayın akışına neden olabilir. Abonelikleri barındıran kuyruk yöneticisi, aboneliklere ek yayınların geri dönmemesi için bu tür tutarsızlıkları süzer.

Yetkili sunucu aboneliği toplaması

Yetkili sunucu abonelikleri, yinelenen bir eleme sistemi kullanılarak toplanır. Çözümlenen belirli bir konu dizgisi için, ilk yerel abonelikte ya da alınan yetkili sunucu aboneliğinde bir yetkili sunucu aboneliği gönderilir. Aynı konu dizesine ilişkin sonraki abonelikler, var olan bu yetkili sunucu aboneliğinden yararlanır.

Yetkili sunucu aboneliği, son yerel abonelik ya da alınan yetkili sunucu aboneliği iptal edildikten sonra iptal edilir.

Yayın toplaması

Bir kuyruk yöneticisinde aynı konu dizesine birden fazla abonelik varsa, yayınlama/abone olma topolojisindeki diğer kuyruk yöneticilerinden bu konu dizgisiyle eşleşen her yayının tek bir kopyası gönderilir. İleti geldiğinde, yerel kuyruk yöneticisi iletinin bir kopyasını eşleşen her aboneliğe teslim eder.

Yetkili sunucu abonelikleri genel arama karakterleri içerdiğinde, birden çok yetkili sunucu aboneliği tek bir yayının konu dizgisiyle eşleşebilir. Bir ileti, bağlı tek bir kuyruk yöneticisi tarafından yaratılan iki ya da daha çok yetkili sunucu aboneliğine uyan bir kuyruk yöneticisinde yayınlandıysa, birden çok yetkili sunucu aboneliğini karşılamak için yayının yalnızca bir kopyası uzak kuyruk yöneticisine iletilir.

İlgili kavramlar

Dağıtılmış bir yayınlama/abone olma ağında döngü algılaması

Yetkili sunucu aboneliklerinde genel arama karakterleri

Abonelikler, yayınlarda birden çok konu dizesi ile eşleşmek için konu dizelerindeki genel arama karakterlerini kullanabilir.

Bir aboneliğin kullanabileceği iki genel arama karakteri şeması vardır: *konu tabanlı* ve *karakter tabanlı*. Bkz."Genel arama karakteri şemaları" sayfa 67.

IBM MQ içinde genel arama karakteri aboneliklerine ilişkin tüm yetkili sunucu abonelikleri, konuya dayalı genel arama karakterlerini kullanacak şekilde dönüştürülür. Karakter tabanlı bir genel arama karakteri bulunursa, yerine # karakteri konur ve yerine en yakın / karakteri konur. Örneğin, `/aaa/bbb/c*d` , `/aaa/bbb/#`olarak dönüştürülür. Dönüştürme, uzak kuyruk yöneticilerinin belirtik olarak abone olunandan biraz daha fazla yayın göndermesine neden olur. Ek yayınlar, yayınları yerel abonelerine teslim ettiğinde, yerel kuyruk yöneticisi tarafından süzülür.

WILDCARD özelliğiyle genel arama karakteri kullanımını denetleme

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

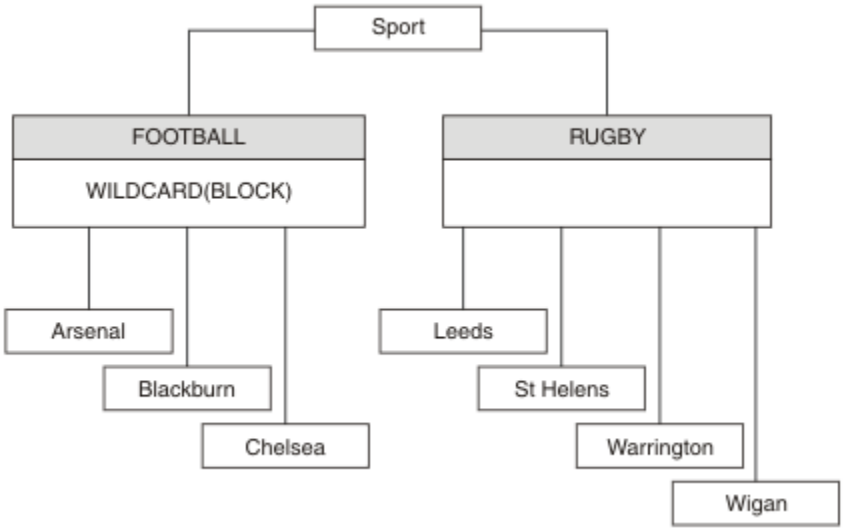
Genel arama karakteri
Bu konuya ilişkin genel arama karakteri aboneliklerinin davranışı.

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.

"Örnek: Sport yayınlama/abone olma kümesinin yaratılması" sayfa 79örneğinde, Şekil 23 sayfa 76içinde gösterilen konu ağacı yapısını oluşturmak için aşağıdaki adımları izleyebilirsiniz.



Şekil 36. WILDCARD özelliğini kullanan bir konu ağacı, BLOCK

joker konu dizgisini kullanan bir abone, Sport konusuna ve Sport/Rugby alt ağacına ilişkin tüm yayınları alır. Sport/Football konusunun WILDCARD özellik değeri BLOCKolduğundan, abone Sport/Football alt ağacına hiçbir yayın almıyor.

PASSTHRU varsayılan ayardır. WILDCARD özellik değerini PASSTHRU Sport ağacındaki düğümlere ayarlayabilirsiniz. Düğümlerde WILDCARD özellik değeri BLOCKyoksa, PASSTHRU ayarı, abonelerin Sports ağacındaki düğümlere gözlemlediği davranışı değiştirmez.

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

```
pub QMA
```

Şekil 37. QMA ' da Yayınla

Sonuçlar Çizelge 3 sayfa 76içinde gösterilir. WILDCARD özellik değerinin BLOCKayarlanmasının, joker karakter içeren aboneliklerin genel arama karakteri kapsamındaki konularla ilgili yayınları almasını nasıl önlediğine dikkat edin.

Çizelge 6. QMA tarihinde alınan yayınlar

Abonelik	Konu dizgisi	Alınan yayınlar	Notlar
SPORTS	Sports/#	Sports Sports/Rugby Sports/Rugby/Leeds	Futbol alt ağacına ilişkin tüm yayınlar Sports/Football üzerinde WILDCARD(BLOCK) tarafından engellendi
SARSENAL	Sports/#/Arsenal	-	WILDCARD(BLOCK) on Sports/Football , Arsenal üzerinde genel arama karakteri aboneliğini engeller
SLEEDS	Sports/#/Leeds	Sports/Rugby/Leeds	Sports/Rugby üzerinde varsayılan WILDCARD , Leedsüzerinde genel arama karakteri aboneliğini engellemez.

Not:

Bir aboneliğin, WILDCARD özellik değeriyle BLOCKbir konu nesnesiyle eşleşen bir genel arama karakteri olduğunu varsayın. Aboneliğin, eşleşen genel arama karakterinin sağında da bir konu dizgisi varsa, abonelik hiçbir zaman bir yayın almaz. Engellenmeyen yayınlar, engellenen joker karakterin üst karakterleri olan konulara ilişkin yayınlardır. BLOCK özellik değerine sahip konunun alt öğeleri olan konulara ilişkin yayınlar, genel arama karakteri tarafından engellenir. Bu nedenle, joker karakterin sağındaki bir konuyu içeren abonelik konusu dizgileri hiçbir zaman eşleşecek herhangi bir yayın almaz.

WILDCARD özellik değerinin BLOCK olarak ayarlanması, genel arama karakterleri içeren bir konu dizgisi kullanarak abone olamayacağınız anlamına gelmez. Böyle bir abonelik normal. Aboneliğin, WILDCARD özellik değerine BLOCKsahip bir konu nesnesiyle konuyu eşleyen belirtik bir konusu vardır. WILDCARD özellik değeri BLOCKile konunun üst ya da alt öğeleri olan konular için genel arama karakterleri kullanır. Şekil 23 sayfa 76örneğinde, Sports/Football/# gibi bir abonelik yayınları alabilir.

Genel arama karakterleri ve küme konuları

Küme konusu tanımlamaları, bir kümedeki her kuyruk yöneticisine yayılır. Kümede bir kuyruk yöneticisinde bir küme konusuna abonelik, kuyruk yöneticisinin yetkili sunucu abonelikleri yaratmasıyla sonuçlanır. Kümedeki diğer her kuyruk yöneticisinde bir yetkili sunucu aboneliği yaratılır. Genel arama karakterleri içeren konu dizelerini kullanan abonelikler, küme konularıyla birleştirildiğinde, davranışı tahmin etmek zor olabilir. Davranış aşağıdaki örnekte açıklanmıştır.

"Örnek: Sport yayınlama/abone olma kümesinin yaratılması" sayfa 79örneği için ayarlanan kümede QMB , QMAile aynı abonelik kümesine sahiptir, ancak QMB , yayıncı QMA' e yayınladıktan sonra herhangi bir yayın almadı, bkz. Şekil 24 sayfa 76. Sports/Football ve Sports/Rugby konuları küme konuları olsa da, fullsubs.tst içinde tanımlanan abonelikler bir küme konusuna başvurmaz. QMB 'den QMA' a herhangi bir yetkili sunucu aboneliği dağıtılmadı. Yetkili sunucu abonelikleri olmadan, QMA 'e ilişkin hiçbir yayın QMB' e iletilmez.

Sports/#/Leedsgibi aboneliklerin bazıları bir küme konusuna başvurmuş gibi görünebilir (bu durumda Sports/Rugby). Sports/#/Leeds aboneliği aslında SYSTEM.BASE.TOPICkonu nesnesine çözülür.

Sports/#/Leeds gibi bir abonelik tarafından başvurulan konu nesnesini çözme kuralı aşağıdaki gibidir. Konu dizgisini ilk genel arama karakterine kadar kısaltın. İlişkili bir yönetim konusu nesnesine sahip ilk konuyu aramak için konu dizesini tarayın. Konu nesnesi bir küme adı belirtebilir ya da yerel bir konu nesnesi tanımlayabilir. Sports/#/Leedsörneğinde, kesildikten sonra konu dizgisi Sportsolur; konu nesnesi yoktur ve Sports/#/Leeds , yerel konu nesnesi olan SYSTEM.BASE.TOPICnesnesinden devralır.

Kümelenmiş konulara abone olmanın joker karakter yayılımının çalışma şeklini nasıl değiştirebileceğini görmek için toplu iş komut dosyasını (upsubs.bat) çalıştırın. Komut dosyası abonelik kuyruklarını temizler ve fullsubs.tstiçinde küme konusu aboneliklerini ekler. Bir yayın kümesi oluşturmak için puba.bat dosyasını yeniden çalıştırın; bkz. Şekil 24 sayfa 76.

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

Çizelge 7. QMA tarihinde alınan yayınlar

Abonelik	**Konu dizgisi**	**Alınan yayınlar**	**Notlar**
SPORTS	Sports/#	Sports Sports/Rugby Sports/Rugby/Leeds	Futbol alt ağacına ilişkin tüm yayınlar Sports/Football üzerinde WILDCARD(BLOCK) tarafından engellendi
SARSENAL	Sports/#/Arsenal	-	WILDCARD(BLOCK) on Sports/Football , Arsenal üzerinde genel arama karakteri aboneliğini engeller
SLEEDS	Sports/#/Leeds	Sports/Rugby/Leeds	Sports/Rugby üzerinde varsayılan WILDCARD , Leedsüzerinde genel arama karakteri aboneliğini engellemez.
FARSENAL	Sports/Football/Arsenal	Sports/Football/Arsenal	Aboneliğin genel arama karakteri olmadığı için Arsenal bir yayın alır.
FLEEDS	Sports/Rugby/Leeds	Sports/Rugby/Leeds	Leeds , her durumda bir yayın alır.

Çizelge 5 sayfa 78 içinde, QMB 'a iki yeni abonelik eklenmesinin ve QMA' ta yayınlanmasının sonuçları gösterilir. QMB ' inin bu iki yeni abonelik olmadan hiçbir yayın almadığını hatırlayın. Beklendiği gibi, Sports/FootBall ve Sports/Rugby küme konuları olduğundan, iki yeni abonelik yayınları alır. QMB , Sports/Football/Arsenal ve Sports/Rugby/Leeds için yetkili sunucu aboneliklerini QMA'e iletti, daha sonra yayınları QMB' e gönderdi.

Beklenmeyen sonuç, daha önce hiçbir yayın almayan Sports/# ve Sports/#/Leeds aboneliklerinin artık yayınları almaktır. Bunun nedeni, diğer abonelikler için QMB 'e iletilen Sports/Football/Arsenal ve Sports/Rugby/Leeds yayınlarının artık QMB' e bağlı tüm aboneler için kullanılabilir olması olabilir. Sonuç olarak, Sports/# ve Sports/#/Leeds yerel konularına ilişkin abonelikler Sports/Rugby/Leeds yayınını alır. Sports/#/Arsenal , Sports/Football 'un WILDCARD özellik değeri BLOCKolarak ayarlandığı için yayın almaya devam eder.

Çizelge 8. QMB tarihinde alınan yayınlar			
Abonelik	**Konu dizgisi**	**Alınan yayınlar**	**Notlar**
SPORTS	*Sports/#*	*Sports/Rugby/Leeds*	*Futbol alt ağacına ilişkin tüm yayınlar Sports/Football üzerinde WILDCARD(BLOCK) tarafından engellendi*
SARSENAL	Sports/#/Arsenal	-	WILDCARD(BLOCK) on Sports/Football , Arsenal üzerinde genel arama karakteri aboneliğini engeller
SLEEDS	*Sports/#/Leeds*	*Sports/Rugby/Leeds*	*Sports/Rugby üzerinde varsayılan WILDCARD , Leedsüzerinde genel arama karakteri aboneliğini engellemez.*
FARSENAL	Sports/Football/Arsenal	Sports/Football/Arsenal	Aboneliğin genel arama karakteri olmadığı için Arsenal bir yayın alır.
FLEEDS	Sports/Rugby/Leeds	Sports/Rugby/Leeds	Leeds , her durumda bir yayın alır.

Çoğu uygulamada, bir aboneliğin başka bir aboneliğin davranışını etkilemesi istenmez. BLOCK değeriyle WILDCARD özelliğinin önemli bir kullanımı, joker karakter içeren aynı konu dizgisine aboneliklerin aynı şekilde davranmasını sağlamaktır. Aboneliğin yayıncı ile aynı kuyruk yöneticisinde ya da farklı bir kuyruk yöneticisinde olması fark etmez, aboneliğin sonuçları aynıdır.

Joker karakterler ve akışlar

Yayınlama/abone olma API 'sine yazılan yeni bir uygulama için, * aboneliğinin hiçbir yayın almamasıdır. Tüm Sports yayınlarını almak için Sports/*, ya da Sports/#ve benzer şekilde Business yayınlarına abone olmanız gerekir.

Yayınlama/abone olma aracısı daha sonraki bir IBM MQsürümüne geçirildiğinde, kuyruğa alınmış bir yayınlama/abone olma uygulamasının davranışı değişmez. **Publish**, **Register Publisher**ya da **Subscriber** komutlarındaki **StreamName** özelliği, akışın geçirildiği konunun adıyla eşlenir.

Genel arama karakterleri ve abonelik noktaları

Yayınlama/abone olma API 'sine yazılan yeni bir uygulama için geçişin etkisi, * aboneliğinin hiçbir yayın almamasıdır. Tüm Sports yayınlarını almak için Sports/*, ya da Sports/#ve benzer şekilde Business yayınlarına abone olmanız gerekir.

Yayınlama/abone olma aracısı daha sonraki bir IBM MQsürümüne geçirildiğinde, kuyruğa alınmış bir yayınlama/abone olma uygulamasının davranışı değişmez. **Publish**, **Register Publisher**ya da **Subscriber** komutlarındaki **SubPoint** özelliği, aboneliğin geçirildiği konunun adıyla eşlenir.

Örnek: Sport yayınlama/abone olma kümesinin yaratılması

Aşağıdaki adımlar, dört kuyruk yöneticisine sahip bir küme (CL1) oluşturur: iki tam havuz, CL1A ve CL1Bve iki kısmi havuz, QMA ve QMB. Tam havuzlar yalnızca küme tanımlamalarını tutmak için kullanılır. QMA , küme konusu anasistemi olarak atanmıştır. Sürekli abonelikler hem QMA hem de QMBüzerinde tanımlanır.

Not: Örnek, Windowsiçin kodlanmıştır. Örneği diğer platformlarda yapılandırmak ve test etmek için Create qmgrs.bat ve create pub.bat kodlarını yeniden oluşturmanız gerekir.

1. Komut dosyalarını oluşturun.
   a. topics.tst
   b. wildsubs.tstyarat

c. fullsubs.tstoluştur

d. qmgrs.bat oluştur

e. yaratma pub.bat

2. Yapılandırmayı oluşturmak için Create qmgrs.bat komutunu çalıştırın.

```
qmgrs
```

Şekil 23 sayfa 76içinde konuları oluşturun. Şekil 5 'teki komut dosyası, `Sports/Football` ve `Sports/Rugby`küme konularını oluşturur.

Not: REPLACE seçeneği, bir konunun TOPICSTR özelliklerinin yerine geçmez. TOPICSTR , farklı konu ağaçlarını test etmek için örnekte kullanılan bir özelliktir. 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 38. Konuları silin ve oluşturun: topics.tst

Not: Konu dizgilerini REPLACE değiştirmediği için konuları silin.

Genel arama karakterleriyle abonelikler oluşturun. Şekil 23 sayfa 76içindeki konu nesnelerine karşılık gelen genel arama karakterleri. Her abonelik için bir kuyruk oluşturun. Komut dosyası çalıştırıldığında ya da yeniden çalıştırıldığında kuyruklar temizlenir ve abonelikler silinir.

Not: REPLACE seçeneği, bir aboneliğin TOPICOBJ ya da TOPICSTR özelliklerinin yerine geçmez. TOPICOBJ ya da TOPICSTR , farklı abonelikleri test etmek için örnekte kullanışlı olarak kullanılan özelliklerdir. 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 39. Genel arama karakteri abonelikleri yarat: wildsubs.tst

Küme konusu nesnelerine başvuran abonelikler oluşturun.

Not:

/sınırlayıcısı, TOPICOBJtarafından başvurulan konu dizgisi ile TOPICSTRtarafından tanımlanan konu dizgisi arasına otomatik olarak eklenir.

DEFINE SUB(FARSENAL) TOPICSTR('Sports/Football/Arsenal') DEST(QFARSENAL) tanımlaması aynı aboneliği yaratır. TOPICOBJ , önceden tanımladığınız konu dizgisine gönderme yapmanın hızlı bir yolu olarak kullanılır. Abonelik, oluşturulduğunda, artık konu nesnesine başvurmaz.

```
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 40. Abonelikleri silin ve oluşturun: fullsubs.tst

İki havuzlu bir küme oluşturun. Yayınlama ve abone olmak için iki kısmi havuz oluşturun. Her şeyi silmek ve yeniden başlamak için komut dosyasını yeniden çalıştırın. Komut dosyası, konu sıradüzenini ve ilk genel arama karakteri aboneliklerini de oluşturur.

Not:

Diğer platformlarda benzer bir komut dosyası yazın ya da tüm komutları yazın. Bir komut dosyası kullanılması, her şeyi silmeyi ve aynı yapılandırmayla yeniden başlamayı hızlı hale getirir.

```
@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 41. Kuyruk yöneticileri oluştur: qmgrs.bat

Küme konularına abonelikler 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 42. Abonelikleri güncelle: upsubs.bat

Yayın konusu dizgisini içeren iletileri yayınlamak için, parametre olarak bir kuyruk yöneticisiyle pub.batkomutunu çalıştırın. Pub.bat örnek programı kullanır **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 43. Yayınla: pub.bat

İlgili kavramlar

Genel arama karakteri abonelikleri ve alıkonan yayınlar

Yayın kapsamı

Bir yayınlama/abone olma kümesini ya da sıradüzenini yapılandırdığınızda, bir yayının kapsamı, kuyruk yöneticilerinin bir yayını uzak kuyruk yöneticilerine iletip iletmeyeceğini de 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 noktalarda kümelenmiş konu nesnelerinin tanımlanmasıyla denetlenir. Yayın kapsamı, yayınların kümedeki diğer kuyruk yöneticilerine akışına izin verecek şekilde ayarlanmalıdır. Belirli kuyruk yöneticilerindeki belirli konuların ayrıntılı denetimine gereksinim duyduğunuzda, yalnızca kümelenmiş bir konuya ilişkin yayın kapsamını sınırlamanız gerekir.

Bir yayınlama/abone olma sıradüzeni kullandığınızda, yayınların kapsamı öncelikle abonelik kapsamı özniteliğiyle birlikte bu öznitelik tarafından denetlenir.

PUBSCOPE özniteliği, belirli bir konuda yapılan yayınların kapsamını belirlemek için kullanılır. Özniteliği aşağıdaki değerlerden birine ayarlayabilirsiniz:

QMGR
: Yayın yalnızca yerel abonelere teslim edilir. Bu yayınlara *yerel yayınlar*denir. Yerel yayınlar uzak kuyruk yöneticilerine iletilmez ve bu nedenle uzak kuyruk yöneticilerine bağlı aboneler tarafından alınmaz.

TÜMÜ
: Yayın, bir yayınlama/abone olma kümesinde ya da sıradüzeninde uzak kuyruk yöneticilerine bağlı yerel abonelere ve abonelere teslim edilir. Bu yayınlara *genel yayınlar*denir.

GöNDERILEN
: Konu ağacında üst konunun **PUBSCOPE** ayarını kullanın.

Yayıncılar, MQPMO_SCOPE_QMGR koyma iletisi seçeneğini kullanarak bir yayının yerel mi, yoksa genel mi olduğunu da belirtebilir. Bu seçenek kullanılırsa, **PUBSCOPE** konu özniteliği kullanılarak ayarlanan tüm davranışları geçersiz kılar.

İlgili kavramlar

"Yönetim konusu nesneleri" sayfa 73
Bir denetim konusu nesnesini kullanarak, konulara belirli, varsayılan olmayan öznitelikler atayabilirsiniz.

İlgili görevler

Dağıtılmış yayınlama/abone olma ağlarının yapılandırılması

Abonelik kapsamı

Abonelik kapsamı, bir kuyruk yöneticisindeki aboneliğin yayınlama/abone olma kümesinde ya da sıradüzeninde başka bir kuyruk yöneticisinde yayınlanan yayınları mı, yoksa yalnızca yerel yayıncılardan gelen yayınları mı alacağını denetler.

Abonelik kapsamının bir kuyruk yöneticisiyle sınırlanması, yetkili sunucu aboneliklerinin yayınlama/abone olma topolojisindeki diğer kuyruk yöneticilerine iletilmesini durdurur. Bu, kuyruklar arası yönetici yayınlama/abone olma ileti alışverişi trafiğini azaltır.

Bir yayınlama/abone olma kümesi kullandığınızda, aboneliklerin kapsamı öncelikle konu ağacındaki belirli noktalarda kümelenmiş konu nesnelerinin tanımlanmasıyla denetlenir. Abonelik kapsamı, kümedeki diğer kuyruk yöneticilerine yetkili sunucu abonelikleri akışına izin verecek şekilde ayarlanmalıdır. Belirli kuyruk yöneticilerindeki belirli konuların ayrıntılı denetimine gereksinim duyduğunuzda, yalnızca kümelenmiş bir konuya ilişkin abonelik kapsamını sınırlamanız gerekir.

Bir yayınlama/abone olma sıradüzeni kullandığınızda, aboneliklerin kapsamı öncelikle bu öznitelik tarafından yayın kapsamı özniteliğiyle birlikte denetlenir.

SUBSCOPE konu özniteliği, belirli bir konuya yapılan aboneliklerin kapsamını belirlemek için kullanılır. Özniteliği aşağıdaki değerlerden birine ayarlayabilirsiniz:

QMGR
Abonelik yalnızca yerel yayınları alır ve yetkili sunucu abonelikleri uzak kuyruk yöneticilerine yayılmaz.

TÜMÜ
Yetkili sunucu aboneliği, bir yayınlama/abone olma kümesinde ya da sıradüzeninde uzak kuyruk yöneticilerine yayılır ve abone, yerel ve uzak yayınları alır.

GöNDERILEN
Konu ağacında üst konunun **SUBSCOPE** ayarını kullanın.

Bir konuya ilişkin abonelik kapsamı ALLolarak ayarlandığında, doğrudan ya da ASPARENTaracılığıyla çözümlendiğinde, bu konuya ilişkin abonelikler, abonelik oluştururken MQSO_SCOPE_QMGR belirtilerek kapsamını QMGR ile sınırlayabilir. QMGR kapsamı olan bir konuya abonelik, kapsamı ALLolarak genişletemez.

İlgili kavramlar

"Yönetim konusu nesneleri" sayfa 73
Bir denetim konusu nesnesini kullanarak, konulara belirli, varsayılan olmayan öznitelikler atayabilirsiniz.

İlgili görevler

Dağıtılmış yayınlama/abone olma ağlarının yapılandırılması

Konu alanları

Konu alanı, abone olabileceğiniz ve yayınlayabileceğiniz konular kümeidir. Dağıtılmış yayınlama/abone olma topolojisindeki bir kuyruk yöneticisinin, o topolojideki bağlı kuyruk yöneticilerine abone olunan ve yayınlanan konuları içerebilecek bir konu alanı vardır.

Not: Bir kuyruk yöneticisi içindeki yönetim konusu nesneleri, konu dizgileri ve konu ağaçları gibi konulara genel bakış için bkz."Konular" sayfa 65. Aksi belirtilmedikçe, yürürlükteki makalede *konulara* ilişkin ek başvurular *konu dizgilerine* atıfta bulunur.

Konular başlangıçta aşağıdaki yollardan biriyle oluşturulur:

- denetimci olarak, bir konu nesnesi ya da sürekli abonelik tanımladığınızda.
- dinamik olarak, bir uygulama yeni bir konuya dinamik olarak bir yayın ya da abonelik oluşturduğunda.

Konular, hem yetkili sunucu abonelikleri aracılığıyla hem de denetim kümesi konu nesneleri yaratılarak diğer kuyruk yöneticilerine yayılır. Yetkili sunucu abonelikleri, yayıncının bağlı olduğu kuyruk yöneticisinden abonelerin kuyruk yöneticilerine iletilmesine neden olur.

Yetkili sunucu abonelikleri, bir kuyruk yöneticisi sıradüzenindeki üst-alt ilişkileri tarafından birbirine bağlanan tüm kuyruk yöneticileri arasında yayılır. Sonuç olarak, bir kuyruk yöneticisine sıradüzendeki başka bir kuyruk yöneticisinde tanımlı bir konuya abone olabilirsiniz. Kuyruk yöneticileri arasında bağlı bir yol olduğu sürece, kuyruk yöneticilerinin nasıl bağlandığı önemli değildir.

Yetkili sunucu abonelikleri, bir yayınlama/abone olma kümesindeki küme konularına abonelikler için de yayılır. Küme konusu, **CLUSTER** özniteliğine sahip bir konu nesnesine eklenen ya da özniteliği üst

öğesinden devralan bir konudur. Küme konusu olmayan konular, yerel konular olarak bilinir ve kümeye eşlenmez. Aboneliklerden yerel konulara kümeye herhangi bir yetkili sunucu aboneliği geçirilmez.

Özetlemek gerekirse, aboneler için iki durumda yetkili sunucu abonelikleri oluşturulur.

1. Kuyruk yöneticisi bir sıradüzenin üyesidir ve kuyruk yöneticisinin üst ve alt öğelerine yetkili sunucu aboneliği iletilir.
2. Kuyruk yöneticisi bir kümenin üyesidir ve abonelik konusu dizgisi, bir küme konusu nesnesiyle ilişkili bir konuya çözülür. Konu *doğrudan yöneltilen* bir küme konusu olduğunda, yetkili sunucu abonelikleri kümenin tüm üyelerine iletilir. Konu bir *konu anasistemi tarafından yönlendirilen* küme konusu olduğunda, yetkili sunucu abonelikleri yalnızca kümelenmiş konu nesnesini tanımlayan kümedeki kuyruk yöneticilerine iletilir. Daha fazla bilgi için bkz.“Kümeleri yayınla/abone ol” sayfa 88.

Bir kuyruk yöneticisi bir kümenin ve sıradüzeninin üyesiyse, yetkili sunucu abonelikleri aboneye yinelenen yayınlar teslim edilmeden her iki mekanizma tarafından da yayılır.

Üç yayınlama/abone olma topolojisinin konu alanları aşağıdaki listede açıklanmıştır:

- “Birinci dava. Kümeleri yayınla/abone ol” sayfa 100.
- “Dava 2. Sıradüzenlerini yayınla/abone ol” sayfa 101.

Ayrı konularda, aşağıdaki yapılandırma görevleri konu alanlarının nasıl birleştirileceğini açıklar.

- Yayınlama/abone olma kümesinde tek bir konu alanı yaratılması.
- Birden çok kümenin konu alanlarını birleştirme.
- Birden çok kümede konu alanlarını birleştirme ve yalıtma.
- Birden çok kümede konu alanlarını yayınlama ve bunlara abone olma.

Birinci dava. Kümeleri yayınla/abone ol

Örnekte, kuyruk yöneticisinin bir yayınlama/abone olma sıradüzenine bağlı olmadığını varsayın.

Bir kuyruk yöneticisi bir 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, konu alanı, yerel olarak tanımlanmış konu nesneleri de bulunan kümedeki başka bir kuyruk yöneticisinden farklıdır.

Bir yayınlama/abone olma kümesinde, abone olduğunuz konu bir küme konusu nesnesine çözülmedikçe, başka bir kuyruk yöneticisinde tanımlı bir konuya abone olamazsınız.

Birden çok kuyruk yöneticisindeki bir küme konusu nesnesinin aynı adlandırılmış tanımlamaları gerektiğinde (örneğin, *konu anasistem yöneltmesi*kullanılırken), tüm tanımlamaların gerektiği yerde eşleşmesi önemlidir. Daha fazla bilgi için Yayınlama/abone olma kümesinde tek bir konu alanı oluşturmabaşlıklı konuya bakın.

Tanım bir küme konusu ya da yerel konu için olsun, bir konu nesnesinin yerel tanımlaması, kümenin başka bir yerinde tanımlanan aynı konu nesnesinden önceliklidir. Başka bir yerde tanımlanan nesne daha yeni olsa da, yerel olarak tanımlanan konu kullanılır.

Bir küme konusu nesnesinin, kümenin her yerinde aynı konu dizgisiyle ilişkilendirilmesi önemlidir. Bir konu nesnesinin ilişkilendirildiği konu dizgisini değiştiremezsiniz. Aynı konu nesnesini farklı bir konu dizgisiyle ilişkilendirmek için, konu nesnesini silmeli ve yeni konu dizgisiyle yeniden oluşturmalısınız. Konu kümelenirse, etki, kümenin diğer üyelerinde saklanan konu nesnesinin kopyalarının silinmesini ve daha sonra, kümenin her yerinde yeni konu nesnesinin kopyalarının oluşturulmasını sağlar. Konu nesnesinin tüm kopyaları aynı konu dizgisine başvurmaktadır.

Farklı konu dizgileriyle, kümedeki farklı kuyruk yöneticilerindeki aynı adlandırılmış konu nesnesine ilişkin iki tanımlama yanlışlıkla yaratılabilir. Farklı konu dizgilerine sahip aynı konu nesnesinin birden çok tanımlaması, konuya nasıl ve nerede başvurulduğuna bağlı olarak farklı sonuçlar üretebilir. Bu önemli noktaya ilişkin ek bilgi için Aynı adı taşıyan birden çok küme konusu tanımlaması başlıklı konuya bakın.

Dava 2. Sıradüzenlerini yayınla/abone ol

Örnekte, kuyruk yöneticisinin bir yayınlama/abone olma kümesinin üyesi olmadığını varsayın.

IBM MQ' de bir kuyruk yöneticisi bir yayınlama/abone olma sıradüzeninin üyesiyse, konu alanı yerel olarak ve bağlı kuyruk yöneticileriyle ilgili olarak tanımlanan tüm konulardan oluşur. Bir sıradüzendeki tüm kuyruk yöneticilerinin konu alanı aynı. Konuların yerel konulara ve genel konulara bölünmesi yoktur.

PUBSCOPE ve **SUBSCOPE** seçeneklerinden birini `QMGR`olarak ayarlayın; böylece, bir yayınlayıcıdan sıradüzendeki farklı kuyruk yöneticilerine bağlı bir aboneye yayın yapmasını önleyebilirsiniz.

`Alabama` konu nesnesini, kuyruk yöneticisinde `USA/Alabama` konu dizgisiyle `QMA`tanımladığınızı varsayın. Sonuç aşağıdaki gibidir:

1. `QMA` ' daki konu alanı artık `Alabama` konu nesnesini ve `USA/Alabama`konu dizgisini içerir.
2. Bir uygulama ya da yönetici, `Alabama`konu nesnesi adını kullanarak `QMA` ' da bir abonelik oluşturabilir.
3. Bir uygulama, sıradüzendeki herhangi bir kuyruk yöneticisinde `USA/Alabama`da dahil olmak üzere herhangi bir konuya abonelik oluşturabilir. `QMA` yerel olarak tanımlanmadıysa, `USA/Alabama` konu nesnesine çözülür `SYSTEM.BASE.TOPIC`.

İlgili kavramlar

“Yayın kapsamı” sayfa 98
Bir yayınlama/abone olma kümesini ya da sıradüzenini yapılandırdığınızda, bir yayının kapsamı, kuyruk yöneticilerinin bir yayını uzak kuyruk yöneticilerine iletip iletmeyeceğini de denetler. Yayınların kapsamını yönetmek için **PUBSCOPE** konu özniteliğini kullanın.

“Abonelik kapsamı” sayfa 98
Abonelik kapsamı, bir kuyruk yöneticisindeki aboneliğin yayınlama/abone olma kümesinde ya da sıradüzeninde başka bir kuyruk yöneticisinde yayınlanan yayınları mı, yoksa yalnızca yerel yayıncılardan gelen yayınları mı alacağını denetler.

İlgili görevler

Dağıtılmış yayınlama/abone olma ağlarının yapılandırılması

IBM MQ Çok hedefli

IBM MQ Multicast, düşük gecikme süresi, yüksek yelpazeli, güvenilir çok hedefli ileti sistemi sunar.

Multicast, performansta zarar vermeden çok sayıda aboneye ölçeklenebildiği için yayınlama/abone olma mesajının verimli bir biçimidir. IBM MQ , yüksek dağılma ile düşük gecikme süreli ileti sistemi elde etmek için alındı bildirimleri, olumsuz alındı bildirimleri ve sıra numaraları kullanarak güvenilir Multicast ileti sistemi sağlar.

IBM MQ Multicast 'in adil teslimatı, aynı anda teslim edilmesine olanak sağlayarak hiçbir alıcının avantaj elde etmesini sağlamaz. IBM MQ Multicast, iletileri göndermek için ağı kullandığından, verileri dağıtmak için bir yayınlama/abone olma motoru gerekmez. Bir konu bir grup adresiyle eşlendikten sonra, yayınlayıcılar ve aboneler eşler arası kipte çalışabildiği için kuyruk yöneticisine gerek yoktur. Bu, kuyruk yöneticisi sunucularında yükün azaltılmasını sağlar ve kuyruk yöneticisi sunucusu artık olası bir hata noktası değildir.

İlk çoklu yayın kavramları

IBM MQ Multicast, İletişim Bilgileri (COMMINFO) nesnesi kullanılarak var olan sistemlerle ve uygulamalarla kolayca bütünleştirilebilir. İki TOPIC nesne alanı, çoklu yayın trafiğini desteklemek ya da yoksaymak için var olan TOPIC nesnelerinin hızlı yapılandırmasını etkinleştirir.

Çoklu yayın için gerekli nesneler

Aşağıdaki bilgiler, IBM MQ Multicast için gerekli olan iki nesneye ilişkin kısa bir genel bakışıdır:

COMMINFO nesnesi
COMMINFO nesnesi, çoklu yayın iletimi ile ilişkili öznitelikleri içerir. COMMINFO nesne parametreleriyle ilgili ek bilgi için DEFINE COMMINFObaşlıklı konuya bakın.

Ayarlanması ZORUNDA olan tek COMMINFO alanı, COMMINFO nesnesinin adıdır. Bu ad daha sonra COMMINFO nesnesini bir konuya tanıtmak için kullanılır. Değerin geçerli bir çoklu yayın grup adresi olduğundan emin olmak için COMMINFO nesnesinin **GRPADDR** alanı denetlenmelidir.

TOPIC nesnesi
Konu, yayınlama/abone olma iletisinde yayınlanan bilgilerin konusudur ve konu, bir TOPIC nesnesi yaratılarak tanımlanır. TOPIC nesne parametreleriyle ilgili ek bilgi için DEFINE TOPICbaşlıklı konuya bakın.

Var olan konular, şu TOPIC nesne parametrelerinin değerleri değiştirilerek çoklu yayın ile kullanılabilir: **COMMINFO** ve **MCAST**.

- **COMMINFO** Bu değiştirge, çok hedefli iletişim bilgileri nesnesinin adını belirtir.
- **MCAST** Bu parametre, konu ağacında bu konumda çoklu yayına izin verilip verilmeyeceğini belirtir. Varsayılan olarak **MCAST** , konunun çok hedefli özniteliğinin üst öğeden devralındığı anlamına gelen ASPARENT olarak ayarlanır. **MCAST** ' ın ETKINLEŞTIRILMIŞ olarak ayarlanması, bu düğümde çok hedefli trafiğe izin verir.

Çok hedefli ağlar ve konular

Aşağıdaki bilgiler, farklı abonelik ve konu tanımı tiplerine sahip aboneliklere ne olduğuna ilişkin bir genel bakışıdır. Bu örneklerin tümü, TOPIC nesnesinin **COMMINFO** değiştirgesinin geçerli bir COMMINFO nesnesinin adına ayarlandığını varsayar:

Konu çoklu yayın etkin olarak ayarlandı
MCAST konu dizgisi ETKINLEŞTIRILMIŞolarak ayarlanırsa, çok hedefli istemcilerden aboneliklere izin verilir ve aşağıdakiler dışında çok hedefli bir abonelik yapılır:

- Bu, çok hedefli bir istemcinin sürekli bir aboneliğidir.
- Çok hedefli bir istemciden yönetilen olmayan bir aboneliktir.
- Bu, çok hedefli olmayan bir istemcinin aboneliğidir.

Bu durumlarda çok hedefli olmayan bir abonelik yapılır ve abonelikleri normal yayınlama/abone olma olarak indirilir.

Konu çoklu yayın devre dışı olarak ayarlandı
MCAST konu dizgisi DISABLEDolarak ayarlanırsa, çok hedefli olmayan bir abonelik her zaman yapılır ve abonelikleri normal yayınlama/abone olma olarak indirilir.

Konu yalnızca çoklu yayın olarak ayarlandı
MCAST konu dizgisi değiştirgesi YALNIZCAolarak ayarlanırsa, çok noktaya yayın yeteneği olan istemcilerden aboneliklere izin verilir ve aşağıdakiler dışında çok noktaya yayın aboneliği yapılır:

- Bu sürekli bir aboneliktir: Durabilir abonelikler 2436 (0984) (RC2436): MQRC_DURABILITY_NOT_ALLOWED neden koduyla reddedilir
- Yönetilmeyen bir aboneliktir: Yönetilen olmayan abonelikler 2046 (07FE) (RC2046): MQRC_OPTIONS_ERROR neden koduyla reddedilir.
- Bu, çok hedefli olmayan bir istemcinin aboneliğidir: Bu abonelikler 2560 (0A00) (RC2560): MQRC_MULTICAST_ONLY neden koduyla reddedilir
- Bu, yerel olarak bağlı bir uygulamadan gelen bir aboneliktir: Bu abonelikler 2560 (0A00) (RC2560): MQRC_MULTICAST_ONLY neden koduyla reddedilir

Linux Windows AIX MQ Telemetry'e genel bakış

MQ Telemetry , bir kuyruk yöneticisinin parçası olan bir telemetri (MQXR) hizmetinden, kendi kendinize yazabileceğiniz ya da ücretsiz olarak yükleyebileceğiniz telemetri istemcilerinden ve komut satırı ve gezgin yönetim arabirimlerinden oluşur. Telemetri, çok çeşitli uzak aygıtlardan veri toplanması ve

yönetilmesi anlamına gelir. MQ Telemetry ile veri toplamayı ve aygıtların denetimini web uygulamalarıyla bütünleştirebilirsiniz.

MQ Telemetry , bir IBM MQbileşenidir. Bu sürümler için yükseltme esasen IBM MQ' in daha sonraki bir sürümünü kurmaktır.

Örnek uygulamalar Eclipse Paho ve MQTT.orgtarafından ücretsiz olarak kullanıma sunulmaya devam eder. Bkz. IBM MQ Telemetry Transport örnek programları.

MQ Telemetry bir IBM MQbileşeni olduğundan, MQ Telemetry ana ürünle birlikte ya da ana ürün kurulduktan sonra kurulabilir. Geçiş bilgileri için bkz. Windows üzerinde MQ Telemetry geçişi yapılıyor ve Linux üzerinde MQ Telemetry geçişi yapılıyor.

MQ Telemetry içinde aşağıdaki bileşenler bulunur:

Telemetri kanalları
MQTT istemcilerinin IBM MQ ile bağlantısını yönetmek için telemetri kanallarını kullanın. Telemetri kanalları, IBM MQile etkileşim kurmak için `SYSTEM.MQTT.TRANSMIT.QUEUE`gibi yeni IBM MQ nesnelerini kullanır.

Telemetry (MQXR) hizmeti
MQTT müşterileri, telemetri kanallarına bağlanmak için `SYSTEM.MQXR.SERVICE` telemetri hizmetini kullanır.

IBM MQ Explorer desteği MQ Telemetry
MQ Telemetry , IBM MQ Explorerkullanılarak yönetilebilir.

Belgeleme
MQ Telemetry belgeleri, standart IBM MQ ürün belgelerinde bulunur. Java ve C istemcileri için SDK belgeleri, ürün belgelerinde Javadoc ve HTML olarak sağlanır.

Telemetri kavramları

Ne yapacağınıza karar vermek için çevrenizdeki ortamdan bilgi toplayabilirsiniz. Bir tüketici olarak, hangi yiyeceği alacağınıza karar vermeden önce, mağazada ne olduğunu kontrol edin. Şimdi ayrılırsanız, bağlantı kurmadan önce bir yolculuğun ne kadar süreceğini bilmek istiyorsunuz. Doktoru ziyaret edip etmeyeceğinize karar vermeden önce belirtilerinizi kontrol edin. Beklemeye karar vermeden önce bir otobüsün ne zaman geleceğini kontrol edin. Bu kararlara ilişkin bilgiler doğrudan sayaçlardan ve aygıtlardan, kağıtta yazılı sözcükten veya bir ekrandan ve sizden gelir. Nerede olduğunuzda ve ne zaman ihtiyacınız olursa, bilgi toplayıp, bir araya getirir, analiz eder ve ona göre hareket edebilirsiniz.

Bilgi kaynakları geniş çapta dağılmış veya erişilemez ise, en doğru bilgilerin toplanması zor ve maliyetli hale gelir. Yapmak istediğiniz birçok değişiklik varsa ya da değişiklikleri yapmak zorsa, değişiklikler yapılmaz ya da daha az etkili olduklarında yapılır.

Ya cihazları dijital teknolojiyle internete bağlayarak geniş çapta dağılmış cihazlardan bilgi toplama ve kontrol etme maliyetleri büyük ölçüde düşerse? Bilgiler, internet ve işletmenin kaynakları kullanılarak analiz edilebilir. Bilinçli kararlar almak ve bunlara göre hareket etmek için daha fazla fırsatınız var.

Teknolojik eğilimler ve çevresel ve ekonomik baskılar, bu değişikliklerin gerçekleşmesine neden oluyor:

1. Sensörleri ve aktüatörleri bağlama ve kontrol etme maliyeti, standardizasyon ve düşük maliyetli dijital işlemcilere bağlantı nedeniyle azalmaktadır.
2. Internet ve internet teknolojileri, cihazları bağlamak için giderek daha fazla kullanılmaktadır. Bazı ülkelerde, cep telefonları internet uygulamalarına bağlantı sayısında kişisel bilgisayarları aşmaktadır. Diğer cihazlar da kesinlikle takip ediyor.
3. Internet ve internet teknolojileri, bir uygulamanın veri elde edilmesini çok daha kolay hale getirir. Verilere kolay erişim, verileri algılayıcılardan çok daha fazla çözümde yararlı olan bilgilere çevirmek için veri analitiğinin kullanılmasını teşvik ediyor.
4. Kaynakların akıllı kullanımı genellikle karbon emisyonlarını ve maliyetlerini azaltmanın daha hızlı ve daha ucuz bir yoludur. Alternatifler: Yeni kaynaklar bulmak veya var olan kaynakları kullanmak için yeni teknolojiler geliştirmek, uzun vadeli çözüm olabilir. Kısa vadede yeni teknolojiler geliştirmek veya yeni kaynaklar bulmak, mevcut çözümleri geliştirmekten daha riskli, daha yavaş ve daha pahalıdır.

Örnek

Bir örnek, bu eğilimlerin, ortamla akıllı bir şekilde etkileşim kurmak için nasıl yeni fırsatlar oluşturduğunu gösterir.

Uluslararası Denizlerde Yaşam Güvenliği Sözleşmesi (SOLAS), birçok gemiye Otomatik Tanımlama Sistemi (Automatic Identification System; AIS) kurulmasını gerektirir. 300 ton üzerindeki ticari gemilerde ve yolcu gemilerinde gereklidir. AIS öncelikle kıyı taşımacılığı için bir çarpışma önleme sistemidir. Deniz yetkilileri tarafından kıyı sularını izlemek ve kontrol etmek için kullanılır.

Dünyanın dört bir yanındaki meraklılar, düşük maliyetli AIS izleme istasyonları dağıtıyor ve kıyı nakliye bilgilerini internete yerleştiriyor. Diğer meraklılar, AIS ' den gelen bilgileri İnternet 'ten gelen diğer bilgilerle birleştiren uygulamalar yazmaktadır. Sonuçlar Web sitelerine konup Twitter ve SMS kullanılarak yayınlanır.

Bir uygulamada, Southampton yakınlarındaki AIS istasyonlarından gelen bilgiler gemi sahipliği ve coğrafi bilgilerle birleştirilir. Uygulama, feribot seferleri ve gidiş ile ilgili canlı bilgileri Twitter' a aktarıyor. Southampton ve Wight Adası arasındaki feribotu kullanan düzenli yolcular, Twitter veya SMS kullanarak haber akışına abone olur. Eğer yem feribotlarının geç çalıştığını gösterirse, yolcular kalkışlarını geciktirebilir ve planlanan varış saatinden daha sonra yanaşınca feribota yetişebilirler.

Daha fazla örnek için bkz. “Telemetri kullanım senaryoları” sayfa 105.

İlgili görevler

kurmaMQ Telemetry
YönetmeMQ Telemetry
Windows üzerinde MQ Telemetry geçişi yapılıyor
Linux üzerinde MQ Telemetry geçişi yapılıyor
MQ Telemetry için uygulama geliştirilmesi
MQ Telemetry sorunlarının giderilmesi

İlgili başvurular

MQ Telemetry Başvuru

Linux Windows AIX

MQ Telemetry ' a Giriş

İnsanlar, işletmeler ve hükümetler, yaşadığımız ve çalıştığımız ortamla daha akıllıca etkileşim kurmak için MQ Telemetry ' i giderek daha fazla kullanmak istiyor. MQ Telemetry , her türlü aygıtı İnternet 'e ve kuruluşa bağlar ve akıllı aygıtlar için uygulama oluşturma maliyetlerini düşürür.

MQ Telemetry nedir?

- Bu, IBM MQ tarafından sağlanan evrensel ileti sistemi omurgasını çok çeşitli uzak algılayıcılara, aktüatörlere ve telemetri aygıtlarına genişleten bir IBM MQ özelliğidir. Akıllı kurumsal uygulamalar, hizmetler ve karar alıcılar ile donanımlı aygıt ağları arasında bağlantı kurabilmesi için MQ Telemetry genişletir IBM MQ .
- MQ Telemetry ' in çekirdek parçaları şunlardır:

  MQ Telemetry (MQXR) hizmeti.
  Bu hizmet, IBM MQ sunucusu içinde çalışır ve telemetri aygıtlarıyla iletişim kurmak için IBM MQ Telemetry Transport (MQTT) iletişim kuralını kullanır.

  Yazdığınız MQTT uygulamaları.
  Bu uygulamalar, telemetri aygıtları ile IBM MQ kuyruk yöneticisi arasında taşınan bilgileri ve bu bilgilere yanıt olarak gerçekleştirilen işlemleri denetler. Bu uygulamaların oluşturulmasına yardımcı olmak için MQTT istemci kitaplıklarını kullanırsınız.

Benim için ne yapabilir?

- MQTT , çok çeşitli aygıtlar için MQTT uygulamalarının yaratılmasına olanak sağlayan bir açık ileti sistemi iletimi.

- MQTT istemcileri, sınırlı kaynakları olan küçük aygıtlarda çalışabilirler.
- MQTT , bant genişliğinin düşük olduğu, veri gönderme maliyetinin pahalı olduğu ya da kırılgan olabileceği ağlarda verimli bir şekilde çalışır.
- İleti teslimatı kesindir ve uygulamadan ayrışmıştır.
- 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 uygulamasıyla (`mqttv3app.jar`) çalışır. Bkz. IBM MQ Telemetry Transport örnek programları.
- IBM MQtelemetri özelliğini yönetmek için IBM MQ Explorer ve ilişkili araçlarını kullanın.
- Bir kuyruk yöneticisine bağlanan ve yayınlama/abone olma ileti sistemini kullanan MQTT uygulamaları yaratmanıza yardımcı olması için istemci kitaplıklarını kullanın.
- Uygulamanızı ve istemci kitaplığınızı, uygulamanızın çalıştırılacağı aygıta dağıtın.

Nasıl çalışır?

- MQTT , bir yayınlama/abone olma iletişim kuralıdır. MQTT istemci uygulaması, bir MQTT sunucusunda ileti yayınlayabilir ya da bir MQTT sunucusuna bağlanan uygulamalar tarafından gönderilen iletilere abone olabilir.
- MQTT istemci uygulamaları, MQTT ileti aktarımını gerçekleştiren istemci kitaplıklarını kullanır.
- Temel bir MQTT istemci uygulaması standart bir MQ istemcisi gibi çalışır, ancak çok daha geniş bir platform ve ağ yelpazesinde çalışabilir.
- MQ Telemetry (MQXR) hizmeti, bir IBM MQ kuyruk yöneticisini MQTT sunucusuna dönüştürür.
- Bir IBM MQ kuyruk yöneticisi MQTT sunucusu gibi davrandığında, kuyruk yöneticisine bağlanan diğer uygulamalar MQTT istemcisi için abone olabilir ve iletileri alabilir.
- Kuyruk yöneticisi, yayınlama uygulamalarından abone uygulamalara ileti dağıtan yönlendirici görevi görür.
- İletiler, farklı istemci uygulamaları tipleri arasında dağıtılabilir. Örneğin, Telemetry istemcileri ile JMS istemcileri arasında.

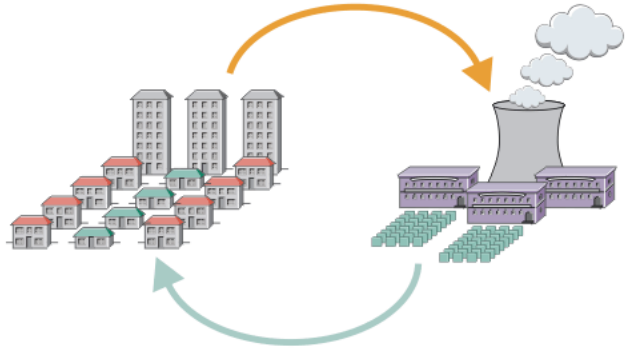
Not: MQ Telemetry , WebSphere Message Broker sürüm 7 'de pazarlamadan kaldırılan SCADA düğümlerinin yerine geçer (şimdi IBM Integration Bus olarak bilinir) ve Windows, Linuxve AIXüzerinde çalışır.

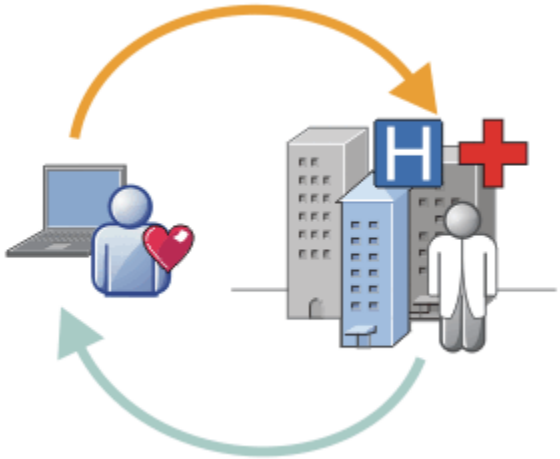
Linux Windows AIX Telemetri kullanım senaryoları

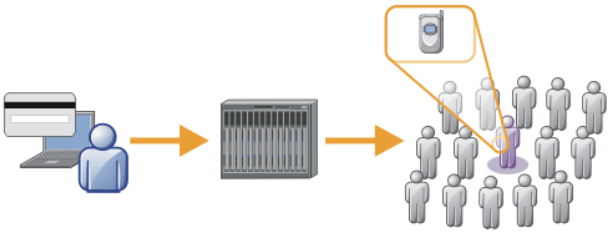
Telemetri, otomatik algılama, veri ölçümü ve uzak aygıtların denetimidir. Verilerin cihazlardan merkezi bir kontrol noktasına iletilmesi üzerinde durulmaktadır. Telemetri, aygıtlara yapılandırma ve denetim bilgilerinin gönderilmesini de içerir.

MQ Telemetry , MQTT protocolkullanarak küçük aygıtları bağlar ve IBM MQkullanarak aygıtları diğer uygulamalara bağlar. MQ Telemetry , aygıtlar ve internet arasındaki boşluğu kapatarak "akıllı çözümler" oluşturulmasını kolaylaştırır. Akıllı çözümler, aygıtları izleyen ve denetleyen uygulamalar için internet ve kurumsal uygulamalarda bulunan bilgi zenginliği açığa çıkartır.

Aşağıdaki çizgeler, MQ Telemetry' in bazı tipik kullanımlarını göstermektedir:

Telemetri: Akıllı Elektrik	
	• Hizmet sağlayıcıya gönderilen enerji kullanım verilerini içeren MQTT iletisi. • MQ Telemetry , enerji kullanımı verilerinin analizine dayalı olarak CONTROL COMMANDS (Denetim Komutları) gönderir. • Daha fazla bilgi için aşağıdaki kullanım senaryosına bakın: “Telemetri kullanım senaryosu: Ev enerjisi izleme ve denetimi” sayfa 108

Telemetri: Akıllı Sağlık Hizmetleri	
• MQ Telemetry , Sağlık Verilerini Hastanenize ve Doktorunuza gönderir. • MQTT ileti uyarıları ya da geribildirim, Sağlık Verilerinin analizine dayalı olarak gönderilebilir. • Daha fazla bilgi için aşağıdaki kullanım senaryosına bakın: “Telemetri kullanım senaryosu: Evde hasta izleme” sayfa 106	

Telemetri: Bir kalabalıkta bir	
	• Banka sunucusuna basit bir kart işlemi gönderilir. • MQ Telemetry , binlerden bir kişiyi tanımlar ve müşteriye kartının kullanıldığını bildirerek. • MQ Telemetry , en basit bilgi girişini kullanabilir ve o kişiyi bulabilir.

Alt konularda açıklanan kullanım senaryoları gerçek örneklerden çizilir. Telemetri kullanmanın bazı yollarını ve telemetri teknolojisinin çözmesi gereken bazı yaygın sorunları gösterirler.

Linux Windows AIX Telemetri kullanım senaryosu: Evde hasta izleme

IBM ile bir sağlık hizmeti sağlayıcısı arasında bir kalp hasta bakım sistemi üzerinde işbirliği içinde, implante edilmiş bir kalp yetmezliği defibrilatörü bir hastaneyle iletişim kurar. Hastaya ve implante edilen aygıta ilişkin veriler, RF telemetrisi kullanılarak hastanın evindeki MQTT aygıtına aktarılır.

Tipik olarak, aktarım her gece yatağın yanında bulunan bir vericiye gerçekleşir. Verici, verileri telefon sistemi üzerinden güvenli bir şekilde, verilerin analiz edildiği hastaneye aktarır.

Sistem, bir hastanın bir hekime yapması gereken ziyaret sayısını azaltır. Hastanın ya da cihazın ne zaman dikkat edilmesi gerektiğini belirler ve acil bir durumda nöbetçi hekimi uyarır.

IBM ile sağlık hizmeti sağlayıcısı arasındaki işbirliği, bir dizi telemetri kullanım senaryosu için ortak özelliklere sahiptir:

Görünmezlik

Aygıt, günün bir kısmı için güç, telefon hattı ve aygıta yakın olmak dışında kullanıcı müdahalesi gerektirmez. Çalışması güvenilir ve kullanımı basittir.

Hastanın aygıtı ayarlama gereksinimini ortadan kaldırmak için aygıt sağlayıcısı aygıtı önceden yapılandırır. Hastanın sadece fişini takması gerekiyor. Hasta tarafından yapılandırmanın ortadan kaldırılması, aygıtın çalışmasını basitleştirir ve aygıtın yanlış yapılandırılma olasılığını azaltır.

MQTT istemcisi, aygıtın bir parçası olarak yerleştirilmiştir. Aygıt geliştirici, MQTT istemci uygulamasını aygıta ve geliştiriciye ya da sağlayıcıya yerleştirerek MQTT istemcisini ön yapılandırmanın bir parçası olarak yapılandırır.

MQTT istemcisi, geliştiricinin Java uygulamasına dahil ettiği bir Java SE JAR dosyası olarak verilir. Bunun gibiJava dışı ortamlar için aygıt geliştiricisi, yayınlanan MQTT biçimlerini ve iletişim kuralını kullanarak farklı bir dilde bir istemci uygulayabilir. Alternatif olarak, geliştirici Windows, Linux ve ARM platformları için paylaşılan kitaplık olarak gönderilen C istemcilerinden birini kullanabilir.

Eşit olmayan bağlanırlık

Defibrilatör ile hastane arasındaki iletişimin ağ özellikleri eşit değildir. Hastadan veri toplama ve verileri hastaneye gönderme gibi farklı sorunları çözmek için iki farklı ağ kullanılır. Patent ve MQTT aygıtı arasında kısa menzilli düşük güçlü RF ağı kullanılır. Verici, düşük bant genişlikli bir telefon hattı üzerinden VPN TCP/IP bağlantısı kullanarak hastaneye bağlanır.

Her aygıtı doğrudan bir Internet Protocol ağına bağlamanın bir yolunu bulmak genellikle pratik değildir. Bir göbeğe bağlı iki ağ kullanmak yaygın bir çözümdür. MQTT aygıtı, hastadan gelen bilgileri depolayan ve hastaneye ileten basit bir merkezdir.

Durumu

Hekim hasta verilerinin gerçekliğine güvenebilmeli ve hasta verilerinin gizliliğine saygı duyulmasını istemelidir.

Bazı durumlarda VPN ya da TLS kullanarak bağlantıyı şifrelemek yeterlidir. Diğer durumlarda, depolandıktan sonra bile verilerin güvenliğini sağlamak istenir.

Bazen telemetri cihazı güvenli değildir. Ortak bir konutta olabilir, örneğin. Verilerin doğru hastadan alındığından emin olmak için aygıtın kullanıcısının kimliği doğrulanmalıdır. Aygıtın kendisinin kimliği TLS kullanılarak sunucuda doğrulanabilir ve sunucunun kimliği aygıtta doğrulanabilir.

Aygıt ile kuyruk yöneticisi arasındaki telemetri kanalı, kullanıcı kimlik doğrulaması için JAAS 'ı, iletişim şifrelemesi için TLS' yi ve aygıt kimlik doğrulamasını destekler. Bir yayına erişim, IBM MQiçindeki nesne yetkisi yöneticisi tarafından denetlenir.

Kullanıcının kimliğini doğrulamak için kullanılan tanıtıcı, ortak bir hasta kimliği gibi farklı bir tanıtıcıyla eşlenebilir. Ortak bir tanıtıcı, IBM MQiçinde konuları yayınlamak için yetkilendirmenin yapılandırılmasını kolaylaştırır.

Bağlanırlık

MQTT aygıtı ile hastane arasındaki bağlantı çevirmeli bağlantı kullanır ve 300 baud kadar düşük bant genişliğiyle çalışır.

300 baud hızında etkili bir şekilde çalışmak için MQTT protocol , TCP/IP üstbilgilerine ek olarak bir iletiye yalnızca birkaç fazladan bayt ekler.

MQTT protocol , gecikme sürelerini düşük tutan tek iletim *yangın ve unut* ileti sistemi sağlar. Garantili teslim, yanıt süresinden daha önemliyse, *en az bir kez* ve *tam olarak bir kez* teslimatı garanti etmek için birden çok iletim de kullanabilir. Teslimatı garanti etmek için, iletiler başarıyla teslim edilinceye kadar aygıtta depolanır. Bir aygıt kablosuz olarak bağlıysa, garantili teslimat özellikle kullanışlıdır.

Ölçeklenebilirlik

Telemetri cihazları tipik olarak on binlerden milyonlara kadar çok sayıda devreye alınır.

Birçok aygıtın bir sisteme bağlanması, bir çözüme büyük talepler getirmektedir. Aygıtların ve yazılımlarının maliyeti ve lisansların, aygıtların ve kullanıcıların yönetim talepleri gibi iş talepleri vardır. Teknik talepler, ağdaki ve sunuculardaki yükü içerir.

Bağlantıların açılması, açık bağlantıların korunmasından daha fazla sunucu kaynağı kullanır. Ancak bunun gibi telefon hatlarını kullanan bir kullanım durumunda, bağlantıların gideri, bağlantıların gerekenden daha uzun süre açık kalmadığı anlamına gelir. Veri aktarımları büyük ölçüde toplu niteliktedir. Bağlantılar, uyku zamanında ani bir bağlantı zirvesinden kaçınmak için gece boyunca zamanlanabilir.

İstemcide, istemcilerin ölçeklenebilirliği, gereken en düşük istemci yapılandırmasıyla yardımcı olur. MQTT istemcisi aygıtta yerleşik olarak yer almaktadır. Aygıtların hastalara dağıtılması için bir yapılandırma ya da MQTT istemci lisansı kabul adımının oluşturulmasına gerek yoktur.

Sunucuda, MQ Telemetry ' in kuyruk yöneticisi başına 50.000 açık bağlantı hedefi vardır.

Bağlantılar IBM MQ Explorerkullanılarak yönetilir. IBM MQ Explorer , görüntülenecek bağlantıları yönetilebilir bir sayıya süzer. Müşterilere uygun olarak seçilen bir tanıtıcı ayırma şemasıyla, bağlantıları coğrafi bölgeye göre ya da hasta adına göre alfabetik olarak süzebilirsiniz.

Linux Windows AIX Telemetri kullanım senaryosu: Ev enerjisi izleme ve denetimi

Akıllı sayacaklar, enerji tüketimi hakkında geleneksel ölçümlerden daha fazla ayrıntı toplar.

Akıllı sayaçlarla genellikle bir evdeki tek tek cihazları izlemek ve kontrol etmek için yerel bir telemetri ağı eşlenir. Bazıları da uzaktan uzaktan izleme ve kontrol için bağlanır.

Uzak bağlantı bir kişi tarafından, bir güç yardımcı programı tarafından ya da merkezi bir denetim noktası tarafından kurulabilir. Uzaktan kumanda noktası, güç kullanımını okuyabilir ve kullanım verileri sağlayabilir. Sürekli fiyatlandırma ve hava durumu bilgileri gibi kullanımı etkilemek için veri sağlayabilir. Genel güç üretim verimliliğini artırmak için yükü sınırlayabilir.

Akıllı sayaç geniş çapta devreye alınmaya başlıyor. Örneğin Birleşik Krallık hükümeti, 2020 yılına kadar tüm İngiltere evlerine akıllı sayaçların konuşlandırılması konusunda danışmanlık yapıyor.

Ev ölçümü kullanım senaryolarının bir dizi ortak özelliği vardır:

Görünmezlik

Kullanıcı ölçümü kullanarak enerji tasarrufu yapmak istemedikçe, ölçüm kullanıcı müdahalesi gerektirmemelidir. Bu, bireysel cihazlara sağlanan enerji arzının güvenilirliğini azaltmamalıdır.

MQTT istemcisi, ölçümle birlikte devreye alınan yazılıma yerleştirilebilir ve ayrı bir kuruluş ya da yapılandırma gerektirmez.

Eşit olmayan bağlanırlık

Aygıtlar ve akıllı sayaç arasındaki iletişim, sayaç ile uzak bağlantı noktası arasında olduğundan farklı bağlantı standartları gerektirir.

Akıllı sayaçtan aygıtlara olan bağlantının yüksek düzeyde kullanılabilir olması ve bir ev ağı için ağ standartlarına uygun olması gerekir.

Uzak ağ çeşitli fiziksel bağlantılar kullanabilir. Bazıları, örneğin hücresel, yüksek bir iletim maliyetine sahip ve aralıklı olabilir. MQTT v3 belirtimi, uzak bağlantıları ve yerel bağdaştırıcılar ile akıllı sayaç arasındaki bağlantıları hedeflemektedir.

Güç çıkışları ve uygulamalar ile sayaç arasındaki bağlantı, Zigbee gibi bir ana alan ağı kullanır. MQTT algılayıcı ağları için (MQTT-S), Zigbee ve diğer düşük bant genişliği ağ iletişim kurallarıyla çalışacak şekilde tasarlanmıştır. MQ Telemetry , MQTT-S ' yi doğrudan desteklemez. MQTT-S 'yi MQTT v3' e bağlamak için bir ağ geçidi gerekir.

Evde hasta izleme gibi, ev enerjisini izleme ve kontrol için çözümler, akıllı sayacı bir göbek olarak kullanarak birbirine bağlı birden çok ağ gerektirir.

Durumu

Akıllı sayaçlarla ilişkili bir dizi güvenlik sorunu vardır. Bu konular, işlemlerin inkar edilmemesi, başlatılan denetim eylemlerinin yetkilendirilmesi ve güç tüketimi verilerinin gizliliğini içerir.

Gizliliği sağlamak için, MQTT tarafından ölçüm ve uzaktan denetim noktası arasında aktarılan veriler TLS kullanılarak şifrelenebilir. Denetim eylemlerinin yetkilendirildiğinden emin olmak için ölçüm ile uzaktan denetim noktası arasındaki MQTT bağlantısı karşılıklı olarak TLS kullanılarak doğrulanabilir.

Bağlanırlık

Uzak ağın fiziksel yapısı önemli ölçüde değişebilir. Var olan bir geniş bant bağlantısını kullanabilir ya da yüksek çağrı maliyetlerine ve kesintili kullanılabilirliğe sahip bir mobil ağ kullanabilir. Yüksek maliyetli, aralıklı bağlantılar için MQTT verimli ve güvenilir bir iletişim kuralıdır; bkz. “Telemetri kullanım senaryosu: Evde hasta izleme” sayfa 106.

Ölçeklenebilirlik

Sonunda elektrik şirketleri veya merkezi kontrol noktaları, on milyonlarca akıllı sayaçların devreye alınmasını planlıyorlar. Başlangıçta, devreye alma başına sayaçların sayısı on ile yüz binlerce arasında olur. Bu sayı, kuyruk yöneticisi başına 50.000 açık istemci bağlantısının ilk MQTT hedefiyle karşılaştırılabilir.

Ev enerjisini izleme ve kontrol için mimarinin kritik bir yönü, akıllı sayacı bir ağ yoğunlaştırıcı olarak kullanmaktır. Her aygıt bağdaştırıcısı ayrı bir algılayıcıdır. Göbek, MQTTkullanarak bunları yerel bir göbeğe bağlayarak, merkezi denetim noktası ile veri akışlarını tek bir TCP/IP oturumuna yoğunlaştırabilir ve oturum kesintilerinin üstesinden gelmek için kısa bir süre boyunca iletileri saklayabilir.

Uzak bağlantılar, iki nedenden ötürü ev enerji kullanım senaryolarında açık bırakılmalıdır. İlk olarak, bağlantıların açılması isteklerin gönderilmesine göre uzun sürdüğü için. Kısa bir aralıkta "yükleme sınırlaması" istekleri göndermek için birçok bağlantı açma süresi çok uzun. İkinci olarak, elektrik şirketinden gelen yük sınırlaması isteklerini almak için bağlantının öncelikle istemci tarafından açılması gerekir. MQTTile, bağlantılar her zaman istemci tarafından başlatılır ve güç şirketinden yük sınırlaması istekleri almak için bağlantının açık bırakılması gerekir.

Bağlantı açma hızı kritik önem taşıyorsa ya da sunucu zaman açısından kritik istekleri başlatıyorsa, çözüm genellikle birçok açık bağlantıyı korumaktır.

Linux Windows AIX Telemetri kullanım senaryoları: Radyo Frekansı Tanıtıcısı (RFID)

RFID, bir nesneyi kablosuz olarak tanımlamak ve izlemek için yerleşik bir RFID etiketinin kullanılmasıdır. RFID etiketleri, birkaç metre aralığına kadar okunabilir ve RFID okuyucunun görüş alanı dışında olabilir. Pasif etiketler bir RFID okuyucusu tarafından etkinleştirilir. Etkin etiketler, dış etkinleştirme olmadan iletir. Etkin etiketlerin bir güç kaynağı olmalıdır. Pasif etiketler, aralıklarını artırmak için bir güç kaynağı içerebilir.

RFID birçok uygulamada kullanılır ve kullanım senaryolarının türleri büyük ölçüde değişir. RFID kullanım senaryoları ve evde hasta izleme ve ev enerjisi izleme ve kontrol kullanım senaryoları, bazı benzerlikler ve farklılıklara sahiptir.

Görünmezlik

Birçok kullanım durumunda, RFID okuyucusu çok sayıda devreye alınır ve kullanıcı müdahalesi olmadan çalışmalıdır. Okuyucu, merkezi bir denetim noktasıyla iletişim kurmak için yerleşik bir MQTT istemcisi içerir.

Örneğin, bir dağıtım ambarında, okuyucu bir paleti algılamak için bir hareket algılayıcısı kullanır. Paletteki öğelerin RFID etiketlerini etkinleştirir ve merkezi uygulamalara veri ve istek gönderir. Veriler, stok konumunu güncellemek için kullanılır. İstekler, paleti belirli bir bölmeye taşıma gibi bir sonraki palete ne olacağını denetler. Havayolları ve havaalanı bagaj sistemleri RFID ' yi bu şekilde kullanıyor.

Bazı RFID kullanım senaryolarında okuyucunun Java Platform, Micro Edition (Java ME) gibi standart bir bilgi işlem ortamı vardır. Bu durumlarda, MQTT istemcisi üretimin ardından ayrı bir yapılandırma adımında devreye alınabilir.

Eşit olmayan bağlanırlık
RFID okuyucular, MQTT istemcisi içeren yerel denetim aygıtından ayrılmış olabilir ya da her okuyucu bir MQTT istemcisi katıyor olabilir. Genellikle, coğrafi ya da iletişim faktörleri topoloji seçimini gösterir.

Durumu

Gizlilik ve özgünlük, RFID etiketlerinin eklenmesine ilişkin güvenlik endişeleridir. RFID etiketleri dikkat çekmez ve gizlice izlenebilir, aldatılabilir ya da kurcalanabilir.

RFID güvenlik sorunlarının çözümü, yeni RFID çözümlerini devreye alma fırsatını artırır. Güvenlik açığı RFID etiketinde ve yerel okuyucuda olsa da, merkezi bilgi işlemenin kullanılması farklı tehditlere karşı önlem almak için yaklaşımlar önerir. Örneğin, etiket kurcalama, hisse senedi düzeylerinin teslimatlar ve dağıtılarla dinamik olarak ilintilendirilmesiyle saptanabilir.

Bağlanırlık

RFID uygulamaları genellikle RFID okuyucularından ve anında sorgulardan toplanan bilgilerin toplu olarak depolanmasını ve iletilmesini içerir. Dağıtım ambarı kullanım senaryosunda RFID okuyucu her zaman bağlanır. Bir etiket okunduğunda, okuyucuyla ilgili bilgilerle birlikte yayınlanır. Ambar uygulaması, yanıtı okuyucuya geri yayınlar.

Ambarlama uygulamasında ağ genellikle güvenilirdir ve anlık istekler düşük gecikme başarımı için *yanma ve unutma* iletilerini kullanabilir. Toplu saklama ve iletme verileri, veri kaybetmeyle ilişkili yönetim maliyetlerini en aza indirmek için *tam olarak bir kez* ileti sistemi kullanabilir.

Ölçeklenebilirlik
RFID uygulaması anında yanıtlar gerektiriyorsa, bir ya da iki saniye içinde RFID okuyucuları bağlı kalmalıdır.

Linux Windows AIX Telemetri kullanım senaryoları: Ortam algılama

Çevre algılama, nehir suyu seviyeleri ve kalitesi, atmosferik kirleticiler ve diğer çevre verileri hakkında bilgi toplamak için telemetri kullanır.

Sensörler genellikle uzak yerlerde bulunur, kablolu iletişime erişim olmadan. Kablosuz bant genişliği pahalı ve güvenilirlik düşük olabilir. Genellikle, küçük bir coğrafi bölgedeki bir dizi ortam algılayıcısı, güvenli bir konumdaki yerel bir izleme aygıtına bağlanır. Yerel bağlantılar kablolu ya da kablosuz olabilir.

Görünmezlik

Sensör aygıtlarının merkezi izleme aygıtından daha az erişilebilir, daha düşük güçlü ve daha fazla sayıda devreye alınmış olması olasıdır. Sensörler bazen "aptal" olur ve yerel izleme aygıtı, sensör verilerini dönüştürmek ve depolamak için bağdaştırıcılar içerir. İzleme aygıtı, Java Platform, Standard Edition (Java SE) ya da Java Platform, Micro Edition (Java ME) işletim sistemini destekleyen genel amaçlı bir bilgisayar içerir. MQTT istemcisi yapılandırılırken görünmezliğin önemli bir gereksinim olması olası değildir.

Eşit olmayan bağlanırlık

Sensörlerin yetenekleri ve uzak bağlantının maliyeti ve bant genişliği, genellikle merkezi bir sunucuya bağlı yerel bir izleme göbeğiyle sonuçlanır.

Durumu

Çözüm askeri veya savunma amaçlı bir kullanım durumunda kullanılmadığı sürece, güvenlik önemli bir gereksinim değildir.

Bağlanırlık

Birçok kullanım, verilerin sürekli olarak izlenmesini veya anında kullanılabilirliğini gerektirmez. Sel seviyesi uyarısı gibi kural dışı durum verilerinin hemen iletilmesi gerekir. Sensör verileri, bağlantı ve

iletişim maliyetlerini azaltmak için yerel monitörde toplanır ve daha sonra, zamanlanan bağlantılar kullanılarak aktarılır. Kural dışı durum verileri, izleme programında algılanır algılanmaz iletilir.

Ölçeklenebilirlik

Sensörler yerel göbekler etrafında yoğunlaşır ve sensör verileri bir zaman çizelgesine göre iletilen paketlere toplanır. Her iki faktör de, doğrudan bağlı sensörler kullanılarak empoze edilecek merkezi sunucu üzerindeki yükü azaltır.

Linux Windows AIX Telemetri kullanım senaryoları: Mobil uygulamalar

Mobil uygulamalar, kablosuz aygıtlarda çalışan uygulamalardır. Aygıtlar genel uygulama platformları ya da özel aygıtlardır.

Genel platformlar, telefonlar ve kişisel veri yardımcıları gibi elde taşınabilir aygıtları ve dizüstü bilgisayarlar gibi taşınabilir aygıtları içerir. Özel aygıtlar, belirli uygulamalara uyarlanmış özel amaçlı donanım kullanır. "signed-for" paketi teslimini kaydetmek için bir aygıt, özel bir mobil aygıta örnektir. Özel mobil aygıtlardaki uygulamalar genellikle genel bir yazılım platformunda oluşturulmuştur.

Görünmezlik

Özel mobil uygulamaların devreye alınması yönetilir ve MQTT istemci uygulamasının yapılandırmasını içerebilir. MQTT istemcisi yapılandırılırken görünmezliğin önemli bir gereksinim olması olası değildir.

Eşit olmayan bağlanırlık

Önceki kullanım senaryolarının yerel hub topolojisinden farklı olarak, mobil istemciler uzaktan bağlanır. İstemci uygulama katmanı, merkezi merkezdeki bir uygulamaya doğrudan bağlanır.

Durumu

Çok az fiziksel güvenlik, mobil aygıt ve mobil kullanıcının kimliği doğrulanmalıdır. Aygıtın kimliğini doğrulamak için TLS, kullanıcının kimliğini doğrulamak için JAAS kullanılır.

Bağlanırlık

Mobil uygulama kablosuz kapsama bağlı ise, çevrimdışı çalışabilmeli ve kesintili bir bağlantıyla verimli bir şekilde çalışabilmelidir. Bu ortamda amaç bağlı kalmaktır, ancak uygulama iletileri saklayıp iletebilmelidir. İletiler genellikle siparişler ya da teslimat onaylarıdır ve önemli iş değerine sahiplerdir. Güvenilir bir şekilde saklanmaları ve iletilmeleri gerekiyor.

Ölçeklenebilirlik

Ölçeklenebilirlik önemli bir sorun değil. Özel mobil uygulama kullanım senaryolarında uygulama istemcilerinin sayısı binleri ya da on binleri aşmayacaktır.

Linux Windows AIX Telemetri aygıtlarının kuyruk yöneticisine bağlanması

Telemetri aygıtları, MQTT v3 istemcisini kullanarak bir kuyruk yöneticisine bağlanır. MQTT v3 istemcisi, telemetry (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. IBM MQ istemcileri gibi, ileti göndermek ve almak için bir MQTT istemcisi de kuyruk yöneticisine bağlanmalıdır. Bağlantı, MQ Telemetryile kurulan ve telemetry (MQXR) hizmeti adı verilen bir TCP/IP dinleyicisi kullanılarak sunucuda yapılır. Her kuyruk yöneticisi en çok bir telemetri (MQXR) hizmeti çalıştırır.

Telemetri (MQXR) hizmeti, bir telemetri kanalına bağlantıyı ayırmak için `MqttClient.connect` yönteminde her istemci tarafından ayarlanan uzak yuva adresini kullanır. Yuva adresi, TCP/IP anasistem adı ve kapı numarasının birleşimidir. Aynı uzak yuva adresini kullanan birden çok istemci, telemetri (MQXR) hizmeti tarafından aynı telemetri kanalına bağlanır.

Bir sunucuda birden çok kuyruk yöneticisi varsa, telemetri kanallarını kuyruk yöneticileri arasında bölüştün. Uzak yuva adreslerini kuyruk yöneticileri arasında ayırın. Her bir telemetri kanalını benzersiz bir uzak yuva adresiyle tanımlayın. İki telemetri kanalı aynı yuva adresini kullanmamalıdır.

Aynı uzak yuva adresi birden çok kuyruk yöneticisinde telemetri kanalları için yapılandırılırsa, bağlanacak ilk telemetri kanalı kazanır. Aynı adrese bağlanan sonraki kanallar başarısız oldu.

Sunucuda birden çok ağ bağdaştırıcısı varsa, uzak yuva adreslerini telemetri kanalları arasında bölüştün. Belirli bir yuva adresi tek bir telemetri kanalında yapılandırıldığı sürece yuva adreslerinin ayrılması tamamen isteğe bağlı bir işlemdir.

IBM MQ Exploreriçin MQ Telemetry ekinde sağlanan sihirbazları kullanarak MQTT istemcilerini bağlamak için IBM MQ yapılandırın. Diğer bir seçenek olarak, telemetriyi el ile yapılandırmak için Linux ve AIX üzerinde telemetri için bir kuyruk yöneticisinin yapılandırılması ve Windows üzerinde bir kuyruk yöneticisinin telemetri için yapılandırılması başlıklı bölümdeki yönergeleri izleyin.

İlgili başvurular

MQXR özellikleri

Linux Windows AIX Telemetri bağlantı iletişim kuralları

MQ Telemetry , TCP/IP IPv4 ve IPv6ve TLS ' yi destekler.

Linux Windows AIX Telemetry (MQXR) hizmeti

Telemetry (MQXR) hizmeti, IBM MQ hizmeti olarak yönetilen bir TCP/IP dinleyicidir. Bir IBM MQ Explorer sihirbazını kullanarak ya da bir **runmqsc** komutuyla hizmeti oluşturun.

MQ Telemetry (MQXR) hizmeti SYSTEM.MQXR.SERVICE .

IBM MQ Exploreriçin MQ Telemetry işlevinde sağlanan Telemetry örnek yapılandırma sihirbazı, telemetri hizmetini ve örnek bir telemetri kanalını oluşturur; bkz. IBM MQ Explorer komutunu kullanarak MQ Telemetry kuruluşunun doğrulanması .

Komut satırından örnek yapılandırma oluşturun; bkz. Komut satırını kullanarak MQ Telemetry kuruluşunun doğrulanması.

Telemetri (MQXR) hizmeti, kuyruk yöneticisiyle otomatik olarak başlar ve durur. IBM MQ Exploreriçindeki Services klasörünü kullanarak hizmeti denetleyin. Hizmeti görmek için, IBM MQ Explorer SYSTEM nesnelerini görüntüden süzmeyi durdurmak üzere simgeyi tıklatmanız gerekir.

Hizmetin el ile nasıl oluşturulacağına ilişkin bir örnek için bkz.

- Linux AIX Linux üzerinde SYSTEM.MQXR.SERVICE oluşturma.
- Windows Windows üzerinde SYSTEM.MQXR.SERVICE oluşturma.

V 9.3.0 IBM MQ 9.3.0 ' den itibaren Linux üzerinde SYSTEM.MQXR.SERVICE oluşturmave Windows üzerinde SYSTEM.MQXR.SERVICE oluşturma , MQTT TLS kanallarının şifrelenmesi için geçiş tümcecik kullanılmasını gerektirecek varsayılan anahtarı belirtecek şekilde güncellenir. Daha fazla bilgi için bkz. MQTT TLS kanalları için geçiş tümceleri şifrelemesi.

Linux Windows AIX Telemetri kanalları

Java Kimlik Doğrulama ve Yetkilendirme Hizmeti (JAAS) ya da TLS kimlik doğrulaması gibi farklı özelliklere sahip bağlantılar oluşturmak ya da istemci gruplarını yönetmek için telemetri kanalları oluşturun.

IBM MQ Exploreriçin MQ Telemetry işlevinde sağlanan **New Telemetry Channel** sihirbazını kullanarak Telemetry kanalları oluşturun. Belirli bir TCP/IP kapısındaki MQTT istemcilerinden gelen bağlantıları kabul etmek için sihirbazı kullanarak bir kanal yapılandırın. IBM WebSphere MQ 7.1'den bu yana, komut satırı programını kullanarak MQ Telemetry ' ı yapılandırabilirsiniz (**runmqsc**).

İstemcileri gruplara ayırarak çok sayıda istemci bağlantısının daha kolay yönetilmesini sağlamak için farklı kapılarda birden çok telemetri kanalı oluşturun. Her telemetri kanalının farklı bir adı vardır.

Farklı bağlantı tipleri oluşturmak için farklı güvenlik öznitelikleriyle telemetri kanalları yapılandırabilirsiniz. Farklı TCP/IP adreslerinde istemci bağlantılarını kabul etmek için birden çok kanal yaratın. İletileri şifrelemek ve telemetri kanalını ve istemcisini doğrulamak için TLS ' yi kullanın; bkz. MQTT istemcilerinin ve telemetri kanallarının TLS yapılandırması. IBM MQ nesnelerine erişim yetkisini basitleştirmek için kullanıcı kimliğini belirtin. JAASile MQTT kullanıcısının kimliğini 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, düşük bant genişliğindeki ya da pahalı bağlantılardaki küçük aygıtlar arasında ileti alışverişi yapmak ve güvenilir bir şekilde ileti göndermek için tasarlanmıştır. TCP/IP kullanıyor.

MQTT protocol yayınlanır; bkz. IBM MQ Telemetry Transport biçimi ve iletişim kuralı. Protokolün 3. sürümü yayınlama/abone olma özelliğini kullanır ve şu üç hizmet niteliğini destekler: *fire and unut*, *en az bir kez*ve *tam olarak bir kez*.

Protokol üstbilgilerinin küçük boyutu ve bayt dizisi iletisi bilgi yükü, iletileri küçük tutar. Üstbilgiler, 2 baytlık sabit bir üstbilgi ve 12 bayta kadar ek değişken üstbilgisi içerir. Protokol, abone olmak ve bağlanmak için 12 baytlık değişken üstbilgilerini ve çoğu yayın için yalnızca 2 baytlık değişken üstbilgilerini kullanır.

Üç hizmet kalitesiyle düşük gecikme süresi ve güvenilirlik arasında geçiş yapabilirsiniz; bkz. MQTT istemcisi tarafından sağlanan hizmet nitelikleri. *Yangın ve unutma* , kalıcı aygıt depolama alanı kullanmaz ve yayın göndermek ya da almak için yalnızca bir iletim kullanır. *En az bir kez*ve *tam olarak bir kez* , iletişim kuralı durumunu korumak ve ileti onaylanıncaya kadar iletiyi kaydetmek için aygıtta kalıcı depolama gerektirir.

Linux Windows AIX MQTT müşterileri

MQTT istemci uygulaması, telemetri aygıtından bilgi toplamaktan, sunucuya bağlanmaktan ve bilgileri sunucuda yayınlamaktan sorumludur. Ayrıca, konulara abone olabilir, yayınları alabilir ve telemetri aygıtını kontrol edebilir.

IBM MQ istemci uygulamalarından farklı olarak, MQTT istemci uygulamaları IBM MQ uygulamaları değildir. Bağlantı kurmak için bir kuyruk yöneticisi belirtmezler. Bunlar, belirli IBM MQ programlama arabirimlerini kullanmakla sınırlı değildir. Bunun yerine, MQTT istemcileri MQTT 3 iletişim kuralını uygular. Kendi istemci kitaplığınızı, programlama dilinde ve platform üzerinde MQTT protocol ile arabirim oluşturacak şekilde yazabilirsiniz. Bkz. IBM MQ Telemetry Transport biçimi ve iletişim kuralı.

MQTT istemci uygulamaları yazmayı basitleştirmek için, çeşitli platformlar için MQTT protocol ürününü saran C, Javave JavaScript istemci kitaplıklarını kullanın. Bu kitaplıkları MQTT uygulamalarınıza dahil ederseniz, tam işlevsel bir MQTT istemcisi 15 satır kod kadar kısa olabilir. MQTT istemci kitaplıklarını Eclipse Paho ve MQTT.orgürünlerinden ücretsiz olarak edinebilirsiniz. Bkz. IBM MQ Telemetry Transport örnek programları.

MQTT istemci uygulaması her zaman bir telemetri kanalı ile bağlantı başlatmaktan sorumludur. Bağlandıktan sonra, MQTT istemci uygulaması ya da IBM MQ uygulaması bir ileti alışverişi başlatabilir.

MQTT istemci uygulamaları ve IBM MQ uygulamaları aynı konu kümesini yayınlar ve bunlara abone olur. IBM MQ uygulaması, öncelikle bir abonelik oluşturmadan bir iletiyi doğrudan MQTT istemci uygulamasına gönderebilir. Bkz. MQTT istemcilerine ileti göndermek için dağıtılmış kuyruğa alma yapılandırılması.

MQTT istemci uygulamaları, bir telemetri kanalı kullanılarak IBM MQ ' e bağlanır. Telemetri kanalı, MQTT ve IBM MQtarafından kullanılan farklı ileti tipleri arasında bir köprü görevi görür. MQTT istemci uygulaması adına kuyruk yöneticisinde yayınlar ve abonelikler oluşturur. Telemetri kanalı, bir MQTT istemci uygulamasının abonelikleriyle eşleşen yayınları kuyruk yöneticisinden MQTT istemci uygulamasına gönderir.

Linux Windows AIX MQTT istemcisine ileti gönderilmesi

IBM MQ uygulamaları, istemciler tarafından oluşturulan aboneliklere yayınlayarak ya da doğrudan ileti göndererek MQTT v3 istemcisi iletileri gönderebilir. MQTT istemcileri, diğer istemciler tarafından abone olunan konuları yayınlayarak birbirlerine ileti gönderebilirler.

MQTT istemcisi, IBM MQ ' den aldığı bir yayına abone olur.

IBM MQ olanağından MQTT istemcisine bir yayın göndermek için “IBM MQ Explorer adresinden MQTT istemci yardımcı programına ileti yayınlanması” sayfa 116 görevini gerçekleştirin.

MQTT v3 istemcisinin iletileri almasının standart yolu, bir konuya ya da konu kümesine abonelik oluşturmaktır. Örnek kod parçacığı Şekil 44 sayfa 114' da, MQTT istemcisi `"MQTT Examples"`konu dizgisini kullanarak abone olur. Bir IBM MQ C uygulaması, Şekil 45 sayfa 115, `"MQTT Examples"`konu dizgisini kullanarak konuya yayınlar. Şekil 46 sayfa 115kod parçacığının içinde MQTT istemcisi, `messageArrived`geri çağırma yönteminde yayını alır.

IBM MQ ' in MQTT istemcilerinden gelen aboneliklere yanıt olarak yayınları göndermesi için nasıl yapılandırılacağına ilişkin daha fazla bilgi için MQTT istemci aboneliğine yanıt olarak bir ileti yayınlamabaşlıklı konuya bakın.

IBM MQ uygulaması bir iletiyi doğrudan MQTT istemcisine gönderir

Bir iletiyi doğrudan IBM MQ içinden MQTT istemcisine göndermek için “IBM MQ Explorer kullanarak MQTT istemcisine ileti gönderilmesi” sayfa 120 görevini gerçekleştirin.

MQTT istemcisine bu şekilde gönderilen bir iletiye istenmeyen ileti denir. MQTT v3 istemcileri, konu adı kümesi olan yayınlar olarak istenmeyen iletiler alır. Telemetry (MQXR) hizmeti, konu adını uzak kuyruk adına ayarlar.

IBM MQ ' in MQTT istemcilerine doğrudan ileti gönderecek şekilde nasıl yapılandırılacağına ilişkin ek bilgi için İstemciye doğrudan ileti gönderilmesibaşlıklı konuya bakın.

MQTT istemcisi bir ileti yayınlar

MQTT v3 istemcisi, başka bir MQTT v3 istemcisi tarafından alınan bir iletiyi yayınlayabilir, ancak istenmeyen bir ileti gönderemez. Kod parçacığı Şekil 47 sayfa 115 , Javaiçinde yazılan bir MQTT v3 istemcisinin bir iletiyi nasıl yayınladığını gösterir.

Belirli bir MQTT v3 istemcisine ileti göndermeye ilişkin tipik örüntü, her istemcinin kendi `ClientIdentifier`aboneliği yaratmasıdır. Konu dizgisi olarak `ClientIdentifier` kullanarak bir MQTT istemcisinden başka bir MQTT istemcisine ileti yayınlamak için “Belirli bir MQTT v3 istemcisinde ileti yayınlanması” sayfa 121 görevini kullanın.

Örnek kod parçacıkları

Şekil 44 sayfa 114 içindeki kod parçacığı, Java içinde yazılan bir MQTT istemcisinin nasıl abonelik oluşturduğunu gösterir. Aboneliğe ilişkin yayınları almak için bir geri çağırma yöntemi de (`messageArrived`) gerekir.

```
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 44. MQTT v3 istemci abonesi

Şekil 45 sayfa 115 içindeki kod parçacığı, C dilinde yazılmış bir IBM MQ uygulamasının bir yayını nasıl gönderdiğini gösterir. Kod parçacığı görevden çıkarılır, Değişken bir konu için 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 45. IBM MQ publisher

Yayın geldiğinde, MQTT istemcisi MQTT uygulama istemcisi `MqttCallback` sınıfının `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 46. `messageArrived` yöntemi

Şekil 47 sayfa 115 içinde, Şekil 44 sayfa 114içinde oluşturulan aboneliğe ileti yayınlayan bir MQTT v3 gösterilmektedir.

```
      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 47. MQTT v3 istemci yayınlayıcısı

Linux Windows AIX IBM MQ Explorer adresinden MQTT istemci yardımcı programına ileti yayınlanması

IBM MQ Explorerkomutunu kullanarak bir iletiyi yayınlamak ve MQTT istemci yardımcı programıyla bu iletiye abone olmak için bu görevdeki adımları izleyin. Ek bir görev, varsayılan iletim kuyruğunu `SYSTEM.MQTT.TRANSMIT.QUEUE`olarak ayarlamak yerine, kuyruk yöneticisi diğer adını nasıl yapılandıracağınızı gösterir.

Başlamadan önce

Görev, IBM MQ ve IBM MQ Explorerile ilgili bilgi sahibi olduğunuzu ve IBM MQ ve MQ Telemetry özelliğinin kurulu olduğunu varsayar.

Bu göreve ilişkin kuyruk yöneticisi kaynaklarını yaratan kullanıcının bunun için yeterli yetkisi olmalıdır. Gösterim amacıyla, IBM MQ Explorer kullanıcı kimliğinin `mqm` grubunun üyesi olduğu varsayılır.

Bu görev hakkında

Görevde, IBM MQ içinde bir konu oluşturur ve MQTT istemci yardımcı programını kullanarak konuya abone olabilirsiniz. IBM MQ Explorerkomutunu kullanarak konuda yayınlama yaptığınızda, MQTT istemcisi yayını alır.

Yordam

Aşağıdaki görevlerden birini gerçekleştirin:

- MQ Telemetryürününü kurdunuz, ancak henüz başlatmadınız. Görevi yapın: “Görevi henüz telemetri (MQXR) hizmeti tanımlanmamış olarak başlat” sayfa 116.
- IBM MQ telemetrisini daha önce çalıştırdınız, ancak gösterimi gerçekleştirmek için yeni bir kuyruk yöneticisi kullanmak istiyorsunuz. Görevi yapın: “Görevi henüz telemetri (MQXR) hizmeti tanımlanmamış olarak başlat” sayfa 116.
- Tanımlanmış telemetri kaynağı olmayan var olan bir kuyruk yöneticisini kullanarak görevi gerçekleştirmek istiyorsunuz. **Örnek yapılanışı tanımla** sihirbazını çalıştırmak istemezsiniz.
  a. Telemetri kurmak için aşağıdaki görevlerden birini gerçekleştirin:
     – Linux ve AIX üzerinde telemetri için bir kuyruk yöneticisinin yapılandırılması
     – Windows üzerinde bir kuyruk yöneticisinin telemetri için yapılandırılması
  b. Görevi yapın: “Görevi çalışan bir telemetri (MQXR) hizmetiyle başlat” sayfa 117
- Görevi, önceden tanımlanmış telemetri kaynakları olan var olan bir kuyruk yöneticisini kullanarak gerçekleştirmek istiyorsanız şu görevi gerçekleştirin: “Görevi çalışan bir telemetri (MQXR) hizmetiyle başlat” sayfa 117.

Sonraki adım

Bir iletiyi doğrudan istemci yardımcı programına göndermek için “IBM MQ Explorer kullanarak MQTT istemcisine ileti gönderilmesi” sayfa 120 adresini kullanın.

Görevi henüz telemetri (MQXR) hizmeti tanımlanmamış olarak başlat

Bir kuyruk yöneticisi oluşturun ve kuyruk yöneticisine ilişkin örnek telemetri kaynaklarını tanımlamak için **Örnek yapılandırma tanımla** ' yı çalıştırın. IBM MQ Explorerkullanarak bir ileti yayınlayın ve MQTT istemci yardımcı programıyla iletiye abone olun.

Bu görev hakkında

Örnek telemetri kaynaklarını **Örnek yapılandırmayı tanımla**' yı kullanarak ayarladığınızda, sihirbaz konuk kullanıcı kimliği izinlerini ayarlar. Konuk kullanıcı kimliğinin bu şekilde yetkilendirilmesini istiyorsanız dikkatli olun. `guest` on Windowsve `nobody` on Linux, konu ağacının kökünü yayınlama ve bunlara abone olma ve iletileri `SYSTEM.MQTT.TRANSMIT.QUEUE`üzerine koyma izni verilir.

Sihirbaz ayrıca, varsayılan iletim kuyruğunu SYSTEM.MQTT.TRANSMIT.QUEUEolarak ayarlar; bu, var olan bir kuyruk yöneticisinde çalışan uygulamaları etkileyebilir. Telemetriyi yapılandırmak ve varsayılan iletim kuyruğunu kullanmamak mümkündür, ancak zahmetlidir; görevde aşağıdaki işlemi gerçekleştirin: "Kuyruk yöneticisi diğer adının kullanılması" sayfa 118. Bu görevde, varolan varsayılan iletim kuyruğuna müdahale olasılığını önlemek için bir kuyruk yöneticisi yaratırsınız.

Yordam

1. IBM MQ Explorerkomutunu kullanarak yeni bir kuyruk yöneticisi yaratın ve başlatın.
   a) Queue Managers Klasör > **Yeni** > **Kuyruk yöneticisi ...**öğelerini sağ tıklatın. Bir kuyruk yöneticisi adı yazın > **Son**.

   Bir kuyruk yöneticisi adı belirleyin; 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.

   MQTT Client Utility programını başlat onay kutusunu işaretli 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 Explorer**> **Yeni** > **Konu**içinde Queue Managers*QmgrName*\Topics klasörünü sağ tıklatın.
   b) **Ad** > **İleri**olarak MQTTExampleTopic yazın.
   c) **Konu dizgisi** > **Son**olarak MQTT Example yazın.
   d) Alındı bildirimi penceresini kapatmak için **Tamam** düğmesini tıklatın.
5. IBM MQ Explorerkomutunu kullanarak Hello World! başlıklı konuyu MQTT Example yayınlayın.
   a) IBM MQ Exploreriçindeki Queue Managers*QmgrName*\Topics klasörünü tıklatın.
   b) MQTTExampleTopic > **Test yayını ...** seçeneğini sağ tıklatın.
   c) **İleti verileri** alanına Hello World! yazın > **İleti yayınla** > MQTT Client Utility penceresine geçin.

   İstemci geçmişi bir Received olayını kaydeder.

Görevi çalışan bir telemetri (MQXR) hizmetiyle başlat

Bir telemetri kanalı ve bir konu oluşturun. Kullanıcıya konuyu ve telemetri iletim kuyruğunu kullanma yetkisi verin. IBM MQ Explorerkullanarak bir ileti yayınlayın ve MQTT istemci yardımcı programıyla iletiye abone olun.

Başlamadan önce

Görevin bu sürümünde bir kuyruk yöneticisi (*QmgrName*) tanımlanır ve çalışır. Bir telemetri (MQXR) hizmeti tanımlandı ve çalışıyor. Telemetry (MQXR) hizmeti el ile ya da **Örnek yapılandırmayı tanımla** sihirbazı çalıştırılarak yaratılmış olabilir.

Bu görev hakkında

Bu görevde, var olan bir kuyruk yöneticisini MQTT istemcisi yardımcı programına bir yayın gönderecek şekilde yapılandırırsınız.

Görevin "1" sayfa 118 numaralı adımı, varsayılan iletim kuyruğunu SYSTEM.MQTT.TRANSMIT.QUEUEolarak ayarlar; bu, var olan bir kuyruk yöneticisinde çalışan uygulamaları etkileyebilir. Telemetriyi yapılandırmak ve varsayılan iletim kuyruğunu kullanmamak

mümkündür, ancak zahmetlidir; görevde aşağıdaki işlemi gerçekleştirin: “Kuyruk yöneticisi diğer adının kullanılması” sayfa 118.

Yordam

1. Varsayılan iletim kuyruğu olarak SYSTEM.MQTT.TRANSMIT.QUEUE değerini belirleyin.
   a) Queue Managers*QmgrName* folder > **Özellikler ...** seçeneğini sağ tıklatın.
   b) Gezginde **Communication** (İletişim) seçeneğini tıklatın.
   c) **Seç ...** > SYSTEM.MQTT.TRANSMIT.QUEUE > **Tamam** > **Tamam**seçeneklerini tıklatın.
2. MQTT istemci yardımcı programını IBM MQ' e bağlamak için bir telemetri kanalı MQTTExampleChannel oluşturun ve MQTT istemci yardımcı programını başlatın.
   a) **MQ Explorer**> **Yeni** > **Telemetry channel ...**içinde Queue Managers*QmgrName* \Telemetry\Channels klasörünü sağ tıklatın.
   b) **Kanal adı** alanına MQTTExampleChannel yazın > **Sonraki** > **Sonraki**.
   c) İstemci yetkilendirme panosundaki **Sabit kullanıcı kimliği** değerini, yayınlayacak ve MQTTExample > **İleri**' ye abone olacak kullanıcı kimliğiyle değiştirin.
   d) **İstemci yardımcı programını başlat** seçeneğini işaretli olarak bırakın > **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 Explorer**> **Yeni** > **Konu**içinde Queue Managers*QmgrName*\Topics klasörünü sağ tıklatın.
   b) **Ad** > **İleri**olarak MQTTExampleTopic yazın.
   c) **Konu dizgisi** > **Son**olarak MQTT Example yazın.
   d) Alındı bildirimi penceresini kapatmak için **Tamam** düğmesini tıklatın.
5. mqm grubunda olmayan bir kullanıcının MQTTExample konusunu yayınlamasını ve bu konuya abone olmasını istiyorsanız, aşağıdakileri yapın:
   a) Kullanıcıya MQTTExampleTopickonusunu yayınlama ve bu konuya abone olma yetkisi verme:

      ```
      setmqaut -m qMgrName -t topic -n MQTTExampleTopic -p User ID -all +pub +sub
      ```

   b) Kullanıcıya SYSTEM.MQTT.TRANSMIT.QUEUEüzerine bir ileti yerleştirmesi için yetki ver:

      ```
      setmqaut -m qMgrName -t q -n SYSTEM.MQTT.TRANSMIT.QUEUE -p User ID -all +put
      ```

6. IBM MQ Explorerkomutunu kullanarak Hello World! başlıklı konuyu MQTT Example yayınlayın.
   a) IBM MQ Exploreriçindeki Queue Managers*QmgrName*\Topics klasörünü tıklatın.
   b) MQTTExampleTopic > **Test yayını ...** seçeneğini sağ tıklatın.
   c) **İleti verileri** alanına Hello World! yazın > **İleti yayınla** > MQTT Client Utility penceresine geçin.

   İstemci geçmişi bir Received olayını kaydeder.

Kuyruk yöneticisi diğer adının kullanılması

Varsayılan iletim kuyruğunu SYSTEM.MQTT.TRANSMIT.QUEUEolarak ayarlamadan IBM MQ Explorer komutunu kullanarak MQTT istemci yardımcı programında bir ileti yayınlayın.

Görev, önceki görevin bir devamıdır ve varsayılan iletim kuyruğunun SYSTEM.MQTT.TRANSMIT.QUEUEolarak ayarlanmasını önlemek için bir kuyruk yöneticisi diğer adını kullanır.

Başlamadan önce

"Görevi henüz telemetri (MQXR) hizmeti tanımlanmamış olarak başlat" sayfa 116 görevini ya da "Görevi çalışan bir telemetri (MQXR) hizmetiyle başlat" sayfa 117görevini tamamlayın.

Bu görev hakkında

Bir MQTT istemcisi abonelik yarattığında IBM MQ , uzak kuyruk yöneticisi adı olarak ClientIdentifierkullanarak yanıt gönderir. Bu görevde, ClientIdentifier(MyClient) değerini kullanır.

MyClientadlı bir iletim kuyruğu ya da kuyruk yöneticisi diğer adı yoksa, yanıt varsayılan iletim kuyruğuna yerleştirilir. Varsayılan iletim kuyruğu SYSTEM.MQTT.TRANSMIT.QUEUEolarak ayarlandığında, MQTT istemcisi yanıtı alır.

Kuyruk yöneticisi diğer adlarını kullanarak varsayılan iletim kuyruğunu SYSTEM.MQTT.TRANSMIT.QUEUE olarak ayarlamaktan kaçınabilirsiniz. Her ClientIdentifieriçin bir kuyruk yöneticisi diğer adı ayarlamanız gerekir. Genellikle, kuyruk yöneticisi diğer adlarını kullanmayı pratik hale getiremeyecek kadar çok istemci vardır. Genellikle ClientIdentifier önceden kestirilemez ve telemetrinin bu şekilde yapılandırılmasını imkansız hale getirir.

Bununla birlikte, bazı durumlarda varsayılan iletim kuyruğunu SYSTEM.MQTT.TRANSMIT.QUEUEdışında bir değere yapılandırmanız gerekebilir. Yordam içindeki adımlar, varsayılan iletim kuyruğunu SYSTEM.MQTT.TRANSMIT.QUEUEolarak ayarlamak yerine bir kuyruk yöneticisi diğer adını yapılandırıyor.

Yordam

1. Varsayılan iletim kuyruğu olarak SYSTEM.MQTT.TRANSMIT.QUEUE öğesini kaldırın.
   a) Queue Managers*QmgrName* folder > **Özellikler ...** seçeneğini sağ tıklatın.
   b) Gezginde **Communication** (İletişim) seçeneğini tıklatın.
   c) SYSTEM.MQTT.TRANSMIT.QUEUE öğesini **Varsayılan iletim kuyruğu** alanından > **Tamam**' ı kaldırın.
2. MQTT istemci yardımcı programıyla artık abonelik oluşturamayabileceğinizi 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 Connection lost olayını kaydeder.
3. ClientIdentifier, MyClientiçin bir kuyruk yöneticisi diğer adı yaratın.
   a) Queue Managers*QmgrName*\Queues klasör > **Yeni** > **Uzak kuyruk tanımlaması**öğelerini farenin sağ düğmesiyle tıklatın.
   b) Tanımlamayı MyClient > **İleri**olarak adlandırın.
   c) **Uzak kuyruk yöneticisi** alanına MyClient yazın.
   d) **İletim kuyruğu** alanında SYSTEM.MQTT.TRANSMIT.QUEUE yazın > **Bitir**.
4. MQTT istemci yardımcı programını yeniden bağlayın.
   a) **İstemci tanıtıcısı** ' nı MyClientolarak ayarlayın.
   b) **Bizimle**

   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. IBM MQ Explorerkomutunu kullanarak `Hello World!` başlıklı konuyu `MQTT Example` yayınlayın.
   a) IBM MQ Exploreriçindeki `Queue Managers\`*`QmgrName`*`\Topics` klasörünü tıklatın.
   b) `MQTTExampleTopic` > **Test yayını ...** seçeneğini sağ tıklatın.
   c) **İleti verileri** alanına `Hello World!` yazın > **İleti yayınla** > MQTT Client Utility penceresine geçin.

   İstemci geçmişi bir `Received` olayını kaydeder.

Linux Windows AIX IBM MQ Explorer kullanarak MQTT istemcisine ileti gönderilmesi

IBM MQ Explorerkomutunu kullanarak IBM MQ kuyruğuna bir ileti koyarak MQTT istemci yardımcı programına bir ileti gönderin. Bu görev, bir uzak kuyruk tanımlamasının doğrudan MQTT istemcisine ileti gönderecek şekilde nasıl yapılandırılacağını gösterir.

Başlamadan önce

Görevi yapın, “IBM MQ Explorer adresinden MQTT istemci yardımcı programına ileti yayınlanması” sayfa 116. MQTT istemci yardımcı programını bağlı bırakın.

Bu görev hakkında

Bu görev, bir konuyu yayınlamak yerine kuyruğu kullanan bir MQTT istemcisine ileti gönderilmesini gösterir. İstemcide abonelik oluşturmuyorsunuz. Görevin “2” sayfa 120 numaralı adımı, önceki aboneliğin silindiğini gösterir.

Yordam

1. MQTT istemcisi yardımcı programının bağlantısını keserek ve yeniden bağlayarak var olan abonelikleri atın.

   Varsayılan değerleri değiştirmediğiniz sürece MQTT istemci yardımcı programı temiz bir oturumla bağlantı kurduğundan abonelik atılır; bkz. Temiz oturumlar.

   Görevin daha kolay yapılması için, MQTT istemci yardımcı programı tarafından oluşturulan `ClientIdentifier`'ı kullanmak yerine kendi `ClientIdentifier`yazın.

   a) MQTT istemci yardımcı programının telemetri kanalından bağlantısını kesmek için **Disconnect** (Bağlantıyı Kes) seçeneğini tıklatın.

   İstemci Geçmişi bir `Disconnected` olayını kaydeder

   b) **Client Identifer** değerini `MyClient`olarak değiştirin.

   c) **Bağlan**'ı tıklayın.

   İstemci Geçmişi bir `Connected` olayını kaydeder
2. MQTT istemci yardımcı programının artık `MQTTExampleTopic`için yayın almadığını denetleyin.
   a) IBM MQ Exploreriçindeki `Queue Managers\`*`QmgrName`*`\Topics` klasörünü tıklatın.
   b) `MQTTExampleTopic` > **Test yayını ...** seçeneğini sağ tıklatın.
   c) **İleti verileri** alanına `Hello World!` yazın > **İleti yayınla** > MQTT Client Utility penceresine geçin.

   İstemci geçmişinehiçbir olay kaydedilmez.
3. İstemci için bir uzak kuyruk tanımlaması yaratın.

Uzak kuyruk tanımlamasında uzak kuyruk yöneticisi adı olarak `ClientIdentifier`(`MyClient`) değerini belirleyin. Uzak kuyruk adı olarak istediğiniz herhangi bir adı kullanın. Uzak kuyruk adı, konu adı olarak MQTT istemcisine geçirilir.

a) `Queue Managers\`*`QmgrName`*`\Queues` klasör > **Yeni** > **Uzak kuyruk tanımlaması**öğelerini farenin sağ düğmesiyle tıklatın.

b) Tanımlamayı `MyClientRemoteQueue` > **İleri**olarak adlandırın.

c) **Uzak kuyruk** alanına `MQTTExampleQueue` yazın.

d) **Uzak kuyruk yöneticisi** alanına `MyClient` yazın.

e) **İletim kuyruğu** alanında `SYSTEM.MQTT.TRANSMIT.QUEUE` yazın > **Bitir**.

4. `MyClientRemoteQueue`üzerine bir test iletisi koyun.

a) **MyClientRemoteQueue** > **Put test message ...** öğesini sağ tıklatın.

b) İleti verileri alanına `Hello queue!` yazın > **İletiyi koy** > **Kapat**

İstemci geçmişi bir `Received` olayını kaydeder.

5. Varsayılan iletim kuyruğu olarak `SYSTEM.MQTT.TRANSMIT.QUEUE` öğesini kaldırın.

a) `Queue Managers\`*`QmgrName`* `folder` > **Özellikler ...** seçeneğini sağ tıklatın.

b) Gezginde **Communication** (İletişim) seçeneğini tıklatın.

c) `SYSTEM.MQTT.TRANSMIT.QUEUE` öğesini **Varsayılan iletim kuyruğu** alanından > **Tamam**' ı kaldırın.

6. “4” sayfa 121. adımı yeniden yapın.

`MyClientRemoteQueue` , iletim kuyruğuna belirtik olarak ad veren bir uzak kuyruk tanımıdır. `MyClient`' e ileti göndermek için varsayılan iletim kuyruğunu tanımlamanız gerekmez.

Sonraki adım

Varsayılan iletim kuyruğu artık `SYSTEM.MQTT.TRANSMIT.QUEUE`olarak ayarlanmadığında, MQTT İstemci Yardımcı Programı, `MyClient` `ClientIdentifier`için bir kuyruk yöneticisi diğer adı tanımlanmadıkça yeni bir abonelik oluşturamaz. Varsayılan iletim kuyruğunu `SYSTEM.MQTT.TRANSMIT.QUEUE`olarak geri yükleyin.

Linux Windows AIX Belirli bir MQTT v3 istemcisinde ileti yayınlanması

Konu adı olarak `ClientIdentifier` ve yayınlama/abone olma aracısı olarak IBM MQ kullanarak bir MQTT v3 istemcisinden diğerine ileti yayınlayın.

Başlamadan önce

Görevi yapın, “IBM MQ Explorer adresinden MQTT istemci yardımcı programına ileti yayınlanması” sayfa 116. MQTT istemci yardımcı programını bağlı bırakın.

Bu görev hakkında

Görev iki şeyi gösterir:

1. Bir MQTT istemcisindeki bir konuya abone olma ve başka bir MQTT istemcisinden yayın alma.
2. Konu dizgisi olarak `ClientIdentifier` kullanarak "noktadan noktaya" abonelikleri ayarlama.

Yordam

1. MQTT istemcisi yardımcı programının bağlantısını keserek ve yeniden bağlayarak var olan abonelikleri atın.

Varsayılan değerleri değiştirmediğiniz sürece MQTT istemci yardımcı programı temiz bir oturumla bağlantı kurduğundan abonelik atılır; bkz. Temiz oturumlar.

Görevin daha kolay yapılması için, MQTT istemci yardımcı programı tarafından oluşturulan `ClientIdentifier` ' ı kullanmak yerine kendi `ClientIdentifier`yazın.

a) MQTT istemci yardımcı programının telemetri kanalından bağlantısını kesmek için **Disconnect** (Bağlantıyı Kes) seçeneğini tıklatın.

İstemci Geçmişi bir `Disconnected` olayını kaydeder

b) **Client Identifer** değerini `MyClient`olarak değiştirin.

c) **Bağlan**'ı tıklayın.

İstemci Geçmişi bir `Connected` olayını kaydeder

2. `MyClient` başlıklı konuya abonelik oluşturun

`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 programını başlatın.

a) `Queue Managers\`*`QmgrName`*`\Telemetry\channels` klasörünü açın.

b) **PlainText** kanalını sağ tıklatın > **MQTT Client Utility programını çalıştırın ...**

c) **Bağlan**'ı tıklayın.

İstemci Geçmişi bir `Connected` olayını kaydeder

4. `Hello MyClient!` öğesini `MyClient`konusunda yayınlayın.

a) `MyClient`abonelik konusunu, `MyClient` `ClientIdentifier`ile çalışan MQTT istemci yardımcı programından kopyalayın.

b) `MyClient` dosyasını, MQTT istemci yardımcı programı yönetim ortamlarının her birinin **Yayın \ Konu** alanına yapıştırın.

c) **Yayın \ ileti** alanına `Hello MyClient!` yazın.

d) Her iki eşgörünümde **Yayınla** ' yı tıklatın.

Sonuçlar

`MyClient` `ClientIdentifier`ile MQTT istemci yardımcı programındaki **İstemci geçmişi** , iki **Alınan** olayı ve bir **Yayınlanan** olayı kaydeder. Diğer MQTT istemcisi yardımcı programı bir **Yayınlandı** olayını kaydeder.

Yalnızca bir **Alınan** olayı görüyorsanız, aşağıdaki olası nedenleri denetleyin:

1. Kuyruk yöneticisi için varsayılan iletim kuyruğu `SYSTEM.MQTT.TRANSMIT.QUEUE` olarak ayarlandı mı?
2. Diğer alıştırmaları yaparken `MyClient` ' a gönderme yapan kuyruk yöneticisi diğer adlarını ya da uzak kuyruk tanımlamalarını yarattınız mı? Bir yapılandırma sorununuz varsa, kuyruk yöneticisi diğer adları ya da iletim kuyrukları gibi `MyClient`' e gönderme yapan kaynakları silin. İstemci yardımcı programlarının bağlantısını kesin, telemetry (MQXR) hizmetini durdurun ve yeniden başlatın.

Linux Windows AIX MQTT istemcisinden IBM MQ uygulamasına ileti gönderilmesi

IBM MQ uygulaması, bir konuya abone olarak MQTT v3 istemcisinden ileti alabilir. MQTT istemcisi, bir telemetri kanalı kullanarak IBM MQ ' e bağlanır ve aynı konuyu yayınlayarak IBM MQ uygulamasına bir ileti gönderir.

Bir yayının MQTT istemcisinden IBM MQiçinde tanımlanan bir aboneliğe nasıl gönderileceğini öğrenmek için “MQTT istemcisinden IBM MQ ' e ileti yayınlama” sayfa 123görevini gerçekleştirin.

Konu kümelendiyse ya da bir yayınlama/abone olma sıradüzeni kullanılarak dağıtıldıysa, abonelik MQTT istemcisinin bağlı olduğu kuyruk yöneticisine farklı bir kuyruk yöneticisinde olabilir.

Linux Windows AIX MQTT istemcisinden IBM MQ ' e ileti yayınlama

IBM MQ Explorer komutunu kullanarak bir konuya abonelik oluşturun ve MQTT istemci yardımcı programını kullanarak konuyu yayınlayın.

Başlamadan önce

Görevi yapın, “IBM MQ Explorer adresinden MQTT istemci yardımcı programına ileti yayınlanması” sayfa 116. MQTT istemci yardımcı programını bağlı bırakın.

Bu görev hakkında

Görev, bir MQTT istemcisiyle ileti yayınlanmasını ve IBM MQ Explorerkullanılarak oluşturulan yönetilmeyen bir sürekli abonelik kullanılarak yayının alınmasını gösterir.

Yordam

1. `MQTT Example`konu dizgisine kalıcı bir abonelik oluşturun.

   IBM MQ Explorerkomutunu kullanarak kuyruğu ve aboneliği oluşturmak için aşağıdaki adımları gerçekleştirin.

   a) IBM MQ Explorer> **Yeni** > **Yerel kuyrukta** `Queue Managers\`*`QmgrName`*`\Queues` klasörünü sağ tıklatın.

   b) Kuyruk adı olarak `MQTTExampleQueue` yazın > **Son**.

   c) IBM MQ Explorer> **Yeni** > **Abonelik ...**içinde `Queue Managers\`*`QmgrName`*`\Subscriptions` klasörünü sağ tıklatın.

   d) Kuyruk adı olarak `MQTTExampleSubscription` yazın > **İleri**.

   e) **Seç ...** > `MQTTExampleTopic` > **Tamam**düğmesini tıklatın.

   “IBM MQ Explorer adresinden MQTT istemci yardımcı programına ileti yayınlanması” sayfa 116' un `MQTTExampleTopic` . adımında “4” sayfa 117 konusunu zaten oluşturdunuz.

   f) Hedef adı olarak `MQTTExampleQueue` yazın > **Son**.

2. İsteğe bağlı bir adım olarak, kuyruğu `mqm` yetkisi olmadan farklı bir kullanıcı tarafından kullanılmak üzere ayarlayın.

   `mqm`' den daha az yetkisi olan kullanıcılar için yapılandırmayı ayarlıyorsanız, `MQTTExampleQueue`için `put` ve `get` yetkilerini vermeniz gerekir. Konuya ve iletim kuyruğuna erişim “IBM MQ Explorer adresinden MQTT istemci yardımcı programına ileti yayınlanması” sayfa 116içinde yapılandırıldı.

   a) Kullanıcıya kuyruğa koyma ve alma yetkisi ver `MQTTExampleQueue`:

   ```
   setmqaut -m qMgrName -t queue -n MQTTExampleQueue -p User ID -all +put +get
   ```

3. MQTT istemcisi yardımcı programını kullanarak `MQTT Example` konusunda `Hello IBM MQ!` yayınlayın.

   MQTT istemci yardımcı programını bağlı bırakmadıysanız, **PlainText** kanalını sağ tıklatın > **MQTT Client Utility programını çalıştırın ...** > **Connect**.

   a) **Yayın \ Konu** alanına `MQTT Example` yazın.

   b) **Yayın \ İleti** alanına `Hello IBM MQ!` yazın > **Yayınla**.

4. `Queue Managers\`*`QmgrName`*`\Queues` klasörünü açın ve `MQTTExampleQueue`dosyasını bulun.

   Yürürlükteki kuyruk derinliği alanı 1

5. `MQTTExampleQueue` > **İletilere göz at ...** öğelerini sağ tıklatın. ve yayını inceleyin.

Linux Windows AIX MQTT yayınlama/abone olma uygulamaları

MQTT uygulamaları yazmak için konu tabanlı yayınlama/abone olma özelliğini kullanın.

MQTT istemcisi bağlandığında, yayınlar istemci ile sunucu arasında her iki yönde de akar. Bilgiler istemcide yayınlandığında, yayınlar istemciden gönderilir. Yayınlar, istemci tarafından oluşturulan bir abonelikle eşleşen bir konuya ileti yayınlandığında istemcide alınır.

IBM MQ yayınlama/abone olma aracısı, MQTT istemcileri tarafından yaratılan konuları ve abonelikleri yönetir. MQTT istemcileri tarafından oluşturulan konular, IBM MQ uygulamaları tarafından oluşturulan konularla aynı konu alanını paylaşır.

MQTT istemci aboneliğinde konu dizgisiyle eşleşen yayınlar, uzak kuyruk yöneticisi adı istemcinin `ClientIdentifier` olarak ayarlanmış şekilde `SYSTEM.MQTT.TRANSMIT.QUEUE` üzerine yerleştirilir. Telemetry (MQXR) hizmeti, yayınları aboneliği yaratan istemciye iletir. İstemciyi tanıtmak için uzak kuyruk yöneticisi adı olarak ayarlanan `ClientIdentifier`değerini kullanır.

Genellikle, `SYSTEM.MQTT.TRANSMIT.QUEUE` varsayılan iletim kuyruğu olarak tanımlanmalıdır. MQTT ' in konfigürasyonunu varsayılan iletim kuyruğunu kullanmayacak şekilde tanımlamak mümkündür; bkz. Dağıtılmış kuyruğun MQTT istemcilere ileti gönderecek şekilde yapılandırılması.

MQTT istemcisi kalıcı bir oturum yaratabilir; bkz. “MQTT durumsuz ve durumlu oturumlar” sayfa 127. Kalıcı bir oturumda oluşturulan abonelikler dayanıklıdır. Kalıcı oturumu olan bir istemci için gelen yayınlar `SYSTEM.MQTT.TRANSMIT.QUEUE`içinde saklanır ve yeniden bağlandığında istemciye iletilir.

MQTT istemcisi, alıkonan yayınları yayınlayabilir ve bunlara abone olabilir; bkz. Alıkonan yayınlar ve MQTT istemcileri. Alıkonan bir yayın konusunun abonesi, konunun en son yayınını alır. Abone, bir abonelik oluşturduğunda ya da önceki oturumuna yeniden bağlandığında alıkonan yayını alır.

Linux Windows AIX Telemetri uygulamaları

IBM MQ ya da IBM Integration Bus ileti akışlarını kullanarak telemetri uygulamaları yazın.

IBM MQiçindeki telemetri uygulamalarını programlamak için JMS, MQI ya da diğer IBM MQ programlama arabirimlerini kullanın.

Telemetry (MQXR) hizmeti, MQTT v3 iletileri ile IBM MQ iletileri arasında dönüştürülür. MQTT istemcileri adına abonelikler ve yayınlar oluşturur ve yayınları MQTT istemcilerine iletir. Yayın, MQTT v3 iletisinin bilgi yüküdür. Bilgi yükü, ileti üstbilgilerinden ve `jms-bytes` biçiminde bir bayt dizisinden oluşur. Telemetri sunucusu, bir MQTT v3 iletisi ile bir IBM MQ iletisi arasındaki üstbilgileri eşler; bkz. “MQ Telemetry ' in kuyruk yöneticileriyle bütünleştirilmesi” sayfa 124.

IBM Integration Bus ve MQTT istemcileri arasında yayın göndermek ve almak için Yayın, MQInput ve JMSGiriş düğümlerini kullanın.

İleti akışlarını kullanarak telemetriyi HTTPkullanarak web siteleriyle ve IBM MQ ve WebSphere Adapters kullanan diğer uygulamalarla bütünleştirebilirsiniz.

Linux Windows AIX MQ Telemetry ' in kuyruk yöneticileriyle bütünleştirilmesi

MQTT istemcisi, bir yayınlama/abone olma uygulaması olarak IBM MQ ile bütünleştirilmiştir. IBM MQiçindeki konuları yayınlayabilir ya da bunlara abone olabilir, yeni konular oluşturabilir ya da var olan konuları kullanabilir. Kendisi de dahil olmak üzere MQTT istemcilerinin ya da abonelikleri konularına yayınlayan diğer IBM MQ uygulamalarının sonucu olarak IBM MQ ' den yayınlar alır. Bir yayının özniteliklerine karar vermek için kurallar uygulanır.

IBM MQtarafından sağlanan konularla, yayınlarla, aboneliklerle ve iletilerle ilişkili özniteliklerin çoğu desteklenmez. “MQTT Client to IBM MQ yayınlama/abone olma aracısı” sayfa 125 ve “IBM MQ bir MQTT istemcisine” sayfa 126 yayınların özniteliklerinin nasıl ayarlandığını açıklar. Ayarlar, yayının IBM MQ yayınlama/abone olma aracısına gidip gitmeyeceğine bağlıdır.

IBM MQ yayınlama/abone olma konularında, yönetim konusu nesneleriyle ilişkilendirilir. MQTT istemcileri tarafından yaratılan konular farklı değildir. Bir MQTT istemcisi bir yayın için konu dizgisi yarattığında, IBM MQ yayınlama/abone olma aracısı bunu bir denetim konusu nesnesiyle ilişkilendirir. Aracı, yayındaki konu dizgisini en yakın denetim konusu nesnesi üst öğesiyle eşler. Eşleme, IBM MQ uygulamalarıyla

aynıdır. Kullanıcı tarafından yaratılan bir konu yoksa, yayın konusu SYSTEM.BASE.TOPICile eşlenir. Yayına uygulanan öznitelikler, konu nesnesinden türetilir.

Bir IBM MQ uygulaması ya da bir yönetici bir abonelik oluşturduğunda, abonelik adı verilir. 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 yayınlama/abone olma aracısı

Bir MQTT istemcisi, IBM MQ' e bir yayın gönderdi. Telemetry (MQXR) hizmeti, yayını bir IBM MQ iletisine dönüştürür. IBM MQ iletisi üç bölüm içerir:

1. MQMD
2. RFH2
3. İleti

MQMD özellikleri, Çizelge 9 sayfa 125içinde belirtilenler dışında, varsayılan değerlerine ayarlanır.

Çizelge 9. MQMD özelliklerine ilişkin ayarlar

MQMD alanı	Tip	Değer
Format	MQCHAR8	MQFMT_RF_HEADER_2
UserIdentifier	MQCHAR12	Şunlardan 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 değeri 4 olan JMS ' den farklı olan IBM MQiçin varsayılan 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. Telemetry (MQXR) hizmeti, IBM MQ iletisini varsayılan JMS iletisi olarak oluşturur. Varsayılan JMS ileti-tipi bir jms-bytes iletisidir. Bir uygulama, ileti özellikleri olarak ek üstbilgi bilgilerine erişebilir; bkz. İleti özellikleri.

RFH2 değerleri, Çizelge 10 sayfa 125içinde gösterildiği gibi ayarlanır. Biçim özelliği, RFH2 sabit üstbilgisinde ayarlanır ve diğer değerler RFH2 klasörlerinde ayarlanır.

Çizelge 10. RFH2 özellikleri için ayarlar

RFH2 özelliği	Tip/Klasör	Üstbilgi
Biçim	MQCHAR8	MQFMT_NONE
ClientIdentifier	mqtt/clientId	MqttClient.ClientIdentifier değerini 1...23 bayt uzunluğuyla kopyalayın.
QoS	mqtt/qos	QoS ürününü gelen MQTT iletisinden kopyalayın.
İleti Tanıtıcısı	mqtt/msgid	QoS 1 ya da 2ise, gelen MQTT iletisinden İleti Tanıtıcısı ' nı kopyalayın.

Çizelge 10. RFH2 özellikleri için ayarlar (devamı var)		
RFH2 özelliği	**Tip/Klasör**	**Üstbilgi**
`MQIsRetained`	`mqps/Ret`	Özgün MQTT yayınının `RETAIN` özellik kümesiyle gönderilip gönderildiğini ve iletinin alıkonan bir yayın olarak alınacağını belirler.
`MQTopicString`	`mqps/Top`	MQTT iletisinin yayınlandığı konu.

MQTT yayınındaki bilgi yükü, bir IBM MQ iletisinin içeriğiyle eşlenir:

Çizelge 11. MQTT yayınındaki bilgi yükü, IBM MQ iletisinin içeriğiyle nasıl eşlenir		
İleti içeriği	**Tip**	**IBM MQ iletisinin içeriği**
`Arabellek`	`MQBYTE` *n*	Gelen MQTT iletisindeki baytların kopyası. Uzunluk sıfır olabilir.

IBM MQ bir MQTT istemcisine

Bir istemci bir yayın konusuna abone oldu. Bir IBM MQ uygulaması konuya yayınlandı ve bu, IBM MQ yayınlama/abone olma aracısı tarafından MQTT abonesine bir yayının gönderilmesiyle sonuçlandı. Alternatif olarak, bir IBM MQ uygulaması doğrudan MQTT istemcisine istenmeyen bir ileti gönderdi. Çizelge 12 sayfa 126 , MQTT istemcisine gönderilen iletide sabit ileti üstbilgilerinin nasıl ayarlandığını açıklar. IBM MQ ileti üstbilgisindeki ya da diğer üstbilgilerdeki diğer veriler atılır. IBM MQ iletisindeki ileti verileri, değişiklik olmadan MQTT iletisinde ileti bilgi yükü olarak gönderilir. MQTT iletisi, telemetri (MQXR) hizmeti tarafından MQTT istemcisine gönderilir.

Çizelge 12. MQTT istemcisine gönderilen bir IBM MQ iletisinde sabit ileti üstbilgilerinin nasıl ayarlandığı		
MQTT alan	**Tip**	**Değer**
DUP	`boole`	`QoS = 1` ya da 2iletisinin önceki bir iletimde bu istemciye gönderilip gönderilmediğini ve iletinin bir süre sonra alınmadığını belirleyin.

QoS	`int`	IBM MQ içinde yayınlama/abone olma aracısından giden bir yayında QoS değerinin ayarlanması, gelen yayına bağlıdır. Bu, gelen yayının bir MQTT istemcisinden mi, yoksa bir IBM MQ uygulamasından mı gönderildiğine bağlıdır.
		MQTT
		Gelen yayındaki ve abone tarafından istenen QoS 'deki QoS ' nin alt değeri.
		IBM MQ
		Gelen yayından türetilen QoS ' nin alt değeri:
		`MQPER_NOT_PERSISTENT→QoS=0`
		`MQPER_PERSISTENT→QoS=2`
		ve abone tarafından istenen QoS . İleti istemciye abonelik olmadan gönderilirse, QoS varsayılan olarak 2 değerine ayarlanır. Bir istemci, farklı bir QoSile `DEFAULT.QoS` ' e abone olarak bu değeri değiştirebilir.

RETAIN	`boole`	Gelen yayında alıkonan özellik ayarlıysa ayarlayın.

Çizelge 13 sayfa 127 , MQTT istemcisine gönderilen MQTT iletisinde değişken ileti üstbilgilerinin nasıl ayarlandığını açıklar.

Çizelge 13. MQTT istemcisine gönderilen bir MQTT iletisinde MQTT değişken üstbilgisi özelliklerinin nasıl ayarlandığı

MQTT alan	Tip	Değer
Topic name	Dizgi	İletinin yayınlandığı konu dizgisi.
Message ID	Dizgi	MQMD.MsgId , SYSTEM.MQTT.TRANSMIT.QUEUEiçine yerleştirildiğinde yayının özelliği.
Payload	bayt []	Gelen yayından yayınlama/abone olma aracısına byte 'ların doğrudan kopyası. Uzunluk sıfır olabilir.

Linux Windows AIX MQTT durumsuz ve durumlu oturumlar

MQTT istemcileri kuyruk yöneticisiyle durumlu bir oturum yaratabilir. Durumlu bir MQTT istemcisi bağlantıyı kestiğinde, kuyruk yöneticisi istemci tarafından oluşturulan abonelikleri ve devam eden iletileri korur. İstemci yeniden bağlandığında, hareket eden iletiyi çözer. Teslim için kuyruğa alınan iletileri gönderir ve bağlantısı kesildiğinde abonelikleri için yayınlanan iletileri alır.

MQTT istemcisi bir telemetri kanalına bağlandığında, yeni bir oturum başlatır ya da eski bir oturumu sürdürür. Yeni bir oturumda, onaylanmamış bekleyen ileti, abonelik ve teslim edilmeyi bekleyen yayın yok. Bir istemci bağlandığında, temiz bir oturumla mı, yoksa var olan bir oturumla mı başlayacağını belirler; bkz. Temiz oturumlar.

İstemci varolan bir oturumu sürdürür ve bağlantı kesilmemiş gibi devam eder. Teslim edilmeyi bekleyen yayınlar müşteriye gönderilir ve kesinleştirilmemiş ileti aktarımları tamamlanır. Kalıcı oturumdaki bir istemci telemetri (MQXR) hizmetiyle bağlantısını kestiğinde, istemci tarafından yaratılan abonelikler kalır. Aboneliklere ilişkin yayınlar, yeniden bağlandığında istemciye gönderilir. Eski oturuma devam etmeden yeniden bağlantı kurarsa, yayınlar telemetri (MQXR) hizmeti tarafından atılır.

Oturum durumu bilgileri, kuyruk yöneticisi tarafından SYSTEM.MQTT.PERSISTENT.STATE kuyruğunda saklanır.

IBM MQ yöneticisi, bir oturumun bağlantısını kesebilir ve oturumunu temizleyebilir.

Linux Windows AIX Bir MQTT istemcisi bağlı olmadığında

Bir istemci bağlanmadığında, kuyruk yöneticisi kendi adına yayınları almaya devam edebilir. Yeniden bağlandığında istemciye iletilir. İstemci beklenmedik bir şekilde bağlantıyı keserse, kuyruk yöneticisinin istemci adına yayınlayacağı bir "Son irade ve vasiyet"yaratabilir.

İstemci beklenmedik bir şekilde bağlantınızı kestiğinde size bildirim gönderilmesini istiyorsanız, son bir vasiyet ve vasiyet yayınını kaydedebilirsiniz; bkz. Son vasiyet ve vasiyet yayını. Telemetri (MQXR) hizmeti tarafından, istemci tarafından istenmeden istemciyle bağlantının kesildiğini saptarsa gönderilir.

Bir müşteri, alıkonan bir yayını istediği zaman yayınlayabilir; bkz. Alıkonan yayınlar ve MQTT istemcileri. Bir konuya ilişkin yeni bir abonelik, konuyla ilişkili alıkonan herhangi bir yayının gönderilmesini isteyebilir. Son vasiyeti ve vasiyeti alıkonan bir yayın olarak oluşturursanız, istemcinin durumunu izlemek için bunu kullanabilirsiniz.

Örneğin, istemci bağlandığında, kullanılabilirliğini duyurarak, alıkonan bir yayını yayınlar. Aynı zamanda, kullanılamamasını duyuran bir son vasiyet ve vasiyet yayını oluşturur. Buna ek olarak, planlı bir bağlantı kesilmeden hemen önce, onun kullanılamamasını korunan bir yayın olarak yayınlar. İstemcinin kullanılabilir olup olmadığını öğrenmek için, alıkonan yayının konusuna abone olmanız gerekir. Her zaman üç yayından birini alacaksınız.

İstemci bağlantısı kesildiğinde yayınlanan iletileri alacaksa, istemciyi önceki oturumuna yeniden bağlayın; bkz. “MQTT durumsuz ve durumlu oturumlar” sayfa 127. Abonelikleri silininceye kadar ya da istemci temiz bir oturum oluşturuncaya kadar etkindir.

Linux Windows AIX MQTT istemcileri ve IBM MQ uygulamaları arasındaki gevşek bağlaşım

MQTT istemcileri ve IBM MQ uygulamaları arasındaki yayınların akışı gevşek bir şekilde birbirine bağlanmıştır. Yayınlar MQTT istemcisinden ya da IBM MQ uygulamasından kaynaklanabilir ve herhangi bir sıralama düzeni belirlenmez. Yayıncılar ve aboneler gevşek bir şekilde birbirine bağlanmıştır. Yayınlar ve abonelikler aracılığıyla dolaylı olarak birbirleriyle etkileşim kurarlar. Bir IBM MQ uygulamasından doğrudan MQTT istemcisine ileti de gönderebilirsiniz.

MQTT istemcileri ve IBM MQ uygulamaları iki şekilde gevşek bir şekilde birbirine bağlanmıştır:

1. Yayıncılar ve aboneler, bir yayının ilişkilendirmesi ve bir konuyla ilgili bir abonelik ile gevşek bir şekilde birleştirilmiştir. Yayıncılar ve aboneler normalde bir yayının veya aboneliğin diğer kaynağının adresini veya kimliğini bilmezler.
2. MQTT istemcileri, ayrı iş parçacıklarında yayınlama, abone olma, yayınları alma ve teslim alma onaylarını işleme.

Bir MQTT istemci uygulaması, bir yayın teslim edilinceye kadar beklemez. Uygulama, MQTT istemcisine bir ileti iletir ve uygulama kendi iş parçacığında devam eder. Uygulamayı bir yayının teslimiyle eşitlemek için bir teslim simgesi kullanılır; bkz. Teslim belirteçleri.

MQTT istemcisine bir ileti iletildikten sonra, uygulama teslim belirtecini bekleme seçeneğine sahiptir. İstemci, beklemek yerine, yayın IBM MQ' e teslim edildiğinde çağrılan bir geri çağırma yöntemi sağlayabilir. Teslim simgesini de yoksayabilir.

İletiyle ilişkili hizmet kalitesine bağlı olarak, teslim belirteci hemen geri çağırma yöntemine ya da büyük olasılıkla önemli bir süreden sonra döndürülür. Teslim simgesi, istemci bağlantısı kesildikten ve yeniden bağlandıktan sonra bile döndürülebilir. Hizmet kalitesi *yangın ve unutun*ise, teslim belirteci hemen döndürülür. Diğer iki durumda, teslim belirteci yalnızca müşteri, yayının abonelere gönderildiğine dair bir bildirim aldığında döndürülür.

Bir istemci aboneliğinin sonucu olarak MQTT istemcisine gönderilen yayınlar, `messageArrived` geri çağırma yöntemine teslim edilir. `messageArrived` , ana uygulamadan farklı bir iş parçacığında çalışır.

İletilerin MQTT istemcisine doğrudan gönderilmesi

Belirli bir MQTT istemcisine iki yoldan biriyle ileti gönderebilirsiniz.

1. IBM MQ uygulaması, aboneliği olmayan bir MQTT istemcisine doğrudan ileti gönderebilir; bkz. İstemciye doğrudan ileti gönderilmesi.
2. Diğer bir yaklaşım, `ClientIdentifier` adlandırma kuralınızı kullanmaktır. Tüm MQTT abonelerinin konu olarak benzersiz `ClientIdentifier` ' larını kullanarak abonelikler oluşturmalarını sağlayın. *ClientIdentifier*' a 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 dikkatle kontrol edilemeyen yerlerde kullanıldığı için telemetri cihazlarının güvenliğinin sağlanması önemli olabilir. MQTT aygıtından telemetri (MQXR) hizmetine bağlantıyı güvenli kılmak için VPN ' yi kullanabilirsiniz. MQ Telemetry , diğer iki güvenlik mekanizmasını sağlar: TLS ve JAAS.

TLS, esas olarak aygıt ile telemetri kanalı arasındaki iletişimi şifrelemek ve aygıtın doğru sunucuya bağlandığını doğrulamak için kullanılır; bkz. TLS kullanarak Telemetry channel authentication. İstemci aygıtının sunucuya bağlanmasına izin verilip verilmediğini denetlemek için TLS ' yi de kullanabilirsiniz; bkz. MQTT TLS kullanarak istemci kimlik doğrulaması.

JAAS birincil olarak, aygıt kullanıcısının bir sunucu uygulamasını kullanma izni olup olmadığını denetlemek için kullanılır; bkz. MQTT parola kullanarak istemci kimlik doğrulaması. JAAS , LDAP ile tek oturum açma dizini kullanarak parolayı denetlemek için kullanılabilir.

TLS ve JAAS , iki faktörlü kimlik doğrulaması sağlamak için birlikte kullanılabilir. TLS tarafından kullanılan şifrelemeleri, FIPS standartlarını karşılayan şifrelemelerle sınırlandırabilirsiniz.

En az on binlerce kullanıcı ile, bireysel güvenlik profilleri sağlamak her zaman pratik değildir. Ayrıca, tek tek kullanıcılara IBM MQ nesnelerine erişim yetkisi vermek için profilleri kullanmak her zaman kullanışlıdır. Bunun yerine, yayın ve konulara abonelik yetkisi vermek ve yayınları istemcilere göndermek için kullanıcıları sınıflar halinde gruplayın.

İstemcileri ortak istemci kullanıcı kimlikleriyle eşlemek için her telemetri kanalını yapılandırın. Belirli bir kanalda bağlanan her istemci için ortak bir kullanıcı kimliği kullanın; bkz. MQTT istemci kimliği ve yetkilendirmesi.

Kullanıcı gruplarının yetkilendirilmesi, her bir kişinin kimlik doğrulamasını tehlikeye atmaz. Her bir kullanıcının kimliği istemcide ya da sunucuda `Kullanıcı Adı` ve `Parola`ile doğrulanabilir ve daha sonra, ortak bir kullanıcı kimliği kullanılarak sunucuda yetkilendirilir.

Linux Windows AIX MQ Telemetry küreselleşme

MQTT v3 iletişim kuralındaki ileti bilgi yükü bayt dizisi olarak kodlanır. Genellikle, metin işleyen uygulamalar UTF-8içinde ileti bilgi yükünü oluşturur. Telemetri kanalı, ileti bilgi yükünü UTF-8olarak tanımlar, ancak kod sayfası dönüştürmeleri yapmaz. Yayın konusu dizesi UTF-8olmalıdır.

Uygulama, alfabetik verileri doğru kod sayfasına ve sayısal verileri doğru sayı kodlamasına dönüştürmekten sorumludur.

MQTT Java istemcisi uygun bir `MqttMessage.toString` yöntemine sahiptir. Yöntem, ileti bilgi yükünü genellikle UTF-8olan yerel altyapı varsayılan karakter kümesinde kodlanmış olarak işler. Bilgi yükünü bir Java Dizgisine dönüştürür. Java , bir dizgiyi yerel altyapı varsayılan karakter takımı kullanılarak kodlanmış bir bayt dizisine dönüştüren bir String (Dizilim) yöntemine sahiptir (`getBytes`). Aynı varsayılan karakter kümesine sahip platformlar arasında, ileti bilgi yükünde metin değiş tokuş eden iki MQTT Java programı, UTF-8' de bunu kolay ve verimli bir şekilde yapar.

Platformlardan birinin varsayılan karakter kümesi UTF-8değilse, uygulamaların ileti değiş tokuşuna ilişkin bir kural oluşturması gerekir. Örneğin, yayınlayıcı `getBytes("UTF8")` yöntemini kullanarak bir dizgiden UTF-8 dizgisine dönüştürmeyi belirtir. Bir iletinin metnini almak için abone, iletinin UTF-8 karakter kümesinde kodlandığını varsayar.

Telemetry (MQXR) hizmeti, MQTT istemcisi iletilerinden gelen tüm yayınların kodlamasını UTF-8olarak tanımlar. `MQMD.CodedCharSetId` to UTF-8ve `RFH2.CodedCharSetId` to `MQCCSI_INHERIT` ; bkz. “MQ Telemetry ' in kuyruk yöneticileriyle bütünleştirilmesi” sayfa 124. Yayının biçimi `MQFMT_NONE`olarak ayarlandığı için, kanallar ya da `MQGET`tarafından dönüştürme gerçekleştirilemez.

Linux Windows AIX MQ Telemetry performansı ve ölçeklenebilirliği

Çok sayıda müşteri yönetirken ve MQ Telemetryölçeklenebilirliğini geliştirirken aşağıdaki etkenleri göz önünde bulundurun.

Kapasite planlama

MQ Telemetryperformans raporları hakkında bilgi için bkz. MQ Performans belgeleri.

Bağlantılar

Bağlantılarla ilgili maliyetler arasında şunlar yer alır:

- Bir bağlantının kendisini işlemci kullanımı ve süresi açısından ayarlama maliyeti.
- Ağ maliyetleri.
- Bir bağlantıyı açık tutarken kullanılan, ancak kullanmayan bellek.

Müşteriler bağlı kaldığında fazladan bir yük ortaya çıktı. Bir bağlantı açık tutulursa, TCP/IP akışları ve MQTT iletileri, bağlantının hala var olup olmadığını denetlemek için ağı kullanır. Ayrıca, sunucuda açık tutulan her istemci bağlantısı için bellek kullanılır.

Dakikada birden fazla ileti gönderiyorsanız, yeni bir bağlantı başlatma maliyetini önlemek için bağlantınızı açık tutun. Her 10-15 dakikada bir birden az ileti gönderiyorsanız, açık tutma maliyetini önlemek için bağlantınızı bırakmayı düşünebilirsiniz. TLS bağlantısını daha uzun süre açık, ancak boşta tutmak isteyebilirsiniz; bunun nedeni, TLS bağlantısını kurmanın daha pahalı olmasıdır.

Ayrıca, istemcinin yeteneğini de göz önünde bulundurun. İstemcide bir saklama ve iletme olanağı varsa, iletileri toplu olarak toplayıp toplu işler arasındaki bağlantıyı bırakabilirsiniz. Ancak, istemcinin bağlantısı kesilirse, istemcinin sunucudan ileti alması mümkün değildir. Bu nedenle başvurunuzun amacı karar üzerinde bir durmaya sahiptir.

Sisteminizde birden çok ileti gönderen bir istemci varsa (örneğin, dosya aktarımları), ileti başına sunucu yanıtı bekleme. Bunun yerine, tüm iletileri gönderin ve sonunda bunların alındığını kontrol edin. Alternatif olarak, Hizmet Kalitesi ' ni (QoS) kullanın.

QoS ' i iletiye göre değiştirip QoS 0 ve önemli iletileri 2 'lik QoS kullanarak iletebilirsiniz. İleti verimi, QoS değeri 0 ile, QoS değeri 2 'den iki kat daha yüksek olabilir.

Adlandırma kuralları

Uygulamanızı birçok istemci için tasarlıyorsanız, etkili bir adlandırma kuralı uygulayın. Her bir istemciyi doğru `ClientIdentifier`ile eşlemek için `ClientIdentifier` değerini anlamlı kılın. İyi bir adlandırma kuralı, Yöneticinin hangi istemcilerin çalıştığını anlamasını kolaylaştırır. Adlandırma kuralı, yöneticinin IBM MQ Explorer 'da uzun bir istemci listesini süzmesine yardımcı olur ve sorun belirlemeye yardımcı olur; bkz. İstemci tanıtıcısı.

Verim

Konu adlarının uzunluğu, ağ üzerinden akan bayt sayısını etkiler. Yayınlama ya da abone olma sırasında, bir iletideki bayt sayısı önemli olabilir. Bu nedenle, konu adındaki karakter sayısını sınırlayın. Bir MQTT istemcisi bir konuya abone olduğunda IBM MQ , ona formun adını verir:

```
ClientIdentifier: TopicName
```

Bir MQTT istemcisine ilişkin tüm abonelikleri görüntülemek için IBM MQ MQSC **DISPLAY** komutunu kullanabilirsiniz:

```
DISPLAY SUB(' ClientID1:*')
```

MQTT istemcileri tarafından kullanılmak üzere IBM MQ içindeki kaynakları tanımlama

MQTT istemcisi, bir IBM MQ uzak kuyruk yöneticisine bağlanır. Bir IBM MQ uygulamasının MQTT istemcisine ileti göndermesi için iki temel yöntem vardır: Varsayılan iletim kuyruğunu `SYSTEM.MQTT.TRANSMIT.QUEUE` olarak ayarlayın ya da kuyruk yöneticisi diğer adlarını kullanın. Çok sayıda MQTT istemcisi varsa, bir kuyruk yöneticisinin varsayılan iletim kuyruğunu tanımlayın. Varsayılan iletim kuyruğu ayarının kullanılması, yönetim çalışmasını kolaylaştırır; bkz. Dağıtılmış kuyruğa alma özelliğinin MQTT istemcilere ileti gönderecek şekilde yapılandırılması.

Aboneliklerden kaçınarak ölçeklenebilirliği geliştirme.

Bir MQTT V3 istemcisi bir konuya abone olduğunda, IBM MQiçinde telemetry (MQXR) hizmeti tarafından bir abonelik oluşturulur. Abonelik, istemciye ilişkin yayınları `SYSTEM.MQTT.TRANSMIT.QUEUE`' e yönlendirir. Her yayının iletim üstbilgisindeki uzak kuyruk yöneticisi adı, aboneliği yapan MQTT istemcisinin `ClientIdentifier` değerine ayarlanır. Her biri kendi aboneliklerini yapan çok sayıda

istemci varsa, bu, IBM MQ yayınlama/abone olma kümesi ya da sıradüzeni boyunca birçok yetkili sunucu aboneliğine neden olur. Yayınlama/abone olma yerine, noktadan noktaya iletişim tabanlı bir çözüm kullanma hakkında bilgi için bkz. İstemciye doğrudan ileti gönderme.

Çok sayıda istemcinin yönetilmesi

Eşzamanlı olarak bağlı birçok istemcinin desteklenmesi için, **-Xms** ve **-Xmx**JVM değiştirgelerini ayarlayarak telemetri (MQXR) hizmeti için kullanılabilir bellek miktarını artırın. Aşağıdaki adımları izleyin:

1. Telemetri hizmeti yapılandırma dizininde `java.properties` dosyasını bulun; bkz. Telemetry (MQXR) service configuration directory on Windows ya da Telemetry service configuration directory on Linux .
2. Dosyadaki yönergeleri izleyin; eşzamanlı bağlı 50.000 istemci için 1 GB yığın yeterlidir.

```
# Heap sizing options - uncomment the following lines to set the heap to 1G
#-Xmx1024m
#-Xms1024m
```

3. `java.properties` dosyasında telemetri (MQXR) hizmetini çalıştıran JVM ' ye geçirmek için diğer komut satırı bağımsız değişkenlerini ekleyin; bkz. JVM değiştirgelerinin telemetri (MQXR) hizmetine yapıştırılması.

Linuxüzerinde açık dosya tanımlayıcılarının sayısını artırmak için `/etc/security/limits.conf/'` e aşağıdaki satırları ekleyin ve yeniden oturum açın.

```
@mqm soft nofile 65000
@mqm hard nofile 65000
```

Her yuva için bir dosya tanımlayıcısı gerekir. Telemetri hizmeti bazı ek dosya tanımlayıcıları gerektirdiğinden bu sayı, gerekli açık yuva sayısından daha büyük olmalıdır.

Kuyruk yöneticisi, sürekli olmayan her abonelik için bir nesne tanıtıcısı kullanır. Birçok etkin, sürekli olmayan aboneliği desteklemek için, kuyruk yöneticisindeki etkin tanıtıcı sayısı üst sınırını artırır; örneğin:

```
echo ALTER QMGR MAXHANDS(999999999) | runmqsc qMgrName
```

Şekil 48. Windows üzerinde tanıtıcı sayısı üst sınırını değiştirme

```
echo "ALTER QMGR MAXHANDS(999999999)" | runmqsc qMgrName
```

Şekil 49. Linux üzerinde tanıtıcı sayısı üst sınırını değiştirme

Diğer önemli noktalar

Sistem gereksinimlerinizi planlarken, sistemi yeniden başlatmak için geçen süreyi göz önünde bulundurun. Planlanan kapalı kalma süresi, kuyruğa alınan ve işlenmeyi bekleyen iletilerin sayısı üzerinde etkileri olabilir. İletilerin kabul edilebilir bir süre içinde başarıyla işlenebilmesi için sistemi yapılandırın. Disk depolama, bellek ve işleme gücünü gözden geçirin. Bazı istemci uygulamalarıyla, istemci yeniden bağlandığında iletiler atılabilir. İletileri atmak için istemci bağlantısı parametrelerinde `CleanSession` ayarını yapın; bkz. Temiz oturumlar. Alternatif olarak, bir MQTT istemcisinde `0`en iyi hizmet kalitesini kullanarak yayınlayın ve abone olun; bkz. Hizmet kalitesi. IBM MQ' den ileti gönderirken `kalıcı olmayan` iletileri kullanın. Bu hizmet niteliklerine sahip iletiler, sistem ya da bağlantı yeniden başlatıldığında kurtarılamaz.

Linux Windows AIX MQ Telemetry tarafından desteklenen aygıtlar

MQTT istemcileri, sensörlerden ve aktüatörlerden elde tutulan aygıtlara ve araç sistemlerine kadar çeşitli aygıtlarda çalışabilirler.

MQTT istemcileri küçüktür ve az bellek ve düşük işlem gücüyle kısıtlanan aygıtlarda çalışır. MQTT protocol güvenilirdir ve düşük bant genişliği, yüksek maliyet ve aralıklı kullanılabilirlik ile kısıtlanan ağlara uygun küçük üstbilgilere sahiptir.

MQ Telemetry , MQTT istemci uygulamaları aracılığıyla telemetri aygıtlarıyla iletişim kurar. Bu uygulamalar, MQTT v3 iletişim kuralını uygulayan aşağıdaki kaynakları kullanır:

- Aşağıdaki istemci kitaplıkları:
  - Android, OS X Linux ya da Windows aygıtları için yerel uygulamalar oluşturmak üzere kullanılan *MQTT client for Java*. Bu istemci kitaplığını kullanan uygulamalar, en küçük CLDC (Bağlı Sınırlı Aygıt Yapılandırması) /MIDP (Mobil Bilgi Aygıtı Profili) ile CDC (Bağlı Aygıt Yapılandırması) /Foundation, J2SE (Java Platform, Standard Edition) ve J2EE (Java Platform, Enterprise Edition) arasındaki tüm Java çeşitlemelerinde çalışabilir. IBM jclRM özelleştirilmiş sınıf kitaplığı da desteklenir. Java ME platformu genellikle çalıştırıcılar, algılayıcılar, cep telefonları ve diğer yerleşik aygıtlar gibi küçük aygıtlarda kullanılır. Java SE platformu genellikle masaüstü bilgisayarlar ve sunucular gibi daha üst uç yerleşik aygıtlara kurulur.
  - iOS, OS X Linux ya da Windows aygıtları için yerel uygulamalar oluşturmak üzere kullanılan *MQTT client for C*. Bu istemci kitaplığı, Windows ve Linux sistemleri için önceden oluşturulmuş yerel istemciyle birlikte bir C başvuru uygulaması sağlar. C referans uygulaması, MQTT ' in çok çeşitli aygıtlara ve platformlara taşınmasını sağlar. Windows 7, RedHat, Ubuntuve Eurotech Viper gibi ARM platformlarındaki bazı Linux sistemleri de dahil olmak üzere Intelüzerindeki bazı Windows sistemleri, C istemcisini çalıştıran Linux sürümlerini uygular, ancak IBM , platformlar için hizmet desteği sağlamaz. IBM destek merkezinizi aramayı düşünüyorsanız, desteklenen bir platformda istemciyle ilgili sorunları yeniden oluşturmanız gerekir.
  - Tarayıcı tabanlı web uygulamaları oluşturmak için kullanılan *MQTT client for Java*.

MQTT istemci kitaplıklarını Eclipse Paho ve MQTT.orgürünlerinden ücretsiz olarak edinebilirsiniz. Bkz. IBM MQ Telemetry Transport örnek programları.

IBM MQ içinde güvenlik

IBM MQiçinde, güvenlik sağlamak için çeşitli yöntemler vardır: yetkilendirme hizmeti arabirimi; kullanıcı tarafından yazılan ya da üçüncü kişi, kanal çıkışları; Aktarım Katmanı Güvenliği (TLS) ile kanal güvenliği, kanal kimlik doğrulama kayıtları ve ileti güvenliği.

Yetkilendirme hizmeti arabirimi

MQI çağrılarını, komutlarını ve nesnelere erişimi kullanma yetkisi, varsayılan olarak etkinleştirilen **nesne yetki yöneticisi** (OAM) tarafından sağlanır. IBM MQ varlıklarına 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 yaratılmasıyla ilgili ek bilgi için AIX, Linux, and Windows sistemlerinde güvenliğin ayarlanmasıbaşlıklı konuya bakın.

Kullanıcı tarafından yazılan ya da üçüncü kişi kanal çıkışları

Kanallar, kullanıcı tarafından yazılan veya üçüncü kişi kanal çıkışlarını kullanabilir. Daha fazla bilgi için bkz. İleti kanalları için kanal çıkış programları.

TLS kullanarak kanal güvenliği

TLS (Transport Layer Security; Taşıma Katmanı Güvenliği) protokolü, gizlice dinleme, kurcalama ve taklit edilmeye karşı koruma ile sektör standardında kanal güvenliği sağlar.

TLS, mesaj gizliliği, bütünlüğü ve karşılıklı kimlik doğrulaması sağlamak için açık anahtar ve simetrik teknikler kullanır.

TLS ile ilgili ayrıntılı bilgiler de dahil olmak üzere IBM MQ içindeki güvenliğin kapsamlı bir gözden geçirilmesi için bkz. Securing. Bu bölümde açıklanan komutlara ilişkin işaretçiler de dahil olmak üzere TLS ' ye genel bakış için bkz. Şifreleme güvenliği iletişim kuralları: TLS.

Kanal kimlik doğrulama kayıtları

Bir kanal düzeyinde bağlanan sistemlere verilen erişim üzerinde kesin denetim sağlamak için kanal kimlik doğrulama kayıtlarını kullanın. Daha fazla bilgi için bkz. Kanal kimlik doğrulama kayıtları.

İleti güvenliği

IBM MQkullanılarak gönderilen ve alınan iletilere şifreleme koruması sağlamak için IBM MQ'in ayrı olarak kurulan ve lisanslı bir bileşeni olan Advanced Message Security' i kullanın. Bkz. Advanced Message Security.

İlgili görevler

güvenlik

Güvenlik gereksinimlerinizin planlanması

IBM MQ.NET yönetilen istemci TLS desteği

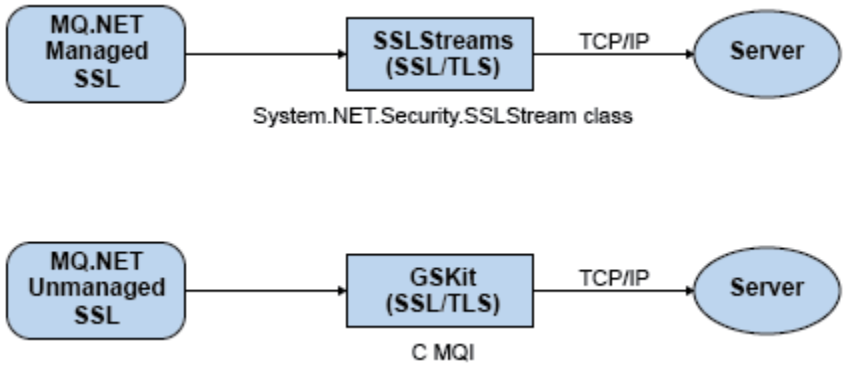
IBM MQ.NET tam olarak yönetilen istemcisi, Microsoft.NET SSLStreams setine dayalı Transport Layer Security (TLS) desteği sağlar. Bu, IBM Global Security Kit (GSKit)tabanlı diğer IBM MQ istemcilerinden farklıdır.

IBM MQ.NET uygulamalarını yönetilen kipte ya da yönetilmeyen kipte çalışacak şekilde geliştirebilirsiniz.

- Yönetilen kipte .NET uygulamaları, C MQI 'ı çağırma gibi platformlar arası bir çağırma olmadan .NET CLR (Common Language Runtime) içinde çalışır.
- Yönetilmeyen kipte, temel MQI işlemleri için C MQI çağrılır. Temel olarak, yönetilmeyen kip arabirimi C MQI 'ın üstündeki .NET sarıcı sınıflarını içerir.

Yönetilen IBM MQ.NET istemcisi, TLS güvenli yuva iletişim kurallarını uygulamak için Microsoft.NET Framework kitaplıklarını kullanır. Microsoft içindeki Sistem.NET.Security.SSLStream sınıfı, IBM MQ.NETiçinde Güvenlik (TLS) uygulamak için kullanılır.

Yönetilmeyen IBM MQ.NET istemci kipi, C MQI (ve GSKit) tabanlı TLS özelliğini zaten destekler. Yani, TLS işlemleri C MQI tarafından işlenir. Bu durumda GSKit , TLS güvenli yuva iletişim kurallarını uygular.



Şekil 50. IBM MQ.NET yönetilen ve yönetilmeyen TLS karşılaştırması

Aşağıdaki tablo, yönetilen ve yönetilmeyen uygulamalar arasındaki farkları özetler:

Çizelge 14. Yönetilen ve yönetilmeyen uygulamalar arasındaki farklar			
Mod	**İletişim kuralları**	**Uygulama**	**Yorumlar**
IBM MQ.NET tarafından yönetilen SSL	TLS	Sistem.NET.Security.SSL Stream sınıfı SSLStream sınıfı, bağlı bir TCP yuvası üzerinden akış olarak çalışır	TLS 1.0 TLS 1.2 (yalnızca Microsoft.NET Framework v4.5 ile)
IBM MQ.NET yönetilmeyen SSL	TLS	GSKIT ve C-MQI	TLS güvenli yuva iletişim kuralları

İlgili kavramlar

.NET için Güvenli Yuva Katmanı (SSL) ve İletim Katmanı Güvenliği (TLS) desteği

IBM MQ MQI clients

IBM MQ MQI client , IBM MQ ürününün hiçbir kuyruk yöneticisinin çalışmadığı bir sisteme kurulabilen bir bileşenidir.

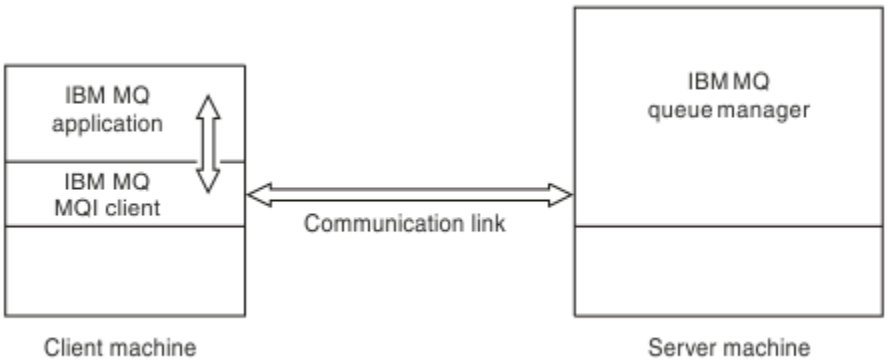
IBM MQ MQI *istemcisi* , sistemde çalışan bir uygulamanın başka bir sistemde çalışan bir kuyruk yöneticisine MQI çağrıları yapmasını sağlayan bir bileşendir. Aramanın çıkışı, istemciye geri gönderilir ve istemciyi uygulamaya geri iletir.

İstemciyle aynı sistemde çalışan bir uygulama, IBM MQ MQI clientkullanarak başka bir sistemde çalışan bir kuyruk yöneticisine bağlanabilir. Uygulama, o kuyruk yöneticisine MQI çağrıları verebilir. Bu tür bir uygulamaya IBM MQ MQI client uygulaması adı verilir ve kuyruk yöneticisine *sunucu kuyruk yöneticisi*adı verilir.

IBM MQ *sunucusu* , bir ya da daha çok istemciye kuyruk hizmetleri sağlayan bir kuyruk yöneticisidir. Kuyruklar gibi tüm IBM MQ nesneleri, istemcide değil, yalnızca kuyruk yöneticisi makinesinde (IBM MQ sunucu makinesi) bulunur. IBM MQ sunucusu, yerel IBM MQ uygulamalarını da destekleyebilir.

IBM MQ sunucusu ile olağan kuyruk yöneticisi arasındaki fark, bir sunucunun her istemciyle özel olarak ayrılmış bir iletişim bağlantısının bulunmasıdır. İstemciler ve sunucular için kanal yaratılmasıyla ilgili ek bilgi için Dağıtılmış kuyruğa alma özelliğini yapılandırmabaşlıklı konuya bakın.

Bir IBM MQ MQI client uygulaması ve bir sunucu kuyruk yöneticisi, *MQI kanalı*kullanarak birbirleriyle iletişim kurar. İstemci uygulaması kuyruk yöneticisine bağlanmak için bir **MQCONN** ya da **MQCONNX** çağrısı yayınladığında ve istemci uygulaması kuyruk yöneticisiyle bağlantısını kesmek için bir **MQDISC** çağrısı yayınladığında MQI kanalı başlar. MQI kanalında bir MQI çağrısı akışının giriş parametreleri ve çıkış parametreleri ters yönde akıyor.



Şekil 51. İstemci ile sunucu arasındaki bağlantı

Aşağıdaki platformlar kullanılabilir. Birleşimler, kullanmakta olduğunuz IBM MQ ürününe bağlıdır ve “IBM MQ istemcileri için platform desteği” sayfa 136içinde açıklanır.

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 tarafından kullanılabilir; kuyruklar ve diğer IBM MQ nesneleri, bir sunucuya kurduğunuz kuyruk yöneticisinde tutulur.

IBM MQ MQI client ortamında çalıştırmak istediğiniz bir uygulama öncelikle ilgili istemci kitaplığına bağlanmalıdır. Uygulama bir MQI çağrısı yayınladığında, IBM MQ MQI client isteği bir kuyruk yöneticisine yönlendirir; burada işlenir ve IBM MQ MQI client' e yanıt geri gönderilir.

Uygulama ile IBM MQ MQI client arasındaki bağlantı, çalıştırma zamanında dinamik olarak oluşturulur.

İstemci uygulamalarını IBM MQ classes for .NET, IBM MQ classes for Java ya da IBM MQ classes for Java Message Service (JMS) kullanarak da geliştirebilirsiniz. Java ve JMS istemcilerini aşağıdaki platformlarda kullanabilirsiniz:

- IBM i
- AIX
- Linux
- Windows

Java ve JMS kullanımı burada açıklanmaz. Kuruluş, yapılandırma ve kullanma hakkında tam ayrıntılar için IBM MQ classes for Java ve IBM MQ classes for JMS başlıklı konuya bakın IBM MQ classes for Java ' yi kullanma ve IBM MQ classes for JMS ' yi kullanma.

İstemci-sunucu ortamındaki IBM MQ uygulamaları

Bir sunucuya bağlandığında, istemci IBM MQ uygulamaları çoğu MQI çağrılarını yerel uygulamalarla aynı şekilde verebilir. 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 uygulama oluşturma.

İlgili kavramlar

“Neden IBM MQ istemcileri kullanılıyor?” sayfa 136
IBM MQ istemcilerinin kullanılması, IBM MQ ileti alışverişi ve kuyruklama işlemini gerçekleştirmenin verimli bir yoludur.

“Genişletilmiş işlem istemcisi nedir?” sayfa 137
IBM MQ genişletilmiş hareket istemcisi, dış hareket yöneticisinin denetimi altında başka bir kaynak yöneticisi tarafından yönetilen kaynakları güncelleyebilir.

“İstemcinin sunucuya bağlanma şekli” sayfa 138
Bir istemci MQCONN ya da MQCONNX kullanarak bir sunucuya bağlanır ve bir kanal aracılığıyla iletişim kurar.

“İşlem yönetimi ve desteği” sayfa 140
İşlem yönetimine ve IBM MQ ' ın işlemleri nasıl desteklediğine ilişkin giriş.

“Kuyruk yöneticisi olanaklarını genişletme” sayfa 141

Kullanıcı çıkışlarını, API çıkışlarını ya da kurulabilir hizmetleri kullanarak kuyruk yöneticisi olanaklarını genişletebilirsiniz.

İlgili bilgiler

IBM MQ MQI client nasıl ayarlanır

Neden IBM MQ istemcileri kullanılıyor?

IBM MQ istemcilerinin kullanılması, IBM MQ ileti alışverişi ve kuyruklama işlemini gerçekleştirmenin verimli bir yoludur.

Bir makinede çalıştırılan MQI ' ı ve farklı bir makinede (fiziksel ya da sanal) çalışan kuyruk yöneticisini kullanan bir uygulamanız olabilir. Bunu yapmanın yararları şunlardır:

- İstemci makinede tam bir IBM MQ uygulamasına gerek yoktur.
- Müşteri sistemindeki donanım gereksinimleri azaltılır.
- Sistem yönetimi gereksinimleri azaltılır.
- Bir istemcide çalışan bir IBM MQ uygulaması, farklı sistemlerde birden çok kuyruk yöneticisine bağlanabilir.
- Farklı iletim protokolleri kullanan alternatif kanallar kullanılabilir.

IBM MQ istemcileri için platform desteği

Desteklenen tüm sunucu platformlarında IBM MQ , çeşitli platformlarda IBM MQ MQI clients ' dan gelen istemci bağlantılarını kabul eder.

Desteklenen tüm sunucu platformlarında *Temel ürün ve Sunucu* olarak kurulan IBM MQ , aşağıdaki platformlarda IBM MQ MQI clients ürününden gelen bağlantıları kabul edebilir:

- IBM i
- AIX
- Linux
- Windows

İstemci bağlantıları, kodlanmış karakter takımı tanıtıcısı (CCSID) ve iletişim protokolündeki farklılıklara tabidir.

Bir IBM MQ MQI clientüzerinde hangi uygulamalar çalışır?

Tam MQI istemci ortamında desteklenir. Bu, hemen hemen her IBM MQ uygulamasının, IBM MQ MQI client üzerindeki uygulamayı MQI kitaplığı yerine MQIC kitaplığına bağlayarak bir IBM MQ MQI client sisteminde çalışacak şekilde yapılandırılmasını sağlar. Kural dışı durumlar şunlardır:

- İşaretimle MQGET
- Diğer kaynak yöneticileriyle eşitleme noktası eşgüdümü gerektiren bir uygulama, genişletilmiş bir işlem istemcisi kullanmalıdır

Önden okuma etkinleştirilirse, kalıcı olmayan ileti sistemi performansını artırmak için tüm MQGET seçenekleri kullanılamaz. Çizelge, izin verilen seçenekleri ve bunların MQGET çağrıları arasında değiştirilip değiştirilemeyeceğini gösterir.

Çizelge 15. Önden okuma etkinleştirildiğinde MQGET seçeneklerine izin verilir

Değerler	Önden okuma etkinleştirildiğinde izin verilir ve MQGET çağrıları arasında değiştirilebilir	Önden okuma etkinleştirildiğinde izin verilir, ancak MQGET çağrıları arasında değiştirilemez [1]	Önden okuma etkinleştirildiğinde izin verilmeyen MQGET seçenekleri [2]
MQGET MD değerleri	MsgId [3] CorrelId [3]	Kodlama CodedCharSetId	

Çizelge 15. Önden okuma etkinleştirildiğinde MQGET seçeneklerine izin verilir (devamı var)			
Değerler	**Önden okuma etkinleştirildiğinde izin verilir ve MQGET çağrıları arasında değiştirilebilir**	**Önden okuma etkinleştirildiğinde izin verilir, ancak MQGET çağrıları arasında değiştirilemez [1]**	**Önden okuma etkinleştirildiğinde izin verilmeyen MQGET seçenekleri [2]**
MQGET MQGMO seçenekleri	MQGMO_WAIT MQGMO_NO_WAIT MQGMO_FAIL_IF_QUIESCING MQGMO_BROWSE_FIRST [4] MQGMO_BROWSE_NEXT [4] MQGMO_BROWSE_MESSAGE_UNDER_CURSOR [4]	MQGMO_SYNCPOINT_IF_PERSISTENT MQGMO_NO_SYNCPOINT MQGMO_ACCEPT_TRUNCATED_MSG MQGMO_CONVERT MQGMO_LOGICAL_ORDER MQGMO_COMPLETE_MSG MQGMO_ALL_MSGS_VAR MQGMO_ALL_SEGMENTS_VAR MQGMO_MARK_BROWSE_HANDLE MQGMO_MARK_BROWSE_CO_OP MQGMO_UNMARK_BROWSE_CO_OP MQGMO_UNMARK_BROWSE_HANDLE MQGMO_UNMARKED_BROWSE_MSG MQGMO_PROPERTIES_FORCE_MQRFH2 MQGMO_NO_PROPERTIES MQGMO_PROPERTIES_IN_HANDLE MQGMO_PROPERTIES_COMPATIBILITY	MQGMO_SET_SIGNAL MQGMO_SYNCPOINT MQGMO_MARK_SKIP_KAPATıLMA MQGMO_MSG_ALTıNDA_CURSOR [4] MQGMO_KİLİK 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.

IBM MQ MQI client üzerinde çalışan bir uygulama, birden çok kuyruk yöneticisine eşzamanlı olarak bağlanabilir ya da bir MQCONN ya da MQCONNX çağrısında yıldız işareti (*) ile bir kuyruk yöneticisi adını kullanabilir (IBM MQ MQI client uygulamalarının kuyruk yöneticilerine bağlanması başlıklı konudaki örneklere bakın).

Genişletilmiş işlem istemcisi nedir?

IBM MQ genişletilmiş hareket istemcisi, dış hareket yöneticisinin denetimi altında başka bir kaynak yöneticisi tarafından yönetilen kaynakları güncelleyebilir.

Hareket yönetimi kavramlarına aşina değilseniz, bkz. “İşlem yönetimi ve desteği” sayfa 140.

XA işlemsel istemcisinin artık IBM MQ' un bir parçası olarak sağlandığını unutmayın.

Bir istemci uygulaması, bağlı olduğu bir kuyruk yöneticisi tarafından yönetilen bir iş birimine katılabilir. İş birimi içinde, istemci uygulaması iletileri o kuyruk yöneticisinin sahip olduğu kuyruklara koyabilir ve bu kuyruklardan ileti alabilir. İstemci uygulaması, iş birimini kesinleştirmek için **MQCMIT** çağrıyı ya da iş birimini geri almak için **MQBACK** çağrıyı kullanabilir. Ancak, aynı iş birimi içinde, istemci uygulaması başka bir kaynak yöneticisinin (örneğin, bir Db2 veritabanının çizelgeleri) kaynaklarını güncelleyemez. IBM MQ genişletilmiş hareket istemcisi kullanıldığında bu sınırlama kaldırılır.

IBM MQ genişletilmiş hareket istemcisi, bazı ek işlevlere sahip bir IBM MQ MQI client istemcisidir. Bu işlevi kullanarak, bir istemci uygulaması aynı iş birimi içinde aşağıdaki görevleri gerçekleştirebilir:

- İletileri, bağlı olduğu kuyruk yöneticisine ait kuyruklara koy ve bu kuyruklardan ileti al
- IBM MQ kuyruk yöneticisinden başka bir kaynak yöneticisinin kaynaklarını güncelle

Bu iş birimi, istemci uygulamasıyla aynı sistemde çalışan bir dış hareket yöneticisi tarafından yönetilmelidir. İş birimi, istemci uygulamasının bağlı olduğu kuyruk yöneticisi tarafından yönetilemiyor. Bu, kuyruk yöneticisinin hareket yöneticisi olarak değil, yalnızca kaynak yöneticisi olarak hareket edebileceği anlamına gelir. Ayrıca, istemci uygulamasının yalnızca dış hareket yöneticisi tarafından sağlanan uygulama programlama arabirimini (API) kullanarak iş birimini kesinleştirebileceği ya da geri alabileceği anlamına da gelir. Bu nedenle, istemci uygulaması MQI çağrılarını (**MQBEGIN**, **MQCMIT**ve **MQBACK**) kullanamıyor.

Dış hareket yöneticisi, kuyruk yöneticisiyle kaynak yöneticisi olarak, kuyruk yöneticisine bağlı istemci uygulamasının kullandığı MQI kanalını kullanarak iletişim kurar. Ancak, başarısızlığın ardından bir kurtarma durumunda, hiçbir uygulama çalışmadığında, hareket yöneticisi, hata sırasında kuyruk yöneticisinin katıldığı tamamlanmamış iş birimlerini kurtarmak için adanmış bir MQI kanalını kullanabilir.

Bu bölümde, genişletilmiş hareket işlevine sahip olmayan bir IBM MQ MQI client , IBM MQ temel istemcisi olarak adlandırılır. Bu nedenle, genişletilmiş işlemsel işlevin eklenmesiyle birlikte bir IBM MQ genişletilmiş işlemsel istemcisini IBM MQ temel istemcisinden oluşacak şekilde düşünebilirsiniz.

Not: IBM i IBM MQ MQI client on IBM i , IBM MQ genişletilmiş hareket işlevini desteklemez.

Genişletilmiş işlemsel istemciler için platform desteği

Multi

Bir temel istemciyi destekleyen tüm Çoklu Platformlar için genişletilmiş işlemsel istemciler kullanılabilir. İstemciler z/OSiçin kullanılamaz.

Genişletilmiş bir işlem istemcisi kullanan bir istemci uygulaması, yalnızca aşağıdaki IBM MQ 9.0ya da daha sonraki bir sürümdeki bir kuyruk yöneticisine bağlanabilir:

- 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 hareket istemcileri olmasa da, genişletilmiş hareket istemcisi 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 istemcisine ilişkin gereksinimlerle aynıdır. Bir programlama dili, IBM MQ temel istemcisi ve kullanmakta olduğunuz hareket yöneticisi tarafından destekleniyorsa, genişletilmiş bir işlem istemcisi tarafından desteklenir.

Tüm platformlara ilişkin dış hareket yöneticileri hakkında bilgi için bkz. IBM MQ için Sistem Gereksinimleri.

İstemcinin sunucuya bağlanma şekli

Bir istemci MQCONN ya da MQCONNX kullanarak bir sunucuya bağlanır ve bir kanal aracılığıyla iletişim kurar.

IBM MQ istemci ortamında çalışan bir uygulamanın, istemci ile sunucu makineleri arasında etkin bir bağlantı sürdürmesi gerekir.

Bağlantı, MQCONN ya da MQCONNX çağrısı yayınlayan bir uygulama tarafından yapılır. İstemciler ve sunucular *MQI kanalları*aracılığıyla iletişim kurar ya da paylaşım etkileşimlerini kullanırken her biri bir MQI kanalı eşgörünümünü paylaşır. Çağrı başarılı olduğunda, uygulama bir MQDISC çağrısı yayınlayana kadar

MQI kanalı eşgörünümü ya da etkileşimi bağlı kalır. Bu, bir uygulamanın bağlanması gereken her kuyruk yöneticisi için geçerli bir durumdur.

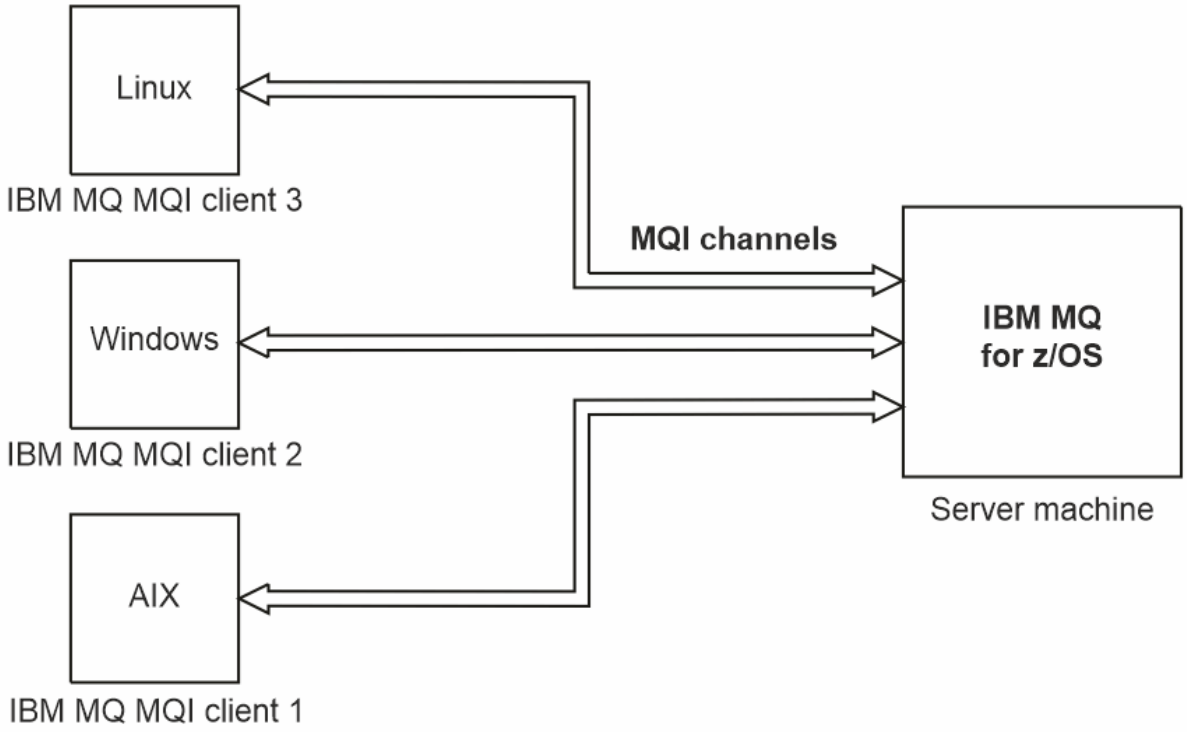
Aynı makinedeki istemci ve kuyruk yöneticisi

Makinenizde bir kuyruk yöneticisi de kuruluysa, IBM MQ MQI client ortamında bir uygulamayı çalıştırabilirsiniz.

Bu durumda, kuyruk yöneticisi kitaplıklarına ya da istemci kitaplıklarına bağlanma seçeneğiniz vardır; ancak, istemci kitaplıklarına bağlantı verdiyseniz, kanal bağlantılarını tanımlamanız gerektiğini unutmayın. Bu, bir uygulamanın geliştirme aşamasında yararlı olabilir. Programınızı, başkalarına bağımlı olmadan kendi makinenizde sınayabilir ve bağımsız bir IBM MQ MQI client ortamına taşıdığınızda bunun çalışmaya devam edeceğinden emin olabilirsiniz.

Farklı platformlardaki istemciler

Bu örnekte, sunucu makinesi farklı platformlarda üç IBM MQ MQI clients ile iletişim kurar.



Şekil 52. Farklı platformlardaki istemcilere bağlı IBM MQ sunucusu

Diğer daha karmaşık ortamlar da mümkündür. Örneğin, bir IBM MQ istemcisi birden çok kuyruk yöneticisine ya da bir kuyruk paylaşım grubunun bir parçası olarak bağlı herhangi bir sayıda 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 ve IBM MQ ' ın işlemleri nasıl desteklediğine ilişkin giriş.

Kaynak yöneticisi , uygulamalar tarafından erişilebilen ve güncellenebilecek kaynaklara sahip olan ve bunları yöneten bir bilgisayar altsistemidir. Aşağıda kaynak yöneticilerine ilişkin örnekler verilmiştir:

- Kaynakları kuyrukları olan bir IBM MQ kuyruk yöneticisi
- Tabloları olan kaynakları olan bir Db2 veritabanı

Bir uygulama bir ya da daha çok kaynak yöneticisinin kaynaklarını güncellediğinde, bazı güncellemelerin grup olarak başarıyla tamamlanmasını ya da hiçbirinin tamamlanmamasını sağlamak için bir iş gereksinimi olabilir. Bu tür bir gereksinimin nedeni, bu güncellemelerden bazıları başarıyla tamamlansa da diğerleri tamamlanmazsa, iş verilerinin tutarsız durumda bırakılması olabilir.

Bu şekilde yönetilen kaynaklarda yapılan güncellemelerin bir *iş birimi*ya da *işlem*içinde gerçekleşeceği belirtilir. Bir uygulama programı, bir güncelleme kümesini bir iş biriminde gruplayabilir.

Bir iş birimi sırasında uygulama, kaynak yöneticilerine kaynaklarını güncellemelerini ister. Uygulama, tüm güncellemeleri kesinleştirmek için bir istek yayınladığında iş birimi sona erer. Güncelleştirmeler kesinleştirilinceye kadar, bunların hiçbiri aynı kaynaklara erişen diğer uygulamalar tarafından görülemez. Alternatif olarak, uygulama herhangi bir nedenle iş birimini tamamlayamayacağına karar verirse, o noktaya kadar istediği tüm güncellemeleri geri almak için bir istek gönderebilir. Bu durumda, güncelleştirmelerin hiçbiri diğer uygulamalar tarafından görülemez. Bu güncellemeler genellikle mantıksal olarak ilişkilidir ve veri bütünlüğünün korunması için başarılı olması gerekir. Bir güncelleme başarılı olursa, diğeri başarısız olursa, veri bütünlüğü kaybolur.

Bir iş birimi başarıyla tamamlandığında, iş biriminin *kesinleştirileceği*söylenir. Kesinleştirildikten sonra, bu iş birimi içinde yapılan tüm güncellemeler kalıcı ve geri alınamaz hale getirilir. Ancak, iş birimi başarısız olursa, tüm güncellemeler *geriletilmiştir*. İş birimlerinin kesinleştirildiği ya da bütünlükle geriletildiği bu süreç, *eşitleme noktası koordinasyonu*olarak bilinir.

Bir iş birimindeki tüm güncellemelerin kesinleştirildiği ya da geriletildiği zaman noktasına *eşitleme noktası*denir. Bir iş birimi içindeki bir güncellemenin *eşitleme noktası denetimi içinde*gerçekleşeceği söylenir. Bir uygulama *Sync Point Control*dışında bir güncelleme isterse, devam eden bir iş birimi olsa bile, kaynak yöneticisi güncellemeyi hemen kesinleştirir ve güncelleme daha sonra geriletilemez.

İş birimlerini yöneten bilgisayar altsistemine *hareket yöneticisi*ya da *nokta koordinatörü*denir.

Yerel iş birimi, güncellenen kaynakların yalnızca IBM MQ kuyruk yöneticisinininkiler olduğu bir iş birimidir. Burada eşitleme noktası eşgüdümü, kuyruk yöneticisinin kendisi tarafından tek aşamalı bir kesinleştirme işlemi kullanılarak sağlanır.

Genel iş birimi, XA uyumlu veritabanları gibi diğer kaynak yöneticilerine ait kaynakların da güncellendiği bir birimdir. Burada, iki aşamalı bir kesinleştirme yordamı kullanılmalı ve iş birimi, kuyruk yöneticisinin kendisi ya da IBM TXSeriesya da BEA Tuxedo gibi başka bir XA uyumlu hareket yöneticisi tarafından eşgüdümlü olarak gerçekleştirilebilir.

Hareket yöneticisi, bir iş birimindeki kaynaklarda yapılan tüm güncellemelerin başarıyla tamamlanmasını ya da hiçbirinin tamamlanmamasını sağlamaktan sorumludur. Bir uygulamanın bir iş birimini kesinleştirme ya da geri alma isteği göndermesi bir hareket yöneticisine yapılır. Hareket yöneticilerine örnek olarak CICS ve WebSphere Application Serververilebilir, ancak bunların her ikisi de diğer işlevlere de sahiptir.

Bazı kaynak yöneticileri kendi hareket yönetimi işlevlerini sağlar. Örneğin, bir IBM MQ kuyruk yöneticisi, kendi kaynakları ve Db2 tablolarındaki güncellemeleri içeren iş birimlerini yönetebilir. Kuyruk yöneticisi bu işlevi gerçekleştirmek için ayrı bir hareket yöneticisine gereksinim duymaz; ancak, bu bir kullanıcı gereksinimiyse kullanılabilir. Ayrı bir hareket yöneticisi kullanılırsa, buna *dış hareket yöneticisi*denir.

Bir dış hareket yöneticisinin bir iş birimini yönetebilmesi için, hareket yöneticisi ile iş birimine katılan her kaynak yöneticisi arasında standart bir arabirim olmalıdır. 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. XA Arabirimi, Open Group tarafından *Distributed Transaction Processing: The XA Specification*(Dağıtılmış Hareket İşleme: XA Belirtimi) içinde yayınlanır.

Bir iş birimine birden çok kaynak yöneticisi katıldığında, bir hareket yöneticisinin iş birimindeki tüm güncellemelerin başarıyla tamamlanmasını sağlamak için *iki aşamalı kesinleştirme* protokolünü kullanması ya da bir sistem arızası olsa da bunların hiçbirinin tamamlanmamasını sağlaması gerekir. Bir uygulama bir iş birimini kesinleştirmek için hareket yöneticisine bir istek gönderdiğinde, hareket yöneticisi aşağıdakileri yapar:

1. Aşama (Kesinleştirmeye hazırla)
Hareket yöneticisi, iş birimine katılan her kaynak yöneticisinden, kaynaklarındaki amaçlanan güncellemelerle ilgili tüm bilgilerin kurtarılabilir durumda olduğundan emin olmasını ister. Bir kaynak yöneticisi normalde bunu, bilgileri bir günlüğe yazarak ve bilgilerin sabit diske yazılmasını sağlayarak yapar. 1. Aşama, hareket yöneticisi her kaynak yöneticisinden, kaynakları için amaçlanan güncellemelere ilişkin bilgilerin kurtarılabilir durumda olduğuna ilişkin bir bildirim aldığında tamamlanır.

2. Aşama (Kesinleştir)
Aşama 1 tamamlandığında, hareket yöneticisi iş birimini kesinleştirmek için geri alınamaz bir karar verir. İş birimine katılan her kaynak yöneticisinden, kaynaklarında yapılan güncellemeleri kesinleştirmesini ister. Bir kaynak yöneticisi bu isteği aldığında, güncellemeleri kesinleştirmelidir. Bu aşamada onları desteklemek gibi bir seçeneği yok. 2. Aşama, hareket yöneticisi her kaynak yöneticisinden kaynaklarıyla ilgili güncellemeleri kesinleştirdiğini bildiren bir bildirim aldığında tamamlanır.

XA Arabirimi iki aşamalı bir kesinleştirme protokolü kullanır.

Daha fazla bilgi için bkz. İşlemsel destek senaryoları.

IBM MQ , Microsoft Transaction Server (COM +) için de destek sağlar. Microsoft Transaction Server (COM +) olanağının kullanılması, COM + desteğinden yararlanmak için IBM MQ ' in nasıl ayarlanacağına ilişkin bilgi sağlar.

Kuyruk yöneticisi olanaklarını genişletme

Kullanıcı çıkışlarını, API çıkışlarını ya da kurulabilir hizmetleri kullanarak kuyruk yöneticisi olanaklarını genişletebilirsiniz.

Kullanıcı çıkışları

Kullanıcı çıkışları, kuyruk yöneticisi işlevine kendi kodunuzu eklemenizi sağlayacak bir mekanizma sağlar. Desteklenen kullanıcı çıkışları şunlardır:

Kanal çıkışları
Bu çıkışlar kanalların çalışma şeklini değiştirir. Kanal çıkışları, İleti sistemi kanalları için kanal çıkış programlarıiçinde açıklanır.

Veri dönüştürme çıkışları
Bu çıkışlar, verileri bir biçimden diğerine dönüştürmek için uygulama programlarına konabilecek kaynak kod parçaları yaratır. Veri dönüştürme çıkışları, Writing data-conversion exits(Veri dönüştürme çıkışlarının yazılması) başlıklı konuda açıklanmıştır.

Küme iş yükü çıkışı
Bu çıkışın gerçekleştirdiği işlev, çıkışın sağlayıcısı tarafından tanımlanır. Çağrı tanımlaması bilgileri MQ_CLUSTER_WORKLOAD_EXIT-Çağrı açıklamasıiçinde verilir.

API çıkışları

API çıkışları, MQPUT ve MQGET gibi IBM MQ API çağrılarının davranışını değiştiren kodu yazmanızı ve ardından bu kodu bu çağrılardan hemen önce ya da hemen sonra eklemenizi sağlar. Ekleme işlemi otomatik olarak gerçekleştirilir; kuyruk yöneticisi kayıtlı noktalarda çıkış kodunu yönlendirir. API çıkışları hakkında daha fazla bilgi için bkz. API çıkışlarını kullanma ve yazma.

Kurulabilir hizmetler

Kurulabilir hizmetler, birden çok giriş noktası olan resmi arabirimlere (API) sahiptir.

Kurulabilir bir hizmetin somutlamasına *hizmet bileşeni*adı verilir. IBM MQile verilen bileşenleri kullanabilir ya da gerek duyduğunuz işlevleri gerçekleştirmek için kendi bileşeninizi yazabilirsiniz.

Şu anda, aşağıdaki kurulabilir hizmetler sağlanır:

Yetkilendirme hizmeti
Yetkilendirme hizmeti, kendi güvenlik olanağınızı oluşturmanıza olanak sağlar.

Hizmeti gerçekleştiren varsayılan hizmet bileşeni, nesne yetki yöneticisidir (OAM). Varsayılan olarak, OAM etkindir ve bunu yapılandırmak için herhangi bir şey yapmanız gerekmez. OAM ' yi değiştirmek ya da genişletmek için başka bileşenler oluşturmak üzere yetkilendirme hizmeti arabirimini kullanabilirsiniz. OAM hakkında daha fazla bilgi için AIX, Linux, and Windows sistemlerinde güvenliği ayarlamabaşlıklı konuya bakın.

Ad hizmeti
Ad hizmeti, uygulamaların uzak kuyrukları yerel kuyruklarmış gibi tanımlayarak kuyrukları paylaşmalarını sağlar.

Kendi ad hizmeti bileşeninizi yazabilirsiniz. Örneğin, ad hizmetini IBM MQile kullanmak istiyorsanız bunu yapmak isteyebilirsiniz. Ad hizmetini kullanmak için, kullanıcı tarafından yazılmış 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 kullanılmak üzere üç uygulama programlama arabirimi (API) sağlar: IBM MQ classes for Jakarta Messaging, IBM MQ classes for JMSve IBM MQ classes for Java.

IBM , açık standartları destekler ve etkin bir katılımcıdır.

- IBM MQ 8.0' den ürün, paylaşılan abonelikler gibi özelliklerle birlikte yeni bir basitleştirilmiş API sunan JMS 2.0 standardını uygular.
- V 9.3.0 V 9.3.0 IBM MQ 9.3.0' dan Jakarta Messaging 3.0 da desteklenir.
- Ayrıca, WebSphere Liberty , IBM MQile JMS 2.0 ve Jakarta Messaging 3.0 ' yi destekler.

IBM MQ içinde Java uygulamalarında kullanılmak üzere üç API vardır:

V 9.3.0 V 9.3.0 JM 3.0 **IBM MQ classes for Jakarta Messaging**
IBM MQ classes for Jakarta Messaging , ileti sistemi olarak IBM MQ için Jakarta Messaging arabirimlerini uygulayan bir Jakarta Messaging sağlayıcısıdır. Jakarta Connectors Architecture , Jakarta EE ortamında çalışan uygulamaları IBM MQ ya da Db2gibi bir Kurumsal Bilgi Sistemine (EIS) bağlamanın standart bir yolunu sağlar.

JMS 2.0 **IBM MQ classes for JMS**
IBM MQ classes for JMS , ileti sistemi olarak IBM MQ için JMS arabirimlerini uygulayan bir JMS sağlayıcısıdır. Java Platform, Enterprise Edition Connector Architecture (JCA), Java EE ortamında çalışan uygulamaları IBM MQ ya da Db2gibi bir Kurumsal Bilgi Sistemine (EIS) bağlamanın standart bir yolunu sağlar.

IBM MQ classes for Java

IBM MQ classes for Java , Java ortamında IBM MQ kullanmanızı sağlar. IBM MQ classes for Java , bir Java uygulamasının IBM MQ istemcisi olarak IBM MQ ' e bağlanmasına ya da IBM MQ kuyruk yöneticisine doğrudan bağlanmasına izin verir.

Not:

- V 9.3.0 V 9.3.0 JMS 2.0 yerine Jakarta Messaginggeçti. IBM MQ classes for JMS , JMS 2.0 standardını desteklemeye devam eder, ancak Java ileti sisteminde gelecekteki geliştirmeler yalnızca Jakarta Messagingiçinde, dolayısıyla IBM MQ classes for Jakarta Messagingiçinde ortaya çıkar. IBM MQ classes for JMS yalnızca var olan JMS 2.0 uygulamalarının bakımı ve genişletilmesi için önerilir. IBM MQ classes for Jakarta Messaging , yeni geliştirme için tercih edilen teknoloji olmalıdır.
- Stabilized IBM MQ classes for Java , IBM MQ 8.0içinde gönderilen düzeyde işlevsel olarak sabitlenir. IBM MQ classes for Java kullanan var olan uygulamalar tam olarak desteklenmeye devam eder, ancak bu API dengelenir; bu nedenle yeni özellikler eklenmez ve geliştirme istekleri reddedilir. Tam olarak desteklenen, IBM MQ Sistem Gereksinimlerinde yapılan değişikliklerle birlikte hataların düzeltileceği anlamına gelir.

V 9.3.0 V 9.3.0 JM 3.0 IBM MQ 9.3içinden IBM MQ classes for Java, IBM MQ classes for JMSve IBM MQ classes for Jakarta Messaging Java 8ile oluşturulur. Bu düzeylerdeki ya da bu düzeylerin üstündeki Java çalıştırma zamanı ortamları, bu arabirimleri kullanarak uygulamaları çalıştırmak için kullanılmalıdır.

İlgili kavramlar

IBM MQ 'nun Java adresinden erişilmesi-API' nin seçimi

V 9.3.0 V 9.3.0 Jakarta Messaging için IBM MQ sınıflarını neden kullanmalıyım?

Neden IBM MQ classes for JMSkullanmalıyım?

Neden IBM MQ classes for Javakullanmalıyım?

IBM MQ classes for JMS/Jakarta Messaging

IBM MQ classes for JMS ve IBM MQ classes for Jakarta Messaging , IBM MQile verilen ileti alışverişi sağlayıcılarıdır. Bu sağlayıcıların her biri, ileti sistemi API 'si için iki uzantı kümesi de sağlar. Hem Java Platform, Standard Edition (Java SE) hem de Java Platform, Enterprise Edition (Java EE) uygulamaları bu ileti alışverişi sağlayıcılarını kullanabilir.

V 9.3.0 V 9.3.0 JM 3.0 IBM MQ 9.3.0 , Jakarta Messaging 3.0için destek sağlar. JMS 2.0 tam olarak desteklenmeye devam etmektedir.

JMS ve Jakarta Messaging belirtimleri, uygulamaların ileti alışverişi işlemlerini gerçekleştirmek için kullanabileceği bir arabirim kümesini tanımlar. Ürün, IBM MQ 8.0' den JMS standardının JMS 2.0 sürümünü destekler. Bu uygulama, klasik API ' nin tüm özelliklerini sunar, ancak daha az arabirim gerektirir ve kullanımı daha basittir. Daha fazla bilgi için bkz. “JMS ve Jakarta Messaging modeli” sayfa 146 ve JMS 2.0 belirtimi: Java.net. JM 3.0 IBM MQ 9.3.0' dan Jakarta Messaging da desteklenir.

jakarta.jms (Jakarta Messaging 3.0) ya da javax.jms (JMS 2.0) paketi, ileti alışverişi arabirimlerinin ayrıntılarını belirtir ve bir ileti alışverişi sağlayıcısı bu arabirimleri belirli bir ileti sistemi ürünü için uygular. Örneğin:

- IBM MQ classes for JMS , IBM MQ için JMS arabirimlerini uygulayan ve JMS API 'si için aşağıdaki iki uzantı kümesini sağlayan bir JMS sağlayıcısıdır:
  - IBM MQ JMS uzantıları
  - IBM JMS uzantıları
- javax.dimsya da jakarta.jmsarabirimleri ya da JMS uzantıları kümesi kullanılarak yaratılan bir bağlantı üreticisi, kuyruk ya da konu nesnesi, bu API ' lardan herhangi biri kullanılarak adreslenebilir; başka bir deyişle, arabirimlerden herhangi birine dönüştürülebilir. Uygulama taşınabilirliğini en üst düzeyde korumak için gereksinimlerinize uygun en genel API ' yi kullanın.

JMS ve Jakarta Messaging ortak çok şey paylaştığı için, bu konuda JMS ile ilgili daha fazla başvuru, her ikisine de atıfta bulunma olarak alınabilir. Tüm farklar gerektiği şekilde vurgulanır.

IBM MQ JMS uzantıları

IBM MQ classes for JMS ayrıca JMS API 'si için uzantılar sağlar. IBM MQ classes for JMS , MQConnectionFactory, MQQueue ve MQTopic nesnelerinde uygulanan uzantıları içerir. Bu nesnelerin IBM MQ' e özgü özellikleri ve yöntemleri vardır. Nesneler denetlenen nesneler olabilir ya da bir uygulama nesneleri yürütme sırasında dinamik olarak yaratabilir. Bu uzantılar, IBM MQ JMS uzantıları olarak adlandırılır. Bu belgede, yürütme sırasında bir uygulama tarafından dinamik olarak yaratılan nesnelerin denetlenen nesneler olarak kabul edilmediğini unutmayın.

IBM JMS uzantıları

IBM MQ JMS uzantılarına ek olarak IBM MQ classes for JMS , kullanılan programlama dili olarak JMS API ya da Java için daha genel bir uzantı kümesi sağlar. Bu uzantılar, IBM JMS uzantıları olarak adlandırılır ve aşağıdaki geniş hedeflere sahip olur:

- IBM JMS sağlayıcıları arasında daha yüksek bir tutarlılık düzeyi sağlamak için.
- İki IBM ileti sistemi arasında bir köprü uygulaması yazmayı kolaylaştırmak için.
- Bir uygulamayı bir IBM JMS sağlayıcısından diğerine bağlamayı kolaylaştırmak için.

Bu uzantıların ana odağı, çalıştırma zamanında bağlantı fabrikalarının ve hedeflerinin dinamik olarak oluşturulmasını ve yapılandırılmasını ilgilendirir, ancak uzantılar, sorun belirleme işlevi gibi ileti sistemiyle doğrudan ilişkili olmayan işlevler de sağlar.

İlgili görevler

JMS/Jakarta İleti Sistemi için IBM MQ sınıflarının kullanılması

JMS ve Jakarta Messaging kaynaklarının yapılandırılması

V 9.3.0 V 9.3.0 JM 3.0 IBM MQ classes for Jakarta Messaging: genel bakış

IBM MQ 9.3.0 , Jakarta Messagingiçin destek sağlar. Jakarta Messaging 3.0için, JMS belirtiminin denetimi Oracle öğesinden Java Topluluk Sürecine taşındı. Ancak Oracle , diğer Java teknolojilerinde kullanılan "javax" adının denetimini elinde tutar. Bu nedenle Jakarta Messaging 3.0 , JMS 2.0ile işlevsel olarak eşdeğer olsa da, adlandırmada bazı farklılıklar vardır. 3.0 sürümünün resmi adı Java Message Serviceyerine Jakarta Messaging ve paket ve sabit adlarının başına `javax`yerine `jakarta` öneki konmuştur.

Arka Plan

Java platformu uzun yıllar boyunca Standard Edition ve Enterprise Editionolmak üzere iki biçimde geldi.

Java Platform, Standard Edition (bazen Java SEolarak kısaltılır), bağımsız bir bağlamda çalıştırılabilen temel dil ve sınıf kitaplıkleridir. Java SE içindeki çoğu Java paketi, "java" ile başlayan adlara sahip.

Java Platform, Enterprise Edition (Java EE), İleti Sistemi, çeşitli Bean 'lar, hareketsellik vb. gibi işlevleri ekleyerek bunu genişletir. Bu teknolojilerden bazıları Java SE bağlamında da kullanılabilir. Java EE içindeki çoğu Java paketi, "javax" ile başlayan adlara sahip. Ancak bazı çapraz tablolar vardır, bu nedenle bazı Java SE paketlerinde "javax" vardır. adlarına önek olarak.

Java Message Service (JMS), Java Platform, Enterprise Edition' in bir parçasıdır. Java EE 7 , JMS 2.0ürününü birleştirir.

Java EE 7'e kadar, teknolojiler Oracle' un himayesi altındaydı.

Java EE teknolojileri kısa bir süre önce Oracle ' un hostesinden Eclipse Foundation tarafından denetlenen bir topluluk sürecine geçti.

"Javax" olarak. ad yeni projeye taşınamadı, yeni adlandırma benimsendi-tüm paketlerin ve özellik adlarının başına "jakarta" öneki eklendi. ve Java Platform, Enterprise Edition , gelecekte "Jakarta EE" olarak adlandırılacaktır. Sürüm numaralandırması devam etti: Sürüm 8, büyük ölçüde göz ardı edilebilecek geçici bir sürümdü ve Jakarta EE 9 , "jakarta" nın olduğu noktadır. önek geçerli olur.

IBM MQ bağlamında geçerli olan ana Jakarta EE teknolojisi Jakarta Messaging 3.0 - Java Message Service (JMS) 2.0' ın ardıl teknolojisidir. Bu nedenle Jakarta EE 9 , Jakarta Messaging 3.0ürününü birleştirir.

IBM MQ , Jakarta EE 9 ve Jakarta Messaging 3.0için destek sunarken Java EE 7 ve JMS 2.0desteklemeye devam eder.

Teslim edilenler: Java SE

Java Platform, Standard Editioniçin, IBM MQ classes for JMS (IBM MQile JMS 2.0 işlemlerini destekleyen) IBM MQ 9.3.0 ek olarak IBM MQ classes for Jakarta Messagingsağlar. Bu sınıflar, IBM MQile bütünleşen bir Jakarta Messaging 3.0 sağlayıcısı sağlayarak IBM MQ kuyruk yöneticilerinin Jakarta Messaging işlemlerini kolaylaştırmasını sağlar.

Bunlar, IBM MQ kuruluşunun `java/lib` alt dizininde standart bir JAR dosyası (`com.ibm.mq.jakarta.client.jar`) olarak sağlanır.

Apache Felix ya da Eclipse Equinox IBM MQ gibi OSGi kaplarında kullanılmak üzere bir çift OSGi kod paketi de sağlar:

- `com.ibm.mq.osgi.jms30.clientprereqs_`*`V.R.M.F`*`.jar`
- `com.ibm.mq.osgi.jms30.client_`*`V.R.M.F`*`.jar`

Burada *V.R.M.F* , IBM MQsürümünü temsil eder; örneğin, 9.3.0.0. Bu kod paketleri, IBM MQ kuruluşunun `java/lib/OSGi` alt dizininde bulunabilir.

Teslim edilenler: Jakarta EE 9

Jakarta EE 9 uyumlu bir uygulama sunucusunda IBM MQtabanlı ileti alışverişini desteklemek için IBM MQ , Jakarta EE 9uyumlu bir Kaynak Bağdaştırıcısı sağlar: `wmq.jakarta.jmsra.rar`. Bu, IBM MQ kuruluşunun `java/lib/jca` alt dizininde bulunabilir.

IBM MQ , IBM MQ kuruluşunun `java/lib/jca` alt dizininde Java EE 7 uyumlu bir Kaynak Bağdaştırıcısı (`wmq.jmsra.rar`) sağlamaya devam eder.

Bu yapay nesnelerin nasıl sağlandığını

These JARs and the RAR file for the Resource Adapter are packaged with the preexisting artifacts in the usual IBM MQ installation media – both the platform-specific installation media, such as "`.rpm`" files, and the redistributable media, such as the self-extracting redistributable client JAR files.

JMS 2.0 ile Jakarta Messaging 3.0 arasında ne değişti?

Jakarta EE 9 ve Jakarta Messaging 3.0 , yeni bir işlev sunmaz. Tüm bu değişimler isimlerden ibaret. Örneğin, JMS 2.0içinde "javax.jms.Connection" kullandığınızda, Jakarta Messaging 3.0içinde "jakarta.jms.Connection" kullanırsınız.

Eclipse Foundation Jakarta EE platformunu ileriye götürdüğü için, bu temel üzerine inşa edilecek ve bu adlandırma kuralı gelecekte tanıtılan yeni işlevler için kullanılacaktır.

IBM MQ classes for JMS ile IBM MQ classes for Jakarta Messaging arasında ne değişti?

Özet

JMS 2.0için destek sağlayan IBM MQ classes for JMS, mevcut olmaya devam eder ve öncelikle var olan uygulamaların bakımı ve genişletilmesi için önerilir. Tam olarak destekleniyorlar.

Jakarta Messaging 3.0için destek sağlayan IBM MQ classes for Jakarta Messaging, yeni geliştirme için önerilir.

IBM MQ 9.3.0adresinde bu iki olanak işlevsel olarak eşdeğerdir. Yalnızca adlandırma farklıdır. Ancak, yeni ileti sistemi işlevselliğinin IBM MQ classes for Jakarta Messaging içinde ortaya çıkma olasılığı IBM MQ classes for JMSiçinde ortaya çıkma olasılığına göre daha yüksektir.

Bu iki olanak birlikte çalışabilir. IBM MQ classes for JMS tarafından üretilen iletiler IBM MQ classes for Jakarta Messagingtarafından tüketilebilir, tersi de geçerlidir. Ancak bu iki olanak tek bir uygulamada birlikte var olmamalıdır.

Adlandırma değişiklikleri

Çizelge 16. Paket adlarında yapılan değişiklikler

IBM MQ classes for JMS paket adı	**IBM MQ classes for Jakarta Messaging paket adı**
com.ibm.mq.jms[.*]	com.ibm.mq.jakarta.jms[.*]
com.ibm.jms	com.ibm.jakarta.jms
com.ibm.msg.client.jms.*	com.ibm.msg.client.jakarta.jms.*
com.ibm.msg.client.wmq.*	com.ibm.msg.client.jakarta.wmq.*

Ortak hizmetlerle (izleme, günlüğe kaydetme, ulusal dil desteği vb.) ve JMQI uygulamalarıyla (yerel ve uzak) ilgili paketler hem IBM MQ classes for JMS hem de IBM MQ classes for Jakarta Messagingiçin ortaktır, bu nedenle bu alanlarda herhangi bir değişiklik yapılması gerekmez.

Özellik adlarının da değiştiğini unutmayın. Örneğin, IBM MQ classes for Jakarta Messaging içinde IBM MQ uzantılarını etkinleştirme özelliği **`com.ibm.mq.jakarta.jms.SupportMQExtensions`**' dir.

Çeşitli **`com.ibm.msg.client.commonservices.trace.*`** özellikleri gibi IBM MQ classes for JMS veya IBM MQ classes for Jakarta Messagingözelliklerinden bağımsız olan özellik adları, her iki olanak için de eşit olarak geçerlidir.

Yönetim yardımcı programları

`crtmqenv` ve **`setmqenv`** yardımcı programları artık IBM MQ classes for JMS (`-j 2.0`) ya da IBM MQ classes for Jakarta Messaging (`-j 3.0`) için sınıf yolunun yapılandırılıp yapılandırılmayacağını belirten bir seçeneği kabul eder ve **`runjms30`** adlı **`runjms`** yardımcı programlarının IBM MQ classes for Jakarta Messaging çeşitlemeleri ve benzeri adlar vardır.

`dspmqver` yardımcı programı, Java bileşenleriyle ilgili raporlama istendiğinde, çıkışında IBM MQ classes for Jakarta Messaging öğesini içerir.

JNDI yoluyla alınacak IBM MQ classes for Jakarta Messaging nesnelerini yapılandırmak için, yeni **`JMS30Admin`** yardımcı programı IBM MQ classes for JMSiçin **`JMSAdmin`** yardımcı programının eşdeğeridir.

Temeldeki nesnelerin farklı paketlerden olduğunu unutmayın. **`JMSAdmin`** tarafından yaratılan JNDI tanımlamaları IBM MQ classes for Jakarta Messagingtarafından kullanılamaz ve **`JMS30Admin`** tarafından yaratılan tanımlamalar IBM MQ classes for JMStarafından kullanılamaz.

Not: IBM MQ Explorertarafından sağlanan IBM MQ classes for Jakarta Messaging nesneleri için destek yoktur; JNDI bütünleştirmesi yalnızca IBM MQ classes for JMS içindir.

İlgili kavramlar

Jakarta Messaging için IBM MQ sınıflarını neden kullanmalıyım?

JMS ve Jakarta Messaging modeli

JMS ve Jakarta Messaging modeli, Java uygulamalarının ileti alışverişi işlemlerini gerçekleştirmek için kullanabileceği bir arabirim kümesini tanımlar. IBM MQ classes for JMS ve IBM MQ classes for Jakarta Messaging , Java ileti alışverişi nesnelerinin IBM MQ kavramlarıyla nasıl ilişkili olduğunu tanımlayan

ileti alışverişi sağlayıcılarıdır. JMS ve Jakarta Messaging belirtimleri, belirli ileti sistemi nesnelerinin denetlenmesini bekler.

IBM MQ 8.0'den ürün, JMS 1.1' den klasik API 'yi korurken basitleştirilmiş bir API' yi tanıtan JMS standardının JMS 2.0 sürümünü destekler.

V 9.3.0 V 9.3.0 JM 3.0 IBM MQ 9.3.0 , Jakarta Messaging 3.0için destek sağlar. JMS 2.0 tam olarak desteklenmeye devam etmektedir. JMS ve Jakarta Messaging ortak çok şey paylaştığı için, bu konuda JMS ile ilgili daha fazla başvuru, her ikisine de atıfta bulunma olarak alınabilir. Tüm farklar gerektiği şekilde vurgulanır.

Basitleştirilmiş API

JMS 2.0 , etki alanına özgü ve etki alanından bağımsız arabirimleri JMS 1.1'den korurken basitleştirilmiş API' yi tanıttı. Basitleştirilmiş API, ileti göndermek ve almak için gereken nesne sayısını azaltır ve aşağıdaki arabirimlerden oluşur:

ConnectionFactory
ConnectionFactory , bir JMS istemcisi tarafından bağlantı yaratmak için kullanılan denetlenen bir nesnedir. Bu arabirim, klasik API 'de de kullanılır.

JMSBağlam
Bu nesne, klasik API ' nin Bağlantı ve Oturum nesnelerini birleştirir. JMSBağlam nesneleri, temeldeki bağlantı yinelenirken diğer JMSBağlam nesnelerinden yaratılabilir.

JMSÜretici
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.

JMSTüketim
JMSConsumer bir JMSContext tarafından oluşturulur ve bir konudan ya da kuyruktan ileti almak için kullanılır.

Basitleştirilmiş API ' nin çeşitli etkileri vardır:

- JMSBağlam nesnesi her zaman temel bağlantıyı otomatik olarak başlatır.
- JMSÜreticiler ve JMSTüketiciler artık iletinin getBody yöntemini kullanarak tüm ileti nesnesini almak zorunda kalmadan doğrudan ileti gövdeleriyle çalışabilirler.
- JMSProducer nesnesinde, bir 'body' göndermeden önce yöntem zincirleme kullanılarak ileti özellikleri ayarlanabilir. JMSÜreticisi, iletiyi göndermek için gereken tüm nesnelerin yaratılmasını işleyecek. JMS 2.0kullanılarak özellikler ayarlanabilir ve aşağıdaki gibi bir ileti gönderilebilir:

```
context.createProducer().
setProperty("foo", "bar").
setTimeToLive(10000).
setDeliveryMode(NON_PERSISTENT).
setDisableMessageTimestamp(true).
send(dataQueue, body);
```

JMS 2.0 , iletilerin birden çok tüketici arasında paylaşılabileceği paylaşılan abonelikleri de tanıttı. Tüm JMS 1.1 abonelikleri paylaşılmayan abonelikler olarak kabul edilir.

Klasik API

Aşağıdaki liste, klasik API ' nin ana JMS arabirimlerini özetler:

Hedef
Hedef, bir uygulamanın iletileri gönderdiği ya da bir uygulamanın iletileri aldığı ya da her ikisini birden aldığı bir kaynaktır.

ConnectionFactory
ConnectionFactory nesnesi, bir bağlantıya ilişkin yapılandırma özellikleri kümesini içerir. Uygulama, bağlantı yaratmak için bir bağlantı üreticisi kullanır.

Bağlantı
Bağlantı nesnesi, bir uygulamanın bir ileti sistemi sunucusuyla olan etkin bağlantısını kapsüller. Bir uygulama, oturum yaratmak için bir bağlantı kullanır.

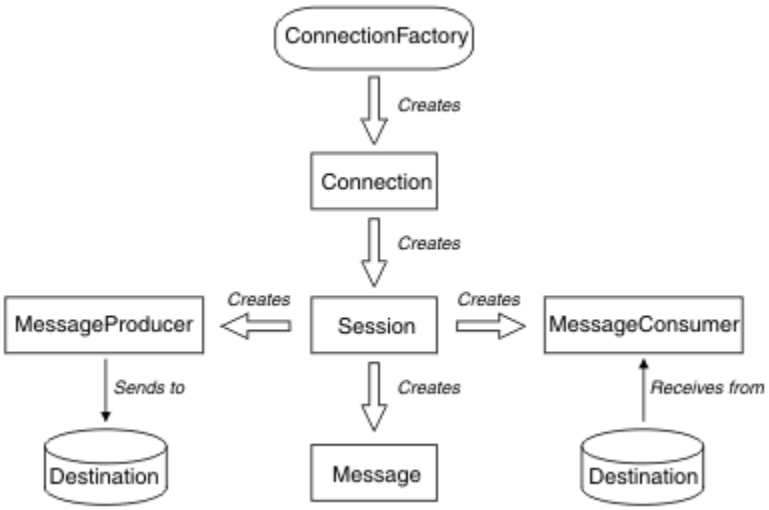
Oturum
Oturum, ileti göndermek ve almak için tek iş parçacıklı bir bağlamdır. Bir uygulama, ileti, ileti üreticileri ve ileti tüketicileri yaratmak için oturum kullanır. Bir oturum işlemden geçildi ya da işlem yapılmadı.

İleti
İleti nesnesi, bir uygulamanın gönderdiği ya da aldığı bir iletiyi içerir.

MessageProducer
Bir uygulama, bir hedefe ileti göndermek için ileti üreticiyi kullanır.

MessageConsumer
Bir uygulama, hedefe gönderilen iletileri almak için bir ileti tüketicisi kullanır.

Şekil 53 sayfa 148 içinde bu nesneler ve ilişkileri gösterilmektedir.



Şekil 53. JMS nesneleri ve ilişkileri

Çizge, ana arabirimleri gösterir: ConnectionFactory, Bağlantı, Oturum, MessageProducer, MessageConsumer, İleti ve Hedef. Bir uygulama, bağlantı yaratmak için bir bağlantı üreticisi kullanır ve oturum yaratmak için bir bağlantı kullanır. Uygulama daha sonra ileti, ileti üreticileri ve ileti tüketicileri yaratmak için bir oturum kullanabilir. Uygulama, bir hedefe ileti göndermek için bir ileti üreticisi kullanır ve hedefe gönderilen iletileri almak için bir ileti tüketicisi kullanır.

Bir Hedef, ConnectionFactoryya da Bağlantı nesnesi, çok iş parçacıklı bir uygulamanın farklı iş parçacıkları tarafından eşzamanlı olarak kullanılabilir, ancak bir Oturum, MessageProducerya da MessageConsumer nesnesi farklı iş parçacıkları tarafından eşzamanlı olarak kullanılamaz. Bir Oturum, MessageProducerya da MessageConsumer nesnesinin eşzamanlı olarak kullanılmamasını sağlamanın en basit yolu, her iş parçacığı için ayrı bir Oturum nesnesi yaratmaktır.

JMS iki ileti alışverişi biçemini destekler:

- Noktadan noktaya ileti sistemi
- İleti alışverişi yayınla/abone ol

Bu ileti alışverişi stillerine *ileti alışverişi etki alanları*da denir ve bir uygulamada her iki ileti alışverişi biçemini de birleştirebilirsiniz. Noktadan noktaya iletişim etki alanında hedef bir kuyruktur ve yayınlama/abone olma etki alanında hedef bir konudur.

JMS before JMS 1.1sürümleriyle, noktadan noktaya iletişim etki alanı için programlama bir arabirim ve yöntem kümesi kullanır ve yayınlama/abone olma etki alanı için programlama başka bir küme kullanır. İki grup birbirine benzer ama ayrı. JMS 1.1' den her iki ileti sistemi etki alanını destekleyen ortak bir arabirim ve yöntem kümesini kullanabilirsiniz. Ortak arabirimler, her ileti sistemi etki alanının etki alanından

bağımsız bir görünümünü sağlar. Çizelge 17 sayfa 149 içinde JMS etki alanından bağımsız arabirimler ve bunlara karşılık gelen etki alanına özgü arabirimler listelenir.

Çizelge 17. JMS etki alanından bağımsız ve bunlara karşılık gelen etki alanına özgü arabirimler

Etki alanından bağımsız arabirimler	**Noktadan noktaya etki alanı için etki alanına özgü arabirimler**	**Yayınlama/abone olma etki alanı için etki alanına özgü arabirimler**
ConnectionFactory	QueueConnectionÜreticisi	TopicConnectionÜreticisi
Bağlantı	QueueConnection	TopicConnection
Hedef	Kuyruk	Konu
Oturum	QueueSession	TopicSession
MessageProducer	QueueSender	TopicPublisher
MessageConsumer	QueueReceiver QueueBrowser	TopicSubscriber

JMS 2.0 IBM MQ classes for JMS 2.0 , hem önceki JMS 1.1 etki alanına özgü arabirimleri hem de JMS 2.0basitleştirilmiş API 'sini destekler. Bu nedenle IBM MQ classes for JMS 2.0 , var olan uygulamalarda yeni işlev geliştirilmesi de dahil olmak üzere var olan uygulamaların bakımı için kullanılabilir.

JM 3.0 IBM MQ classes for Jakarta Messaging 3.0 , aynı arabirimlerin Jakarta Messaging sürümlerini destekler ve yeni uygulama geliştirme için önerilir.

IBM MQ classes for JMS ve IBM MQ classes for Jakarta Messagingsistemlerinde JMS nesneleri IBM MQ kavramlarıyla aşağıdaki şekillerde ilişkilidir:

- Bağlantı nesnesi, bağlantıyı yaratmak için kullanılan bağlantı üreticisinin özelliklerinden türetilen özelliklere sahiptir. 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 verilebilir.
- Oturum nesnesi, oturumun hareket kapsamını tanımlayan bir IBM MQ bağlantı tanıtıcısını içerir.
- MessageProducer nesnesi ve MessageConsumer nesnesi, her biri bir IBM MQ nesne tanıtıcısını kapsıyor.

IBM MQ classes for JMS ya da IBM MQ classes for Jakarta Messagingkullanırken, IBM MQ ' in tüm normal kuralları geçerlidir. Özellikle, bir uygulamanın uzak kuyruğa ileti gönderebildiğini, ancak yalnızca uygulamanın bağlı olduğu kuyruk yöneticisinin iyeliğindeki bir kuyruktan ileti alabildiğini unutmayın.

JMS belirtimi, ConnectionFactory ve Destination nesnelerinin denetlenmesini bekler. Bir denetimci merkezi bir havuzda yönetilen nesneler yaratır ve bunların bakımını yapar; JMS uygulaması bu nesneleri Java Naming and Directory Interface (JNDI) olanağını kullanarak alır.

IBM MQ classes for JMS ve IBM MQ classes for Jakarta Messagingiçinde, Hedef arabiriminin somutlaması Kuyruk ve Konunun soyut bir üst sınıfıdır; dolayısıyla, Hedef somut örneği bir Kuyruk nesnesi ya da Konu nesnesidir. Etki alanından bağımsız arabirimler, bir kuyruğu ya da konuyu hedef olarak görür. Bir MessageProducer ya da MessageConsumer nesnesine ilişkin ileti alışverişi etki alanı, hedefin bir kuyruk mu, yoksa bir konu mu olduğuna göre belirlenir.

Bu nedenle, IBM MQ classes for JMS ve IBM MQ classes for Jakarta Messaging içinde, aşağıdaki tipteki nesneler denetlenebilirler:

- ConnectionFactory
- QueueConnectionÜreticisi
- TopicConnectionÜreticisi
- Kuyruk
- Konu

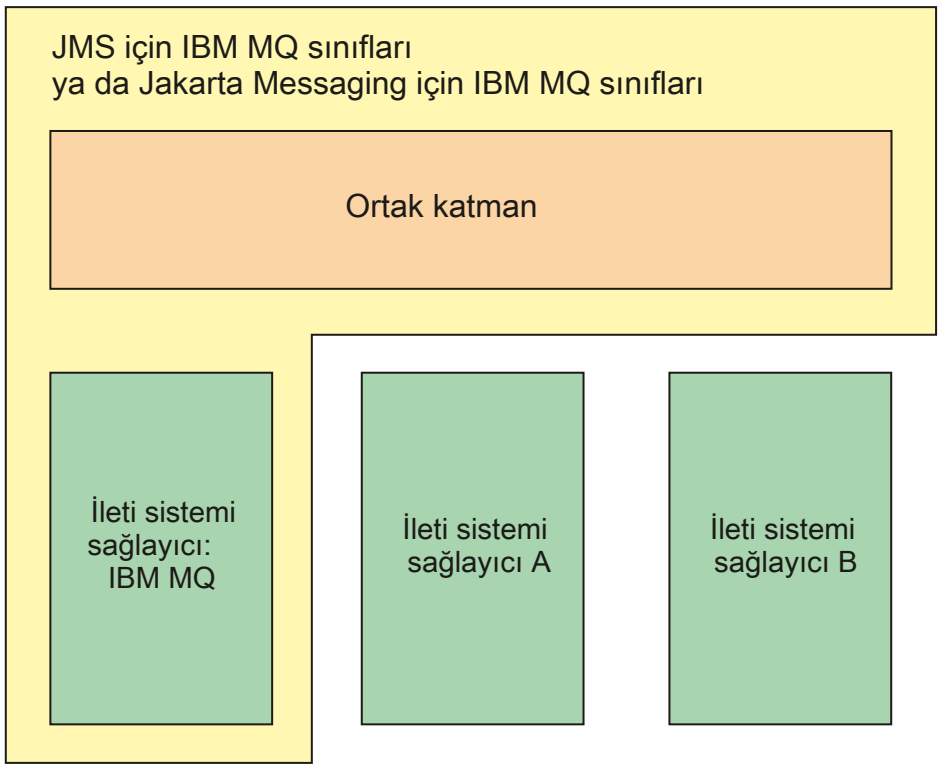
- XAConnectionFactory
- XAQueueConnectionÜreticisi
- XATopicConnectionÜreticisi

IBM MQ classes for JMS/Jakarta Messaging mimarisi

IBM MQ classes for JMS ve IBM MQ classes for Jakarta Messaging katmanlı bir mimariye sahiptir. En üstteki kod katmanı, herhangi bir IBM Java ileti sistemi sağlayıcısının kullanabileceği ortak bir katmandır.

V 9.3.0 V 9.3.0 JM 3.0 IBM MQ 9.3.0 , Jakarta Messaging 3.0için destek sağlar. JMS 2.0 tam olarak desteklenmeye devam etmektedir.

IBM MQ classes for JMS ve IBM MQ classes for Jakarta Messaging , şemada gösterildiği gibi katmanlı bir mimariye sahiptir Şekil 54 sayfa 150. Kodun en üst katmanı, herhangi bir IBM JMS ya da Jakarta Messagingsağlayıcısı tarafından kullanılabilen ortak bir katmandır. Bir uygulama bir JMS ya da Jakarta Messagingyöntemini çağırdığında, bir ileti sistemi sistemine özgü olmayan çağrı işlemleri ortak katman tarafından gerçekleştirilir; bu işlem, çağrıya tutarlı bir yanıt da sağlar. Bir ileti sistemi sistemine özgü çağrı işlemleri daha düşük bir katmana devredilir. Aşağıdaki şemada, IBM MQ ileti alışverişi sağlayıcısı alt katmanda, diğer iki ileti alışverişi sağlayıcısıyla birlikte gösterilir (İleti alışverişi sağlayıcısı A ve İleti alışverişi sağlayıcısı B).



Şekil 54. IBM JMS ve Jakarta Messaging sağlayıcıları için katmanlı mimari

Katmanlı bir mimari aşağıdaki hedefleri karşılar:

- Çeşitli IBM JMS ve Jakarta Messaging sağlayıcılarının davranışlarının tutarlılığını artırmak için
- İki IBM ileti sistemi arasında bir köprü uygulaması yazmayı kolaylaştırmak için
- Bir uygulamayı bir IBM JMS ya da Jakarta Messaging sağlayıcısından diğerine bağlamayı kolaylaştırmak için

İlgili görevler

JMS/Jakarta İleti Sistemi için IBM MQ sınıflarının kullanılması

Denetlenen nesneler için destek

IBM MQ classes for JMS ve IBM MQ classes for Jakarta Messaging , yönetilen nesnelerin kullanımını destekler.

V 9.3.0 V 9.3.0 JM 3.0 IBM MQ 9.3.0' dan Jakarta Messaging 3.0 , yeni uygulamalar geliştirmek için desteklenir. IBM MQ 9.3.0 , var olan uygulamalar için JMS 2.0 ' e destek vermeye devam eder. Aynı uygulamada hem Jakarta Messaging 3.0 API hem de JMS 2.0 API ' nin kullanılması desteklenmez. Daha fazla bilgi için, bkz. JMS/Jakarta İleti Sistemi için IBM MQ sınıflarının kullanılması.

Bir JMS ya da IBM MQ classes for Jakarta Messaging uygulaması içindeki mantık akışı ConnectionFactory ve Destination nesneleriyle başlar. Uygulama, uygulamadan ileti sistemi sunucusuna etkin bağlantıyı gösteren bir Connection nesnesi yaratmak için bir ConnectionFactory nesnesi kullanır. Uygulama, iletileri üretmek ve tüketmek için tek iş parçacıklı bağlam olan bir Oturum nesnesi yaratmak için Connection nesnesini kullanır. Daha sonra uygulama, uygulamanın belirtilen hedefe ileti göndermek için kullandığı bir MessageProducer nesnesi yaratmak için Oturum nesnesini ve bir Hedef nesnesini kullanabilir. Hedef, ileti sistemi sistemindeki bir kuyruk ya da konudur ve Hedef nesne tarafından kapsüllenir. Uygulama, uygulamanın belirtilen hedefe gönderilen iletileri almak için kullandığı bir MessageConsumer nesnesi oluşturmak için Oturum nesnesini ve bir Hedef nesnesini de kullanabilir.

JMS ve Jakarta Messaging belirtimleri, ConnectionFactory ve Destination nesnelerinin denetlenmesini bekler. Denetimci, yönetilen nesneleri merkezi bir havuzda yaratır ve saklar; bir JMS ya da Jakarta Messaging uygulaması, Java Naming Directory Interface (JNDI) kullanarak bu nesneleri alır. Yönetilen nesnelerin havuzu, basit bir dosyadan LDAP (Lightweight Directory Access Protocol; Temel Dizin Erişimi Protokolü) dizinine kadar değişebilir.

IBM MQ classes for JMS ve IBM MQ classes for Jakarta Messaging , yönetilen nesnelerin kullanımını destekler. Bir uygulama, uygulamanın kendisinde sabit olarak kodlanmış herhangi bir IBM MQözel bilgisine sahip olmadan IBM MQ aracılığıyla gösterilen tüm IBM MQ classes for JMS ya da IBM MQ classes for Jakarta Messaging özelliklerini kullanabilir. Bu düzenleme, uygulamaya temel IBM MQ yapılandırmasından bir derece bağımsızlık sağlar.

Bu bağımsızlığı elde etmek için uygulama, yönetilen nesneler olarak saklanan bağlantı üreticilerini ve hedefleri almak üzere JNDI ' yı kullanabilir ve ileti alışverişi işlemlerini gerçekleştirmek için yalnızca `javax.jms` (JMS 2.0) ya da `jakarta.jms` (Jakarta Messaging 3.0) paketinde tanımlanan arabirimleri kullanabilir.

JMS 2.0 JMS 2.0için bir yönetici, merkezi bir havuzda yönetilen nesneler oluşturmak ve bunların bakımını yapmak için IBM MQ JMS yönetim aracını **JMSAdmin**ya da IBM MQ Explorerkullanabilir.

JM 3.0 Jakarta Messaging 3.0için, IBM MQ Explorerkullanarak JNDI ' yi yönetemezsiniz. JNDI yönetimi, **JMSAdmin**' ın **JMS30Admin**olan Jakarta Messaging 3.0 çeşitlemesi tarafından desteklenir.

Bir uygulama sunucusu genellikle yönetilen nesneler için kendi havuzunu ve nesneleri oluşturmak ve korumak için kendi araçlarını sağlar. Bu nedenle bir Java EE JM 3.0 ya da Jakarta EE uygulaması, denetlenen nesneleri uygulama sunucusu havuzundan ya da merkezi bir havuzdan almak için JNDI komutunu kullanabilir.

İlgili görevler

JMS ve Jakarta Messaging kaynaklarının yapılandırılması

Java EE ve Jakarta EE platformlarında desteklenen iletişim tipleri

Java EE ve Jakarta EE platformlarında IBM MQ classes for JMS ve IBM MQ classes for Jakarta Messaging , bir uygulamanın bileşeni ile IBM MQ kuyruk yöneticisi arasında iki tip iletişimi destekler.

V 9.3.0 V 9.3.0 JM 3.0 IBM MQ 9.3.0 , Jakarta Messaging 3.0için destek sağlar. JMS 2.0 tam olarak desteklenmeye devam etmektedir. JMS ve Jakarta Messaging ortak çok şey paylaştığı için, bu konuda JMS ile ilgili daha fazla başvuru, her ikisine de atıfta bulunma olarak alınabilir. Tüm farklar gerektiği şekilde vurgulanır.

Bir uygulamanın bir bileşeni ile bir IBM MQ kuyruk yöneticisi arasında aşağıdaki iki iletişim tipi desteklenir:

- Giden iletişim
- Gelen iletişim

Giden iletişim

JMS ya da Jakarta Messaging API ' yı doğrudan kullanarak bir uygulama bileşeni bir kuyruk yöneticisine bağlantı yaratır ve sonra ileti gönderir ve alır.

Örneğin, uygulama bileşeni bir uygulama istemcisi, sunucu uygulamacığı, Java Server Page (JSP), enterprise Java bean (EJB) ya da ileti odaklı bean (MDB) olabilir. Bu iletişim tipinde, uygulama sunucusu taşıyıcısı yalnızca, bağlantı havuzlama ve iş parçacığı yönetimi gibi ileti alışverişi işlemlerini desteklemek için düşük düzeyli işlevler sağlar.

Gelen iletişim

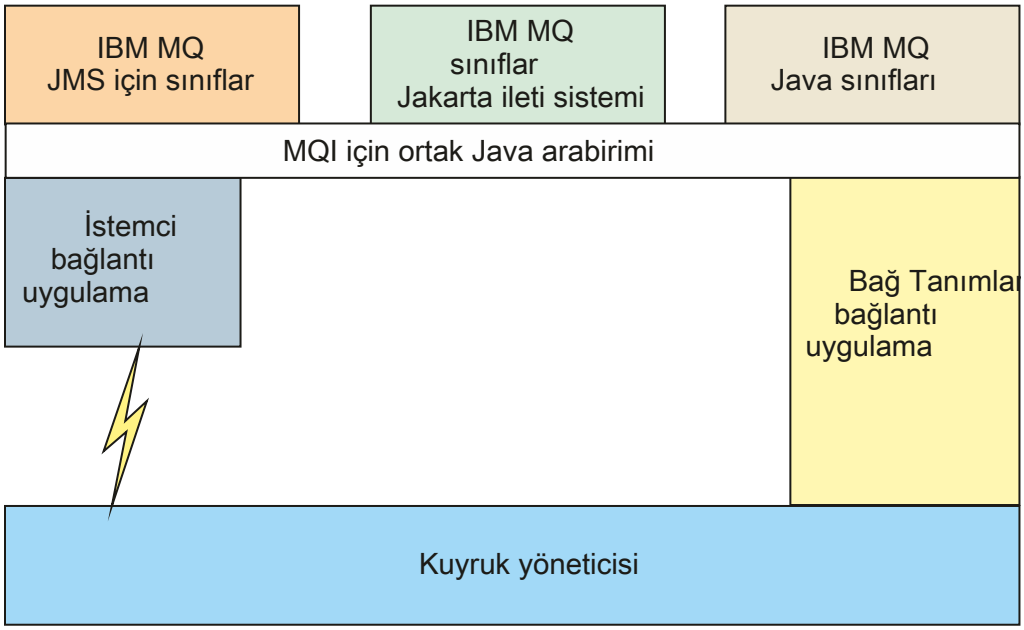
Gelen iletişim durumunda, hedefe ulaşan bir ileti bir MDB ' ye teslim edilir ve daha sonra iletiyi işler.

Java EE JM 3.0 ve Jakarta EE uygulamaları, iletileri zamanuyumsuz olarak işlemek için MDB ' leri kullanır. MDB, JMS ileti dinleyicisi olarak işlev görür ve bir iletinin nasıl işleneceğini tanımlayan bir onMessage() yöntemi tarafından gerçekleştirilir. Bir uygulama sunucusunun EJB taşıyıcısına bir MDB konuşlandırıldı. MDB 'nin yapılandırılma şekli, kullandığınız uygulama sunucusuna bağlıdır, ancak yapılandırma bilgileri hangi kuyruk yöneticisine bağlanacağınızı, kuyruk yöneticisine nasıl bağlanacağınızı, iletilerin izleneceği hedefi ve MDB' nin işlemsel davranışını belirtmelidir. Bu bilgiler daha sonra EJB taşıyıcısı tarafından kullanılır. MDB 'nin seçim ölçütlerini karşılayan bir ileti belirtilen hedefe geldiğinde, EJB taşıyıcısı iletiyi kuyruk yöneticisinden almak için IBM MQ classes for JMS ya da IBM MQ classes for Jakarta Messaging değerini kullanır ve daha sonra, onMessage() yöntemini çağırarak iletiyi MDB' ye teslim eder.

IBM MQ classes for Java ile ilişki

IBM MQ classes for Java, IBM MQ classes for Jakarta Messaging ve IBM MQ classes for JMS , MQI için ortak bir Java arabirimi kullanan eşlerdir.

Şekil 55 sayfa 153 , IBM MQ classes for JMS, IBM MQ classes for Jakarta Messaging ve IBM MQ classes for Javaarasındaki ilişkiyi gösterir.



Şekil 55. IBM MQ classes for JMS, IBM MQ classes for Jakarta Messaging ve IBM MQ classes for Java arasındaki ilişki

Genel olarak, Java programları IBM MQ - IBM MQ classes for Java, IBM MQ classes for Jakarta Messaging ya da IBM MQ classes for JMSile arabirim sağlamak için yalnızca bir arabirim kullanmalıdır. Tek bir kural dışı durum dışında, arabirimlerin karışık kullanılması desteklenmez. IBM WebSphere MQ 7.0'den önceki yayınlarla uyumluluğu sağlamak için, Java içinde yazılan kanal çıkış sınıfları, kanal çıkış sınıfları IBM MQ classes for JMS' den çağrılsa da IBM MQ classes for Java arabirimlerini kullanmaya devam edebilir. Ancak, IBM MQ classes for Java arabirimlerinin kullanılması, uygulamalarınızın aşağıdakilerden birine bağlı olduğu anlamına gelir:

- JMS 2.0 IBM MQ classes for Java JAR dosyası, `com.ibm.mq.jar`. Sınıf yolunuzda `com.ibm.mq.jar` olmasını istemiyorsanız, bunun yerine `com.ibm.mq.exits` paketindeki arabirim kümesini kullanabilirsiniz.
- V 9.3.0 V 9.3.0 JM 3.0 IBM MQ classes for Jakarta Messagingile birlikte çalışırken `com.ibm.mq.jakarta.client.jar`kullanımı.

İlgili kavramlar

V 9.3.0 V 9.3.0 Jakarta Messaging için IBM MQ sınıflarını neden kullanmalıyım?
JMS için neden IBM MQ sınıflarını kullanmalıyım?
Neden Javaiçin IBM MQ sınıflarını kullanmalıyım?

IBM MQ ileti alışverişi sağlayıcısı

IBM MQ ileti alışverişi sağlayıcısının üç çalışma kipi vardır: Normal kip, kısıtlamalar içeren normal kip ve geçiş kipi.

IBM MQ ileti alışverişi sağlayıcısının üç işletim kipi vardır:

- IBM MQ ileti alışverişi sağlayıcısı normal kipi
- Kısıtlamalar içeren IBM MQ ileti alışverişi sağlayıcısı normal kipi
- IBM MQ ileti alışverişi sağlayıcısı geçiş kipi

IBM MQ ileti alışverişi sağlayıcısı normal kipi, JMS' yi gerçekleştirmek için IBM MQ kuyruk yöneticisinin tüm özelliklerini kullanır. Bu kip, JMS 2.0 V 9.3.0 V 9.3.0 JM 3.0 ya da Jakarta Messaging 3.0 API ve işlevselliğini kullanmak üzere eniyilenmiştir.

Eğer:

- İstemci, **ConnectionFactory**üzerinde 6 sağlayıcı sürümünü belirtir; istemci, IBM WebSphere MQ 6.0ile sağlanan istemciyle uyumlu bir şekilde davranır. Yalnızca JMS 1.1 ve JMS 2 arabirimleri desteklenir, ancak paylaşılan abonelikler, teslim gecikmesi ve zamanuyumsuz gönderme gibi bazı JMS 2 işlevleri devre dışı bırakılır. Bağlantı paylaşımı yok.
- İstemci, **ConnectionFactory**üzerinde 7 sağlayıcı sürümünü belirtir; hem JMS 1.1 , hem de JMS 2 arabirimleri tam olarak desteklenir.
- Sağlayıcı sürümü belirtilmedi, sağlayıcı sürümü 7 ile bağlantı kurma girişiminde bulunuldu. Bu başarısız olursa, sağlayıcı sürüm 6 ile başka bir girişimde bulunmaya devam edin.

IBM Integration Bus ' e IBM MQ Enterprise Transport kullanarak bağlanmak istiyorsanız geçiş kipini kullanın. IBM MQ Real-Time Transport özelliğini kullanırsanız, bağlantı üreticisi nesnesinde özellikleri belirtik olarak seçmiş olduğunuz için geçiş kipi otomatik olarak seçilir. IBM Integration Bus ile IBM MQ Enterprise Transport arasındaki bağlantı, 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 tarafından kullanılan kavramların bazıları, z/OS platformu için benzersizdir. Örneğin, günlüğe kaydetme mekanizması, depolama yönetimi teknikleri, kurtarma atma birimi ve kuyruk paylaşım grupları yalnızca IBM MQ for z/OSile sağlanır. Bu konuyu, bu kavramlara ilişkin daha fazla bilgiye giriş olarak kullanın.

Kavramlar, IBM MQ for z/OS' in kullandığı nesnelere ilişkin genel bakışı içerir:

- Kuyruk yöneticisi
- Kanal başlatıcı
- Paylaşılan kuyruklar ve kuyruk paylaşım grupları
- Grup içi kuyruğa alma

Aşağıdaki konular, aşağıdakiler de dahil olmak üzere, gereksinim duyduğunuz çeşitli yordamları da kapsar:

- z/OS üzerindeki sistem tanımları
- Depolama yönetimi
- Kurtarma ve yeniden başlatma
- IBM MQ for z/OS içindeki güvenlik kavramları

İlgili kavramlar

“z/OS üzerindeki kuyruk yöneticisi” sayfa 155
Uygulama programlarınızın z/OS sisteminizde IBM MQ komutunu kullanmasına izin vermeden önce, IBM MQ for z/OS ürününü kurmanız ve bir kuyruk yöneticisi başlatmanız gerekir. Kuyruk yöneticisi, IBM MQtarafından kullanılan kaynak kümesine sahip olur ve bunları yönetir.

“z/OS üzerindeki kanal başlatıcısı” sayfa 156
Kanal başlatıcı, IBM MQ dağıtımlı kuyruğa alma özelliğini etkinleştiren kaynakları sağlar ve yönetir. IBM MQ , bir kuyruk yöneticisinden diğerine ileti göndermek için *Message Channel Agents* (MCA) olanağını kullanır.

“IBM MQ for z/OS yönetimi için koşullar ve görevler” sayfa 158
Bu konuyu, IBM MQ for z/OS' e özgü olan terminolojiye ve görevlere giriş olarak kullanın.

“Paylaşılan kuyruklar ve kuyruk paylaşım grupları” sayfa 160
IBM MQ kaynaklarının yüksek kullanılabilirliğini uygulamak için paylaşılan kuyrukları ve kuyruk paylaşım gruplarını kullanabilirsiniz. Paylaşılan kuyruklar ve kuyruk paylaşım grupları, z/OS platformunda IBM MQ for z/OS için benzersiz işlevlerdir.

“Grup içi kuyruğa alma” sayfa 204

Bu bölümde, z/OS platformuna özgü bir IBM MQ for z/OS işlevi olan grup içi kuyruğa alma açıklanmaktadır. Bu işlev yalnızca, bir kuyruk paylaşım grubuna tanımlanan kuyruk yöneticileri için kullanılabilir.

“z/OS üzerinde depolama yönetimi” sayfa 217
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 kullandığına ilişkin daha fazla ayrıntı sağlar.

“oturum açmaIBM MQ for z/OS” sayfa 221
IBM MQ veri değişikliklerinin ve önemli olayların *günlüklerini* , bunlar oluştuğunda tutar. Bu günlükler, gerektiğinde verileri önceki bir duruma kurtarmak için kullanılabilir.

“z/OS üzerinde kurtarma ve yeniden başlatma” sayfa 242
Yeniden başlatma ve kurtarma için IBM MQ for z/OS özellikleri hakkında bilgi edinmek üzere bu konudaki bağlantıları kullanın.

“IBM MQ for z/OS içindeki güvenlik kavramları” sayfa 258
IBM MQiçin güvenliğin önemini ve sisteminizde yeterli güvenlik ayarlarına sahip olmamanın etkilerini anlamak için bu konuyu kullanın.

“z/OS üzerinde kullanılabilirlik” sayfa 265
IBM MQ for z/OS , yüksek kullanılabilirlik için birçok özelliğe sahiptir. Bu konuda, kullanılabilirlikle ilgili dikkat edilmesi gereken bazı noktalar açıklanmaktadır.

“z/OS üzerinde kurtarma birimi yok etme” sayfa 269
Bazı işlemsel uygulamalar, kuyruk paylaşım grubundaki (QSG) bir kuyruk yöneticisine bağlandığında, kuyruk yöneticisi adı yerine QSG adını belirterek, QMGR yerine bir kurtarma atma birimi kullanabilir. Bu, QSG ' de aynı kuyruk yöneticisine yeniden bağlanma gereksinimini kaldırarak işlem kurtarmanın daha esnek ve güçlü olmasını sağlar.

İlgili başvurular
“z/OS üzerinde sistem tanımı” sayfa 232
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/OS üzerinde izleme ve istatistik” sayfa 268
IBM MQ for z/OS ' in kuyruk yöneticisini izlemek ve istatistikleri toplamak için bir olanaklar kümesi vardır.

z/OS z/OS üzerindeki kuyruk yöneticisi

Uygulama programlarınızın z/OS sisteminizde IBM MQ komutunu kullanmasına izin vermeden önce, IBM MQ for z/OS ürününü kurmanız ve bir kuyruk yöneticisi başlatmanız gerekir. Kuyruk yöneticisi, IBM MQtarafından kullanılan kaynak kümesine sahip olur ve bunları yönetir.

Kuyruk yöneticisi

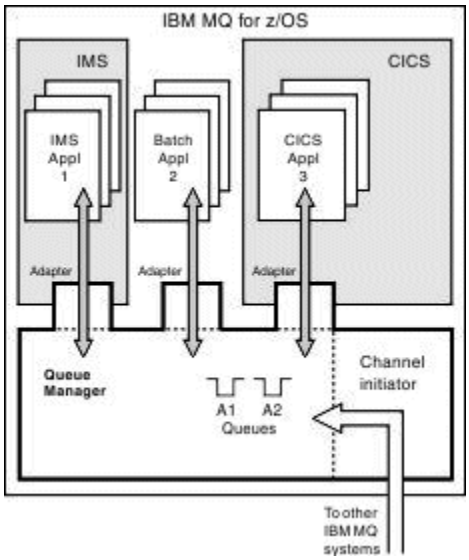
Kuyruk yöneticisi , uygulamalara ileti sistemi hizmetleri sağlayan bir programdır. İleti Kuyruğu Arabirimi 'ni (MQI) kullanan uygulamalar, kuyruklara ileti yerleştirebilir ve kuyruklardan ileti alabilir. Kuyruk yöneticisi, iletilerin doğru kuyruğa gönderilmesini ya da başka bir kuyruk yöneticisine yöneltilmesini sağlar. Kuyruk yöneticisi, hem ona verilen MQI çağrılarını, hem de bu çağrıya gönderilen komutları (herhangi bir kaynaktan) işler. Kuyruk yöneticisi, her çağrı ya da komut için uygun tamamlama kodlarını oluşturur.

Kuyruk yöneticisi tarafından yönetilen kaynaklar şunlardır:

- IBM MQ nesne tanımlamalarını ve ileti verilerini içeren sayfa kümeleri
- Kuyruk yöneticisi hatası durumunda iletileri ve nesneleri kurtarmak için kullanılan günlükler
- İşlemci depolama alanı
- Farklı uygulama ortamlarının (CICS, IMSve Toplu İş) IBM MQ API 'sine erişebileceği bağlantılar
- z/OS sisteminizde IBM MQ ile diğer sistemler arasında iletişim kurulmasını sağlayan IBM MQ kanal başlatıcısı

Kuyruk yöneticisinin bir adı var ve uygulamalar bu adı kullanarak bu ada bağlanabilir.

Şekil 56 sayfa 156 , farklı uygulama ortamlarıyla bağlantıları gösteren bir kuyruk yöneticisini ve kanal başlatıcısını gösterir.



Şekil 56. IBM MQ for z/OS ürününe genel bakış

z/OS üzerindeki kuyruk yöneticisi altsistemi

z/OSişletim sisteminde IBM MQ , IPL zamanında başlatılan bir z/OS altsistemi olarak çalışır. Altsistemde, kuyruk yöneticisi, günlüklere ilişkin bilgileri içeren ve nesne tanımlamalarını ve ileti verilerini (sayfa kümeleri) bulunduran z/OS veri kümelerini belirten bir JCL yordamı yürütülerek başlatılır. Altsistem ve kuyruk yöneticisi en çok dört karakterden oluşan aynı ada sahiptir. Ağınızdaki tüm kuyruk yöneticileri, farklı sistemlerde, sistem sistemlerinde ya da altyapılarda olsalar bile benzersiz adlara sahip olmalıdır.

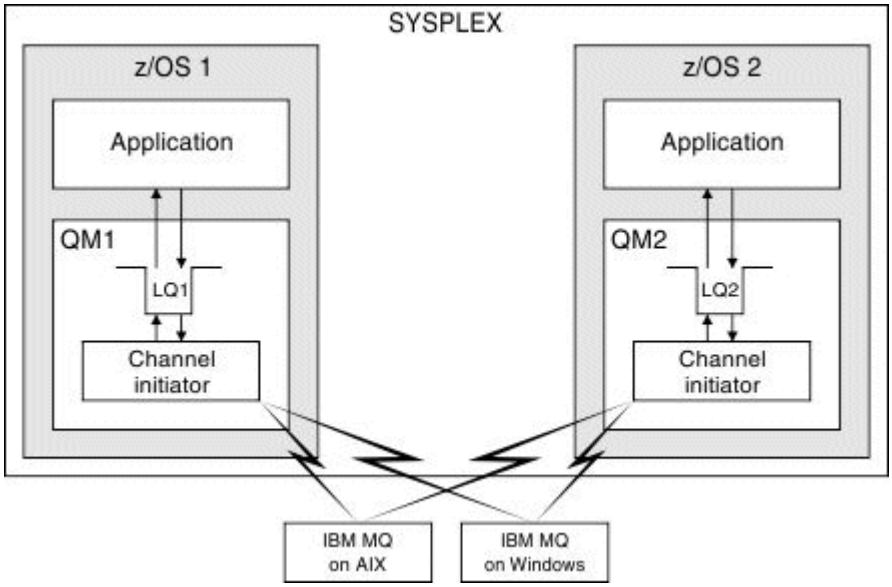
z/OS z/OS üzerindeki kanal başlatıcısı

Kanal başlatıcı, IBM MQ dağıtımlı kuyruğa alma özelliğini etkinleştiren kaynakları sağlar ve yönetir. IBM MQ , bir kuyruk yöneticisinden diğerine ileti göndermek için *Message Channel Agents* (MCA) olanağını kullanır.

A kuyruk yöneticisinden B kuyruk yöneticisine ileti göndermek için, A kuyruk yöneticisinde MCA ' nın *gönderilmesi* , B kuyruk yöneticisine bir iletişim bağlantısı oluşturmalıdır. İletişim bağlantısından ileti almak için kuyruk yöneticisi B ' de *alan* MCA başlatılmalıdır. Gönderen MCA, iletişim bağlantısı ve alan MCA ' dan oluşan bu tek yönlü yol, *kanal*olarak bilinir. Gönderen MCA, iletileri bir iletim kuyruğundan alır ve bunları alan MCA ' ya bir kanaldan gönderir. Alan MCA iletileri alır ve hedef kuyruklara yerleştirir.

IBM MQ for z/OSiçinde, tüm gönderen ve alan MCA ' lar kanal başlatıcı içinde çalışır (kanal başlatıcı *taşıyıcı* olarak da bilinir). Kanal başlatıcı, kuyruk yöneticisinin denetimi altında bir z/OS adres alanı olarak çalışır. Bir kuyruk yöneticisine bağlı tek bir kanal başlatıcısı olabilir ve kuyruk yöneticisiyle aynı z/OS görüntüsü içinde çalıştırılır. Kanal başlatıcı içinde eşzamanlı olarak çalışan binlerce MCA işlemi olabilir.

Şekil 57 sayfa 157 içinde bir sistem şebekesindeki iki kuyruk yöneticisi gösterilir. Her kuyruk yöneticisinin bir kanal başlatıcısı ve bir yerel kuyruğu vardır. AIX ve Windows üzerinde kuyruk yöneticileri tarafından gönderilen iletiler, bir uygulama tarafından alındıkları yerel kuyruğa yerleştirilir. Yanıt iletileri benzer bir rota tarafından döndürülür.



Şekil 57. Kuyruk yöneticileri arasındaki iletişim

Kanal başlatıcısı, kanalların yönetimiyle ilgili diğer süreçleri de içerir. Bu işlemler şunlardır:

Dinleyiciler
Bu işlemler, TCP gibi bir iletişim altsistemindeki gelen kanal isteklerini dinler ve bir gelen istek alındığında adı belirlenmiş bir MCA ' yi başlatır.

Gözetmen
Bu, kanal başlatıcı adres alanını yönetir; örneğin, bir hatadan sonra kanalların yeniden başlatılmasından sorumludur.

Ad sunucusu
Bu, TCP adlarını adreslere çözümlemek için kullanılır.

TLS görevleri
Bunlar, şifreleme ve şifre çözme gerçekleştirmek ve sertifika iptal listelerini denetlemek için kullanılır.

z/OS Kanal başlatıcı için SMF kayıtları

Kanal başlatıcı (CHINIT), görevlere ve kanallara ilişkin bilgilerle SMF istatistik kayıtları ve muhasebe kayıtları üretebilir.

CHINIT, aşağıdaki bilgi tipleriyle SMF istatistik kayıtları ve muhasebe kayıtları üretebilir:

- Görevler: dağıtıcı, bağdaştırıcı, Etki Alanı Ad Sunucusu (DNS) ve SSL. Bu görevler, CHINIT istatistikleri olarak adlandırılan bilgileri oluşturur.
- Kanallar: DIS CHSTATUS komutuyla sağlanana benzer muhasebe bilgileri sağlar. Buna kanal muhasebesi denir.

IBM MQ for Multiplatforms , SYSTEM.ADMIN.STATISTICS.QUEUE. İstatistik bilgilerinin IBM MQ for Multiplatformsüzerinde nasıl kaydedildiğine ilişkin ek bilgi için Kanal istatistikleri ileti verileri konusuna bakın.

İstatistik Verileri

Aşağıdaki bilgileri öğrenmek için bu bilgileri kullanabilirsiniz:

- SSL TCB sayısı ve bu görevler tarafından kullanılan CPU miktarı gibi CHINIT görevlerinden daha fazlasına gerek duyup duymadığınızı belirler.
- Bu görevlere ilişkin isteklerin ortalama süresi.

- DNS ve SSL görevleri için, aralıktaki en uzun süre isteği ve günün saati. Günün bu saatinde kanalla karşılaşabileceğiniz sorunlarla ilişkilendirme yapabilirsiniz.

Muhasebe verileri

Kanal kullanımını izlemek ve aşağıdaki bilgileri öğrenmek için bu bilgileri kullanabilirsiniz:

- En yüksek veri çıkışına sahip kanallar.
- İletilerin gönderilme hızı ve MB/saniye cinsinden veri gönderme hızı.
- Ulaşılan toplu iş boyutu. Ulaşılan toplu iş boyutu kanal için belirtilen toplu iş boyutuna yakınsa, kanal ileti gönderme sınırına yakın olabilir.

Muhasebe izleme ve istatistik izleme verilerinin toplanmasını denetlemek için START TRACE ve STOP TRACE komutlarını kullanabilirsiniz. Kanalların SMF verileri üretip üretmediğini denetlemek için kanal ve kuyruk yöneticisindeki STATCHL ve STATACLS seçeneklerini kullanabilirsiniz.

z/OS IBM MQ for z/OS yönetimi için koşullar ve görevler

Bu konuyu, IBM MQ for z/OS' e özgü olan terminolojiye ve görevlere giriş olarak kullanın.

IBM MQ for z/OS yönetimi için gerekli olan bazı koşullar ve görevler, z/OS platformuna özgüdür. Aşağıdaki listede bu terimlerin ve görevlerin bazıları yer almaktadır.

- Paylaşılan kuyruklar
- Sayfa kümeleri ve arabellek havuzları
- Günlük Kaydı
- Kuyruk yöneticisi ortamının uyarlanması
- Yeniden başlatma ve kurtarma
- Durumu
- Kullanılabilirlik
- Nesnelerin işlenmesi
- İzleme ve İstatistik
- Uygulama ortamları

Paylaşılan kuyruklar

Kuyruklar *paylaşılmayan*, yalnızca bir kuyruk yöneticisine ait ve bu yöneticiye erişilebilir ya da bir *kuyruk paylaşım grubuna*ait *paylaşılan*olabilir. Kuyruk paylaşım grubu, aynı IBM MQ nesne tanımlarına ve ileti verilerine eşzamanlı olarak erişebilen tek bir z/OS sistem şebekesi içinde çalışan kuyruk yöneticilerinden oluşur. Bir kuyruk paylaşım grubu içinde, paylaşılabilir nesne tanımlamaları paylaşılan bir Db2 veritabanında saklanır. Paylaşılan kuyruk iletileri bir ya da daha çok bağlaşım olanağı yapılarının (CF yapıları) içinde tutulur. İleti verileri doğrudan yapıda saklanamayacak kadar büyükse (63 KB ' den büyük) ya da ileti, kuruluş tanımlı kuralların boşaltmak üzere seçeceği kadar büyükse, ileti denetimi bilgileri hala bağlaşım olanağı girişinde saklanır, ancak ileti verileri paylaşılan bir ileti veri kümesine (SMDS) ya da paylaşılan bir Db2 veritabanına boşaltılır. Paylaşılan ileti veri kümeleri, paylaşılan Db2 veritabanı ve bağlaşım olanağı yapıları, gruptaki tüm kuyruk yöneticileri tarafından ortaklaşa yönetilen kaynaklardır.

Sayfa kümeleri ve arabellek havuzları

Bir ileti paylaşılmayan bir kuyruğa konduğunda, kuyruk yöneticisi verileri bir sayfa kümesine, sonraki bir işlem aynı kuyruktan ileti aldığında alınabilecek şekilde saklar. İleti kuyruktan kaldırılırsa, verileri tutan sayfa kümesindeki alan daha sonra yeniden kullanılmak üzere serbest bırakılır. Kuyrukta tutulan iletilerin

sayısı arttıkça, sayfa kümesinde kullanılan alan miktarı artar ve kuyruktaki ileti sayısı azaldıkça, sayfa kümesinde kullanılan alan azalır.

Sayfa kümelerine veri yazma ve sayfa kümelerinden veri okuma performans maliyetini azaltmak için kuyruk yöneticisi, güncellemeleri işlemci depolamasına arabellekler. Sayfa kümesi erişimini arabelleğe almak için kullanılan bellek miktarı, *arabellek havuzları*adı verilen IBM MQ nesneler aracılığıyla denetlenir.

Sayfa kümeleri ve arabellek havuzlarıyla ilgili daha fazla bilgi için bkz. Storage management.

Günlük Kaydı

Sayfa kümelerinde tutulan nesnelerde ve kalıcı iletilerdeki işlemlerde yapılan değişiklikler günlük kaydı olarak kaydedilir. Bu günlük kayıtları, *etkin günlük*adı verilen bir günlük veri kümesine yazılır. Etkin günlük veri kümesinin adı ve boyutu, *bootstrap veri kümesi* (BSDS) adı verilen bir veri kümesinde tutulur.

Etkin günlük veri kümesi dolduğunda, kuyruk yöneticisi başka bir günlük veri kümesine geçer; böylece günlük kaydı devam edebilir ve tam etkin günlük veri kümesinin içeriğini bir *arşiv günlüğü* veri kümesine kopyalar. Arşiv günlüğü veri kümesinin adı da içinde olmak üzere, bu işlemlerle ilgili bilgiler önyükleme veri kümesinde tutulur. Kavramsal olarak, kuyruk yöneticisinin döngüye girdiği bir etkin günlük veri kümesi halkası vardır; etkin bir günlük doldurulduğunda, günlük verileri bir arşiv günlüğüne boşaltılır ve etkin günlük veri kümesi yeniden kullanılabilir.

Günlük ve önyükleme veri kümeleriyle ilgili daha fazla bilgi için bkz. “oturum açmaIBM MQ for z/OS” sayfa 221.

Kuyruk yöneticisi ortamının uyarlanması

Kuyruk yöneticisi başlatıldığında, kuyruk yöneticisinin nasıl çalıştığını denetleyen bir başlatma parametreleri kümesi okunur. Buna ek olarak, IBM MQ komutlarını içeren veri kümeleri okunur ve içerdikleri komutlar yürütülür. Genellikle bu veri kümeleri, IBM MQ ' in çalışması için gerekli sistem nesnelerinin tanımlarını içerir ve bunları işletim ortamınız için gerekli IBM MQ nesnelerini tanımlamak ya da kullanıma hazırlamak üzere uyarlayabilirsiniz. Bu veri kümeleri okunduğunda, bunların tanımladığı nesneler bir sayfa kümesinde ya da Db2içinde saklanır.

Kullanıma hazırlama parametreleri ve sistem nesneleri hakkında daha fazla bilgi için bkz. “z/OS üzerinde sistem tanımı” sayfa 232.

Kurtarma ve yeniden başlatma

IBM MQ' in çalışması sırasında, işlemci depolamasında henüz sayfa kümesine yazılmamış değişiklikler olabilir. Bu değişiklikler, kuyruk yöneticisi içindeki bir arka plan görevi tarafından en son kullanılan sayfa kümesine yazılır.

Kuyruk yöneticisi olağandışı bir şekilde sona ererse, kalıcı ileti verileri günlük kayıtlarında tutulduğundan, kuyruk yöneticisini yeniden başlatmanın kurtarma aşaması, kayıp sayfa kümesi değişikliklerini 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, o kuyruktaki kalıcı iletiler yalnızca bağlaşım olanağı yapınızı yedeklediyseniz kurtarılabilir.

Kurtarma ve yeniden başlatma hakkında daha fazla bilgi için bkz. “z/OS üzerinde kurtarma ve yeniden başlatma” sayfa 242.

Durumu

Security Server (önceki adıyla RACF) gibi bir dış güvenlik yöneticisi kullanabilirsiniz. IBM MQ ' in sahip olduğu ve yönettiği kaynakları yetkisiz kullanıcılar tarafından erişime karşı korumak için. Kanal güvenliği

için TLS (Transport Layer Security; İletim Katmanı Güvenliği) de kullanabilirsiniz. TLS, IBM MQ ürününün bir parçası olarak dahil edilir.

IBM MQ güvenliği hakkında daha fazla bilgi için bkz. “IBM MQ for z/OS içindeki güvenlik kavramları” sayfa 258.

Kullanılabilirlik

Kuyruk yöneticisi ya da iletişim altsistemi arızası durumunda sistem kullanılabilirliğini artırmak için tasarlanmış birkaç IBM MQ özelliği vardır. Bu özelliklerle ilgili daha fazla bilgi için bkz. “z/OS üzerinde kullanılabilirlik” sayfa 265.

Nesneleri işleme

Kuyruk yöneticisi çalışırken, IBM MQ nesnelerini bir z/OS konsol arabirimi aracılığıyla ya da TSO altında ISPF hizmetlerini kullanan bir denetim yardımcı programı aracılığıyla işleyebilirsiniz. Her iki mekanizma da IBM MQ nesnelerini tanımlamanızı, değiştirmenizi ya da silmenizi sağlar. Ayrıca, çeşitli IBM MQ ve kuyruk yöneticisi işlevlerinin durumunu denetleyebilir ve görüntüleyebilirsiniz.

Bu olanaklarla ilgili daha fazla bilgi için bkz. IBM MQ for z/OSüzerinde MQSC ve PCF komutlarını yayınlayabileceğiniz kaynaklar.

IBM MQ nesnelerini, kuyruklar, kuyruk yöneticileri ve diğer nesnelerle görsel çalışma yöntemi sağlayan bir grafik kullanıcı arabirimi olan IBM MQ Gezgini 'ni kullanarak da işleyebilirsiniz.

İzleme ve İstatistik

Kuyruk yöneticilerinizi ve kanal başlatıcılarınızı izlemek için çeşitli olanaklar vardır. Ayrıca, performans değerlendirmesi ve muhasebe amacıyla da istatistik toplayabilirsiniz.

Bu olanaklar hakkında daha fazla bilgi için bkz. “IBM MQ for z/OS üzerinde izleme ve istatistik” sayfa 268.

Uygulama ortamları

Kuyruk yöneticisi başlatıldığında, uygulamalar kuyruk yöneticisine bağlanabilir ve IBM MQ API 'sini kullanmaya başlayabilir. Bunlar CICS, IMS, Toplu İş ya da WebSphere Application Server uygulamaları olabilir. IBM MQ uygulamaları, CICS ve IMS köprülerini kullanarak IBM MQ' in farkında olmayan CICS ve IMS sistemlerindeki uygulamalara da erişebilir.

Bu olanaklar hakkında daha fazla bilgi için bkz. “IBM MQ ve diğer z/OS ürünleri” sayfa 272.

IBM MQ uygulamalarının yazılmasına ilişkin bilgi için aşağıdaki belgelere bakın:

- Uygulama geliştirilmesi
- C++ kullanılması
- IBM MQ classes for Java Olanağının Kullanılması

z/OS Paylaşılan kuyruklar ve kuyruk paylaşım grupları

IBM MQ kaynaklarının yüksek kullanılabilirliğini uygulamak için paylaşılan kuyrukları ve kuyruk paylaşım gruplarını kullanabilirsiniz. Paylaşılan kuyruklar ve kuyruk paylaşım grupları, z/OS platformunda IBM MQ for z/OS için benzersiz işlevlerdir.

Bu bölümde öznitelikler ve yararlar açıklanır ve birden çok kuyruk yöneticisinin aynı kuyrukları ve bu kuyruklardaki iletileri nasıl paylaşabileceğine ilişkin bilgiler yer alır.

Paylaşılan kuyruk nedir?

Paylaşılan kuyruk, yerel kuyruk tipidir. Bu kuyruktaki iletilere, sistem şebekesinde bulunan bir ya da daha çok kuyruk yöneticisi erişebilir.

Kuyruk paylaşım grubu

Aynı paylaşılan kuyruk kümesine erişebilen kuyruk yöneticileri, *kuyruk paylaşım grubu*adı verilen bir grup oluşturur.

Herhangi bir kuyruk yöneticisi iletilere erişebilir

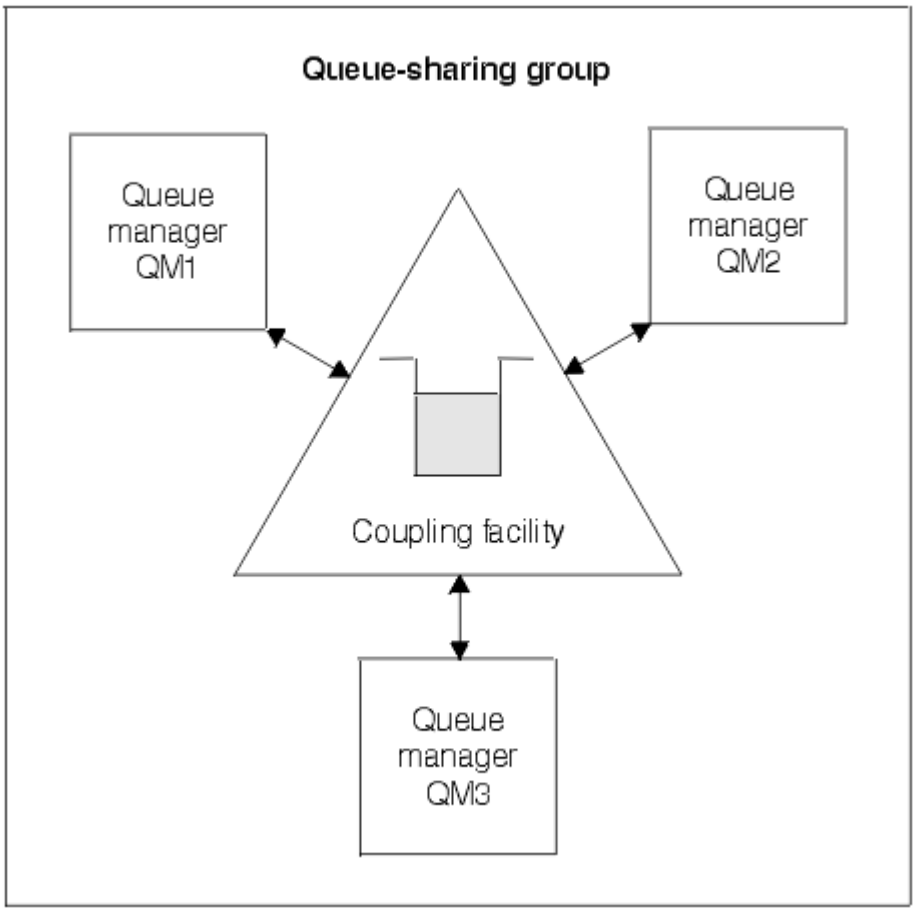
Kuyruk paylaşım grubundaki herhangi bir kuyruk yöneticisi, paylaşılan bir kuyruğa erişebilir. Bu, bir kuyruk yöneticisinde paylaşılan bir kuyruğa ileti yerleştirebileceğiniz ve aynı iletiyi farklı bir kuyruk yöneticisinden kuyruktan alabileceğiniz anlamına gelir. Bu, kuyruk yöneticileri arasında kanalların etkin olmasını gerektirmeyen bir kuyruk paylaşım grubu içinde iletişim için hızlı bir mekanizma sağlar.

IBM MQ , iletilerin Db2 ya da paylaşılan ileti veri kümesine (SMDS) boşaltılmasını destekler. Herhangi bir boyuttaki iletilerin boşaltımı yapılandırılabilir.

Şekil 58 sayfa 161 içinde, bir kuyruk paylaşım grubu oluşturan üç kuyruk yöneticisi ve bir bağlaşım olanağı gösterilir. Her üç kuyruk yöneticisi de bağlaşım olanağındaki paylaşılan kuyruğa erişebilir.

Bir uygulama, kuyruk paylaşım grubu içindeki herhangi bir kuyruk yöneticisine bağlanabilir. Kuyruk paylaşım grubundaki tüm kuyruk yöneticileri tüm paylaşılan kuyruklara erişebildiği için, uygulama belirli bir kuyruk yöneticisinin kullanılabilirliğine bağlı değildir; kuyruk paylaşım grubundaki herhangi bir kuyruk yöneticisi kuyruğa hizmet verebilir.

Kuyruk yöneticilerinden birinin bir sorunu varsa, kuyruk paylaşım grubundaki diğer tüm kuyruk yöneticileri kuyruğu işlemeye devam edebildiğinden, bu daha fazla kullanılabilirlik sağlar.



Şekil 58. Bir kuyruk paylaşım grubu

Kuyruk tanımlaması tüm kuyruk yöneticileri tarafından paylaşılıyor

Paylaşılan kuyruk tanımlamaları Db2 veritabanı çizelgesi OBJ_B_QUEUE içinde saklanır. Bu nedenle, kuyruğu yalnızca bir kez tanımlamanız ve ardından kuyruk paylaşım grubundaki tüm kuyruk yöneticilerine erişmeniz gerekir. Bu, yapılacak daha az tanımlama olduğu anlamına gelir.

Buna karşılık, paylaşılmayan bir kuyruk tanımı, kuyruğun sahibi olan kuyruk yöneticisinin sayfa kümesi sıfırında (Sayfa kümeleri içinde açıklandığı gibi) saklanır.

Bu adı taşıyan bir kuyruk, tanımlayan kuyruk yöneticisinin sayfa kümelerinde önceden tanımlandıysa, paylaşılan kuyruk tanımlayamazsınız. Benzer şekilde, aynı ada sahip bir paylaşılan kuyruk varsa, kuyruk yöneticisi sayfasında bir kuyruğun yerel sürümünü tanımlayamazsınız.

Kuyruk paylaşım grubu nedir?

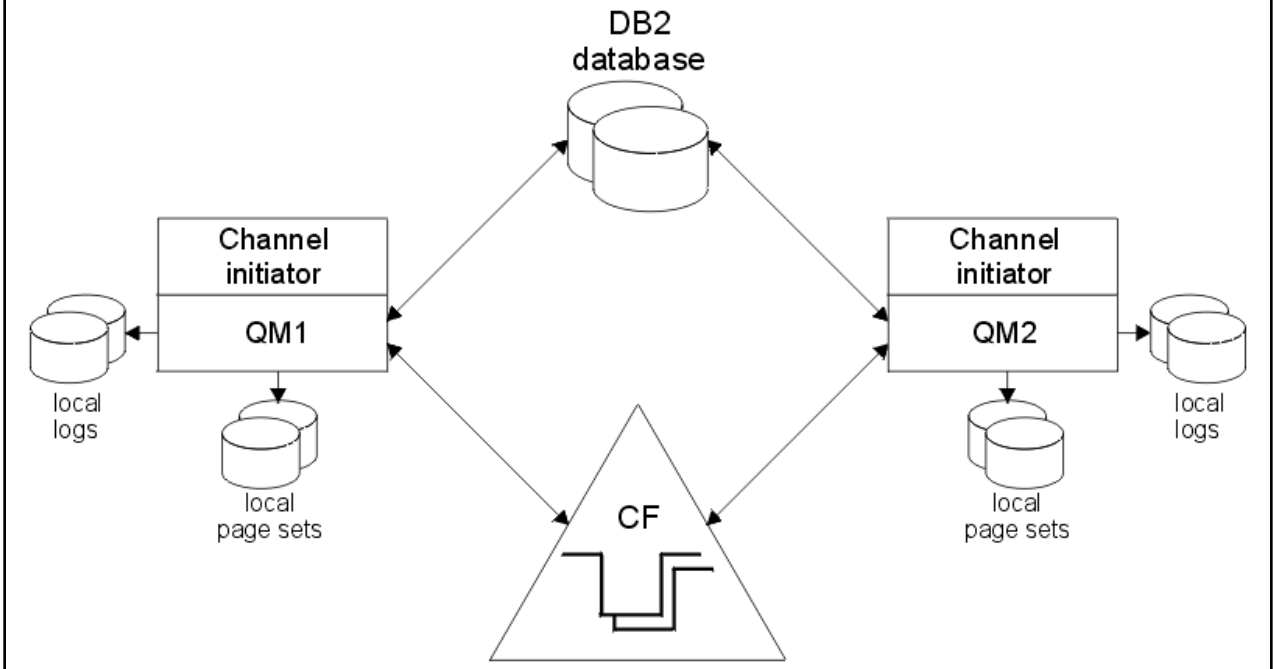
Aynı paylaşılan kuyruklara erişebilen kuyruk yöneticileri grubuna kuyruk paylaşım grubu denir. Kuyruk paylaşım grubunun her üyesinin aynı paylaşılan kuyruk kümesine erişimi vardır.

Kuyruk paylaşım gruplarının adı en çok dört karakterdir. Ad ağınızda benzersiz olmalı ve herhangi bir kuyruk yöneticisi adından farklı olmalıdır.

Şekil 59 sayfa 162 içinde iki kuyruk yöneticisi içeren bir kuyruk paylaşım grubu gösterilir. 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 erişebilmesi için Db2 sistemlerinin tümü aynı Db2 veri paylaşım grubunda olmalıdır. Bunlar, yalnızca bir kez tanımlanan IBM MQ nesnesi (örneğin, kuyruklar ve kanallar) ve daha sonra, gruptaki herhangi bir kuyruk yöneticisi bunları kullanabilir. Bunlara *genel* tanımlar denir ve Özel ve genel tanımlarkısmında açıklanır.

Belirli bir veri paylaşım grubuna birden çok kuyruk paylaşım grubu başvurabilir. Db2 altsisteminin adını ve başlatma sırasında IBM MQ sistem parametrelerinde bir kuyruk yöneticisinin hangi veri paylaşım grubunu kullanacağını belirtirsiniz.



Şekil 59. Bir kuyruk paylaşım grubundaki kuyruk yöneticilerinin bileşenleri

Bir kuyruk yöneticisi bir kuyruk paylaşım grubuna katıldığında, o grup için tanımlanan paylaşılan nesnelere erişimi olur ve bu kuyruk yöneticisini grup içinde yeni paylaşılan nesneler tanımlamak için kullanabilirsiniz. Grup içinde paylaşılan kuyruklar tanımlandıysa, bu paylaşılan kuyruklara ileti koymak ve bu kuyruklardan

ileti almak için bu kuyruk yöneticisini kullanabilirsiniz. Gruptaki herhangi bir kuyruk yöneticisi, paylaşılan bir kuyrukta tutulan iletileri alabilir.

Bir MQSC komutunu bir kez girebilir ve kuyruk paylaşım grubu içindeki tüm kuyruk yöneticilerine her kuyruk yöneticisinde ayrı ayrı girilmiş gibi yürütebilirsiniz. Bunun için *komut kapsamı* özniteliği kullanılır. Bu öznitelik, Komutların farklı kuyruk yöneticilerine yönlendirilmesikonusunda açıklanmıştır.

Bir kuyruk yöneticisi bir kuyruk paylaşım grubunun üyesi olarak çalıştığında, o kuyruk yöneticisine özel olarak tanımlanan IBM MQ nesneleri ile genel olarak tanımlanan ve kuyruk paylaşım grubundaki tüm kuyruk yöneticilerinin kullanımına sunulan IBM MQ nesneleri arasında ayrım yapılması gerekir. Bunun için *kuyruk paylaşım grubu yok etme* özniteliği kullanılır. Bu öznitelik, Özel ve genel tanımlamalarkısmında açıklanmaktadır.

Grubun herhangi bir yerindeki IBM MQ nesnelerine erişimi denetleyen tek bir güvenlik tanıtımları kümesi tanımlayabilirsiniz. Bu, tanımlamanız gereken profil sayısının büyük ölçüde azaldığı anlamına gelir.

Bir kuyruk yöneticisi yalnızca bir kuyruk paylaşım grubuna ait olabilir ve gruptaki tüm kuyruk yöneticileri aynı sistem şebekesinde olmalıdır. Başlatma sırasında, kuyruk yöneticisinin sistem parametrelerinde hangi kuyruk paylaşım grubuna ait olduğunu belirtirsiniz.

İlgili kavramlar

“Paylaşılan kuyruk iletileri nerede tutuluyor?” sayfa 163
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 kuyrukları kullanmanın avantajları” sayfa 179
Paylaşılan kuyruk, IBM MQ uygulamalarının ölçeklenebilir, yüksek düzeyde kullanılabilir olmasını ve iş yükü dengelemenin uygulanmasını sağlar.

“Dağıtılmış kuyruğa alma ve kuyruk paylaşım grupları” sayfa 198
Dağıtılmış kuyruğa alma ve kuyruk paylaşımı grupları, uygulama sistemlerinizin kullanılabilirliğini artırmak için kullanabileceğiniz iki tekniktir. Bu teknikler hakkında daha fazla bilgi edinmek için bu konuyu kullanın.

“Paylaşılan kuyruklar ile iş yükü dağıtımını etkileme” sayfa 202
Bir kuyruk paylaşım grubundaki paylaşılan kuyruklar ile iş yükü dağıtımını etkileyen etkenleri anlamak için bu konuyu kullanın.

İlgili başvurular

“Paylaşılan kuyruklar ve kuyruk paylaşım gruplarıyla ilgili daha fazla bilgi nereden bulunacak?” sayfa 203
IBM MQ for z/OS ' in paylaşılan kuyrukları ve kuyruk paylaşım gruplarını nasıl kullandığına ilişkin daha fazla bilgi edinmek için bu konudaki tabloyu kullanın.

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.

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 165 .

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 parametrelerini kullanabilirsiniz; böylece, her ileti en az bir bağlaşım olanağı girişi ve en az 768 baytlık veri gerektirir; giriş için 256 bayttan ve üstbilgi ve tanımlayıcının iki öğesi için 512 bayttan oluşur.

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 .

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.

Not: IBM MQ kullanırken bir bağlaşım olanağı yapısını şifreleyebilirsiniz. Daha fazla bilgi için bkz. Eşleme tesisi yapısı verilerinin şifrelenmesi .

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 165
Paylaşılan kuyruk iletisine ilişkin ileti verilerinin bir Db2 çizelgesinde ya da paylaşılan ileti veri kümesinde (SMDS) nerede saklanacağını seçebilirsiniz. İletinin büyüklüğüne ve bağlaşım olanağı yapısının (CF) yürürlükteki kullanımına bağlı olarak hangi iletilerin boşaltılabileceğini de seçebilirsiniz.

“Paylaşılan ileti verileri kümesi (SMDS) ortamınızın yönetilmesi” sayfa 167
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*

Paylaşılan kuyruk iletisine ilişkin ileti verilerinin bir Db2 çizelgesinde ya da paylaşılan ileti veri kümesinde (SMDS) nerede saklanacağını seçebilirsiniz. İletinin büyüklüğüne ve bağlaşım olanağı yapısının (CF) yürürlükteki kullanımına bağlı olarak hangi iletilerin boşaltılabileceğini de seçebilirsiniz.

Paylaşılan kuyruklara ilişkin ileti verileri, bağlaşım olanağından boşaltılabilir ve Db2 çizelgesinde ya da *paylaşılan ileti veri kümesi* (SMDS) adı verilen IBM MQ yönetilen veri kümesinde saklanabilir.

63 KB 'lik bağlaşım olanağı giriş boyutundan büyük iletiler için, ileti verilerinin bir SMDS' ye boşaltılmasının Db2' e boşaltılmasıyla karşılaştırıldığında önemli bir performans artışı olabilir.

Her paylaşılan kuyruk iletisi, bir bağlaşım olanağı yapısındaki bir liste girişi kullanılarak yönetilmeye devam eder, ancak ileti verileri SMDS ' ye boşaltıldığında, bağlaşım olanağı girişi yalnızca bazı denetim bilgilerini ve iletinin saklandığı ilgili disk bloklarına yapılan başvuruların bir listesini içerir. Bu mekanizmanın kullanılması, her ileti için gereken bağlaşım olanağı öğe saklama alanı miktarının, iletinin gerçek boyutunun yalnızca bir kısmı olduğu anlamına gelir.

Paylaşılan kuyruk iletilerinin nerede saklanacağını seçme

SMDS ya da Db2 paylaşılan ileti depolama alanı seçimi, **CFSTRUCT** tanımlamasındaki **OFFLOAD(SMDS|DB2)** parametresiyle denetlenir. **OFFLOAD(SMDS)** varsayılan değerdir.

Bu parametre, **CFSTRUCT** ' in **CFLEVEL(5)** ya da daha büyük bir değeri kullanmasını da gerektirir.

OFFLOAD parametresi yalnızca **CFLEVEL(5)** içinden geçerlidir. Daha fazla ayrıntı için bkz. DEFINE CFSTRUCT .

OFFLOAD(DB2) öncelikle geçiş amacıyla desteklenir.

Hangi paylaşılan kuyruk iletilerinin boşaltıldığını seçme

İleti verileri, ileti verilerinin boyutuna ve bağlaşım olanağı yapısının geçerli kullanımına dayalı olarak SMDS ya da Db2 ' e boşaltıldı. Üç kural vardır ve her kural eşleşen bir parametre çiftini belirtir. Bu parametreler, karşılık gelen bir bağlaşım olanağı yapısı kullanım eşiği yüzdesidir (**OFFLDnTH**) ve bir 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	Açıklama
Kural çifti 1	OFFLD1TH(70) ve OFFLD1SZ(32K)	Bağlaşım olanağı yapısı, 32 KB ' yi aşan iletiler için %70 'in üzerinde tam boşaltma verisinden fazlaysa
Kural çifti 2	OFFLD2TH(80) ve OFFLD2SZ(4K)	Bağlaşım olanağı yapısı 4 KB ' yi aşan iletiler için %80 'den fazla tam boşaltma verisi ise
Kural çifti 3	OFFLD3TH(90) ve OFFLD3SZ(0K)	Bağlaşım olanağı yapısı, 0 KB ' yi aşan iletiler için %90 'dan fazlaysa (tüm iletiler)

Bir boşaltma kuralı 64K OFFLD *x* SZ değerine sahipse bu, kuralın etkin olmadığını gösterir. Bu durumda iletiler yalnızca başka bir boşaltma kuralı geçerli olduğunda ya da ileti 63.75 KB ' den büyükse ve bu nedenle yapıda saklanamayacak kadar büyükse boşaltılacaktır.

Boşaltılan her ileti için, bağlaşım olanağında 0.75 KB depolama alanı gerekir.

Her yapı için belirtilebilir olan üç boşaltma kuralı aşağıdaki gibi kullanılmak üzere tasarlanmıştır.

- Performans
  - Uygulama yapısında yeterli alan olduğunda, ileti verileri yalnızca yapıda saklanamayacak kadar büyükse ya da daha düşük bir ileti boyutu eşiğini aşarsa, bunu yapıda depolamaya ilişkin performans değeri, gereksinim duyacağı yapı alanı miktarına değmeyecek şekilde boşaltılmalıdır.
  - Belirli bir ileti büyüklüğü eşiği gerekiyorsa, bu, geleneksel olarak ilk boşaltma kuralı kullanılarak belirtilir.
- Kapasite
  - Uygulama yapısında çok az alan olduğunda, geri kalan alanı en iyi şekilde kullanmak için ileti verileri miktarı üst sınırı boşaltılmalıdır.
  - Üçüncü boşaltma kuralı, yapı neredeyse dolduğunda çoğu iletinin boşaltılması gerektiğini belirtmek için geleneksel olarak kullanılır; bu nedenle, uygulama yapısındaki girişler genellikle boyut alt sınırında olur (yaklaşık 0.75K bayt gerektirir).
  - Kullanım eşiği parametresi, uygulama yapısı boyutuna ve beklenen birikim sayısı üst sınırına dayalı olarak seçilmelidir. Örneğin, beklenen birikim listesi üst sınırı 1M iletise, bu ileti sayısı için gereken yapı depolama miktarı yaklaşık 0.75G bayttır. Bu, örneğin yapı yaklaşık 10G baytsa, tüm iletilerin boşaltılması için kullanım eşiğinin %92 ya da daha düşük bir değere ayarlanması gerektiği anlamına gelir.
  - Yapı alanı öğelere ve girdilere bölünür ve genel olarak yeterli alan olsa da bunlardan biri diğerinden önce tükenebilir. 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 boşaltılması için kullanım eşiği %80 civarında daha iyi ayarlanabilir.

- Yastıklı geçiş:
  - Bağlaşım tesisi yapısında kalan alan miktarı azaldıkça, performans özelliklerinde büyük bir ani değişikliğin olması istenmeyen bir durum olacaktır. Bağlaşım tesis yönetiminin, kullanılan öğelere tipik giriş oranında ani bir eşik değişikliğine sahip olması da istenmeyen bir durum.
  - İkinci boşaltma kuralı, geleneksel olarak performans ve kapasite önyargılı boşaltma kuralları arasında bir ara koruma sağlamak için kullanılır. Bağlaşım olanağı yapısında kullanılan alan bir ara eşiği aştığında boşaltma etkinliğinde önemli bir artışa neden olacak şekilde ayarlanabilir. Bu, kalan alanın daha yavaş kullanıldığı anlamına gelir ve bağlaşım olanağı otomatik değiştirme işlemlerine daha yüksek kullanım düzeylerine uyum sağlaması için daha fazla zaman verir.

Bağlaşım olanağı yapısı genişletilemezse ve önceden belirlenmiş bazı iletilerin saklanması gerekiyorsa, tüm iletilere ilişkin verilerin boşaltılmasının uygun bir eşikte başladığından emin olmak için üçüncü kural gerektiği şekilde değiştirilebilir.

Örneğin, bağlaşım olanağı yapısı boyutu 4 GB ise ve önceden belirlenen ileti sayısı 1 milyon ise, 1.000.000 * 0.75 KB gerekir; bu da 768 MB, 18.754 GB 'dir. Bu durumda, tüm iletilerin boşaltılması için eşiğin %90 yerine %80 civarında ayarlanması gerekir. Bu, OFFLD3TH(80) ve OFFLD3SZ(0K) parametrelerini verir. Diğer boşaltma parametrelerinin de ayarlanması gerekir.

Çok küçük iletilerin boşaltılmasının önemli bir performans etkisi olduğu bulunursa, ancak göreli etki daha büyük iletiler için daha azsa, diğer kurallara ilişkin kullanım eşikleri daha önce daha büyük iletileri boşaltmak için azaltılabilir ve boşaltılması gerekmeden önce küçük iletiler için yapıda daha fazla alan bırakılabilir.

Örneğin, 32KB değerini aşan iletiler sık sık oluşuyorsa, ancak bunların boşaltılması için geçen süre başarımı (RMF istatistiklerinden ya da uygulama performansından belirlendiği gibi) bunları bağlaşım olanağında tutmak için eşik değeri %0 olarak ayarlandıysa, bu tür tüm iletileri boşaltmak için ilk kural eşiği %0 olarak ayarlanabilir. Bu, OFFLD1TH(0) ve OFFLD1SZ(32K) parametrelerini verir. Yine diğer boşaltma parametrelerinin ayarlanması gerekir.

16 KB ve 6 KB gibi belirli ara boyutlarda birçok ileti varsa, ikinci kural için ileti boyutu seçeneğini değiştirmek yararlı olabilir, böylece büyük olanlar oldukça düşük bir kullanım eşiğinde boşaltılır, önemli miktarda alan tasarrufu sağlanmış olur, ancak küçük olanlar hala yalnızca bağlaşım tesisinde depolanır.

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 178

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.

[1] `(Hits / (Hits+Misses))* 100`

- 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.

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 komutlar açıklanır ve bu komutlara erişim sağlanır.

Büyük ileti boşaltma (**OFFLOAD** ve boşaltma kuralları) ve paylaşılan ileti veri kümeleriyle ilgili **CFSTRUCT** seçeneklerini görüntüleyin ve değiştirin (**DSGROUP, DSBLOCK, DSBUFS, DSEXPAND)**:

- CFSTRUCT 'U GöRüNTüLE
- CFSTRUCT 'U TANıMLAYıN
- CFSTRUCT ' YI DEĞIŞTIRIN
- CFSTRUCT 'U SIL

CFSTRUCT Büyük ileti boşaltma ile ilgili durumu **(OFFLDUSE)** görüntüle:

- CFSTATüSü GöRüNTüLE

Geçersiz kılma veri kümesi seçeneklerini görüntüle ve değiştir (**DSEXPAND** ve **DSBUFS**) tek tek kuyruk yöneticileri için:

- SMDS ' LERI GöRüNTüLE
- SMDS ' YI DEĞIŞTIRIN

Kuyruk paylaşım grubu içindeki veri kümelerinin durumunu ve kullanılabilirliğini görüntüleyin ya da değiştirin:

- CFSTATUS TIPINI (SMDS) GöRüNTüLE
- SMDS ' YI SıFıRLA

Bir kuyruk yöneticisine ilişkin SMDS veri kümesi alanı kullanımı ve arabellek kullanımı bilgilerini görüntüler:

- KULLANıM TIPINI (SMDS) GöRüNTüLE

Bağlantıların durumunu ve uygunluğunu görüntüleme ya da değiştirme (**SMDSCONN**) tek bir kuyruk yöneticisinden alınan veri kümelerine:

- SMDSCONN GöRüNTüLE
- SMDSCONN ' U BAşLAT
- SMDSCONN ' U DURDUR

Gerektiğinde SMDS ' deki büyük ileti verileri de dahil olmak üzere paylaşılan iletileri yedekleyin ve kurtarın:

- YEDEKLEME CFSTRUCT
- CFSTRUCT 'U KURTARıN

z/OS *Paylaşılan kuyrukları kullanmanın avantajları*

Paylaşılan kuyruk, IBM MQ uygulamalarının ölçeklenebilir, yüksek düzeyde kullanılabilir olmasını ve iş yükü dengelemenin uygulanmasını sağlar.

Paylaşılan kuyrukların avantajları

Klonlanan sunucuların işi tek bir paylaşılan kuyruktan çektiği paylaşılan kuyruk mimarisi bazı yararlı özelliklere sahiptir:

- Sunucu uygulamasının yeni eşgörünümleri eklenerek ya da kuyruk yöneticisine (kuyruk paylaşım grubunda) ve uygulamanın bir kopyasına yeni bir z/OS görüntüsü eklenerek ölçeklenebilir.
- Oldukça uygun.
- Doğal olarak, kuyruk paylaşım grubundaki her kuyruk yöneticisinin kullanılabilir işleme kapasitesine dayalı olarak *çekme* iş yükü dengelemesi gerçekleştirir.

Yüksek kullanılabilirlik için paylaşılan kuyrukların kullanılması

Aşağıdaki örnekler, uygulama kullanılabilirliğini artırmak için paylaşılan bir kuyruğu nasıl kullanabileceğinizi göstermektedir.

Ağda çalışan istemci uygulamalarının z/OSüzerinde çalışan sunucu uygulamaları için istekte bulunmasını istediği bir IBM MQ senaryosu düşünün. İstemci uygulaması bir istek iletisi oluşturur ve bir istek kuyruğuna yerleştirir. Daha sonra istemci, istek iletisinin ileti tanımlayıcısında adı belirtilen yanıt kuyruğuna gönderilen sunucudan yanıt bekler.

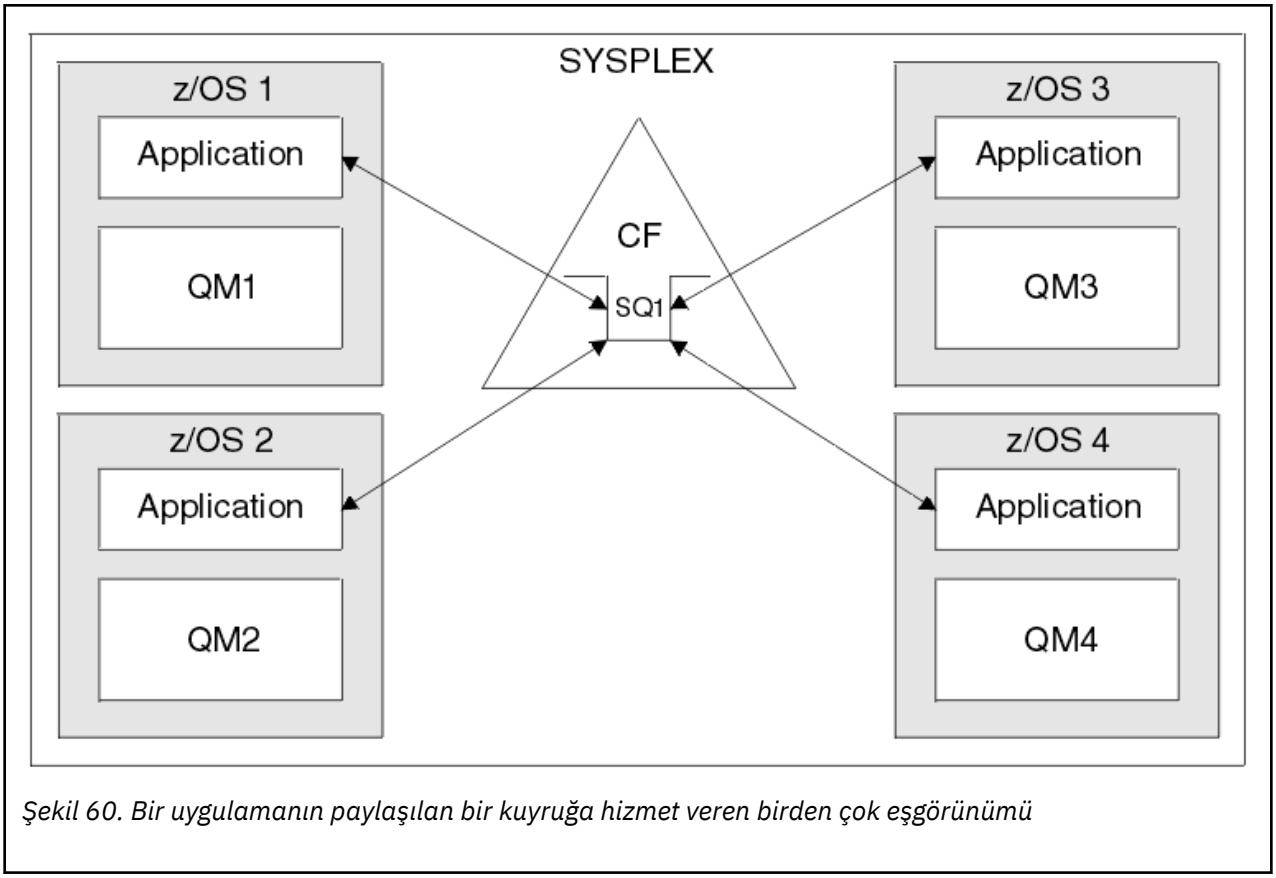
IBM MQ , istek iletisinin istemci makinesinden z/OS üzerindeki sunucunun giriş kuyruğuna iletilmesini ve yanıtın sunucudan istemciye geri gönderilmesini yönetir. Sunucunun giriş kuyruğu paylaşılan kuyruk olarak tanımlanarak, kuyruğa konan iletiler, kuyruk paylaşım grubundaki herhangi bir kuyruk yöneticisine alınabilir. Bu, sysplex 'teki her z/OS görüntüsünde bir kuyruk yöneticisi yapılandırabileceğiniz ve bunların tümünü aynı kuyruk paylaşım grubuna bağlayarak, bunlardan herhangi birinin sunucunun giriş kuyruğundaki iletilere erişebileceği anlamına gelir.

Kuyruk yöneticilerinden biri olağandışı sonlansa da ya da denetimle ilgili nedenlerden ötürü durdurmanız gerekse de, sunucunun giriş kuyruğundaki iletiler hala kullanılabilir. Bir z/OS görüntüsünün tamamını çevrimdışı yapabilirsiniz ve iletiler hala kullanılabilir.

Paylaşılan bir kuyruktaki iletilerin bu kullanılabilirliğinden yararlanmak için, sistem şebekesinde her z/OS görüntüsünde sunucu uygulamasının bir eşgörünümünü çalıştırarak daha yüksek sunucu uygulama kapasitesi ve kullanılabilirliği sağlayın (bkz. Şekil 60 sayfa 180).

Sunucu uygulamasının bir eşgörünümü, paylaşılan kuyruktan bir istek iletisi alır ve içeriğe dayalı olarak işlemesini gerçekleştirerek, istemciye IBM MQ iletisi olarak geri gönderilen bir sonuç üretir. Yanıt iletisi, istek iletisinin ileti tanımlayıcısında adı belirtilen yanıt kuyruğu ve yanıt kuyruğu yöneticisine gönderilir.

Dönüş yolunu yapılandırmak için kullanabileceğiniz birçok seçenek vardır. Bu seçeneklerle ilgili daha fazla bilgi için bkz. “Dağıtılmış kuyruğa alma ve kuyruk paylaşım grupları” sayfa 198.



Şekil 60. Bir uygulamanın paylaşılan bir kuyruğa hizmet veren birden çok eşgörünümü

Eşdüzey kurtarma

Bir kuyruk paylaşım grubundaki iletilerin kullanılabilirliğini daha da geliştirmek için IBM MQ , gruptaki başka bir kuyruk yöneticisinin bağlaşım olanağıyla bağlantısını olağandışı bir şekilde kesmesini ve bu kuyruk yöneticisine ilişkin iş birimlerini tamamlamasını (olanaklı olduğu durumlarda) saptar. Bu özellik *eş kurtarma* olarak bilinir.

Bir kuyruk yöneticisinin, uygulamanın uyumlulaştırma noktasında bir kuyruktan istek iletisi aldığı, ancak henüz yanıt iletisini koymadığı ya da iş birimini kesinleştirmediği bir noktada olağandışı bir şekilde sona erdiğini varsayın. Kuyruk paylaşım grubundaki başka bir kuyruk yöneticisi hatayı algılar ve başarısız olan kuyruk yöneticisinde gerçekleştirilmekte olan iş birimlerini geri alır. Bu, istek iletisinin istek kuyruğuna geri konacağı ve başarısız olan kuyruk yöneticisinin yeniden başlatılmasını beklemeden, işlenecek diğer sunucu eşgörünümlerinden biri için kullanılabilir olduğu anlamına gelir.

IBM MQ bir iş birimini otomatik olarak çözemezse, paylaşılan kısmı, kuyruk paylaşım grubundaki başka bir kuyruk yöneticisinin o işi işlemeye devam etmesini sağlamak için 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 187
- “İyileştirilmiş performans-temel yapılandırma” sayfa 193

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 182 .
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.

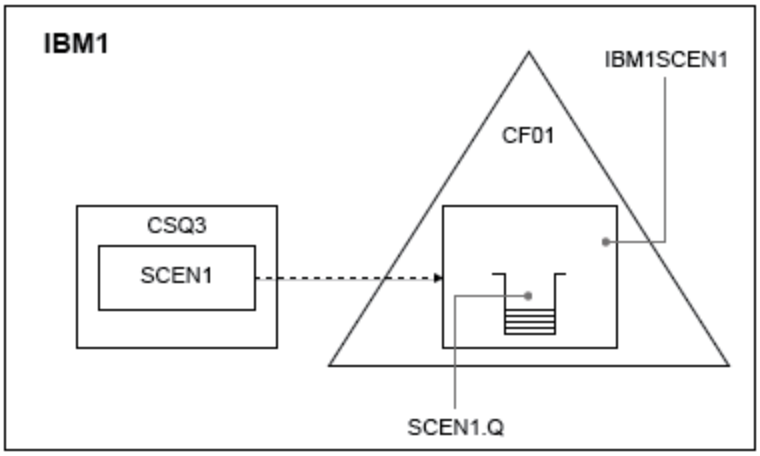
Bu, kullanılmakta olan paylaşılan kuyruğun, koyma uygulamasından iki saatlik iletiler içerebilmesi gerektiği anlamına gelir. Şu anda bunu varsayılan boşaltma kurallarını ve SMDS ' yi kullanarak başarıyorsunuz.

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 61 sayfa 188içinde gösterilmektedir.



Şekil 61. 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.

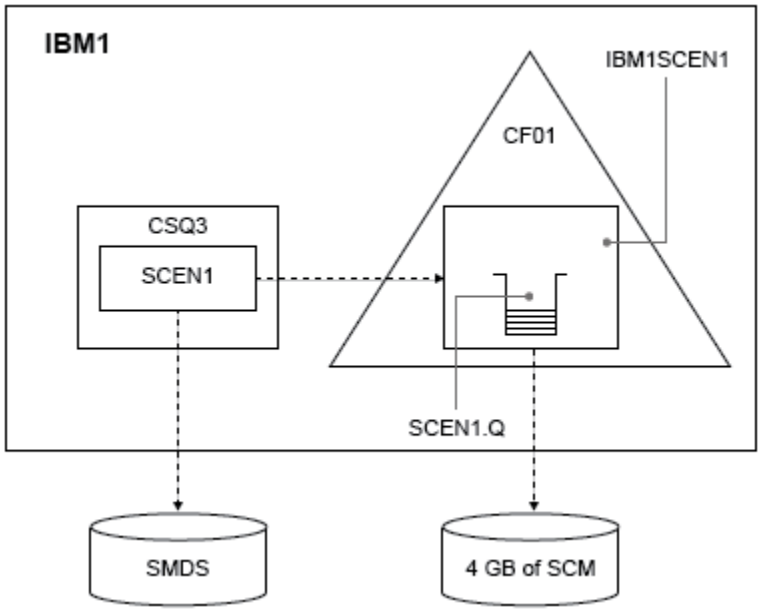
Bu görev hakkında

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 187içinde açıklanan temel yapılandırmayı kullanır. Senaryo, paylaşılan ileti veri kümelerinin (SMDS) ve daha sonra SCM ' nin ilk yapıya eklenmesini açıklar.

Bu son yapılandırma Şekil 62 sayfa 190içinde gösterilmektedir.



Şekil 62. 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 191 .

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 18. 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.

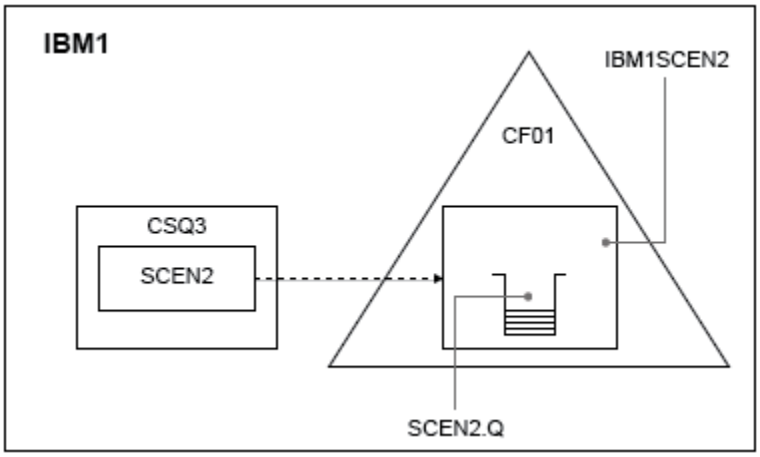
Alternatif olarak, daha büyük yapılar kullanmalı ya da iletileri SMDS ' ye boşaltmalısınız.

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 63 sayfa 194içinde gösterilmektedir.



Şekil 63. 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 196başlıklı 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.

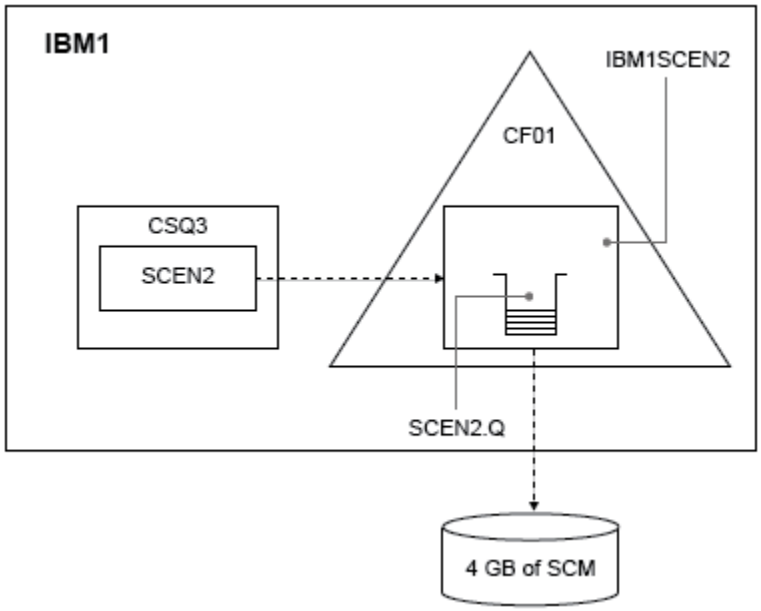
Bu görev hakkında

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 193içinde açıklanan temel yapılandırmayı kullanır. Senaryo, SCM ' nin ilk yapıya eklenmesini açıklar.

Bu son yapılandırma Şekil 64 sayfa 197içinde gösterilmektedir.



Şekil 64. 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 193içinde 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 193' 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.

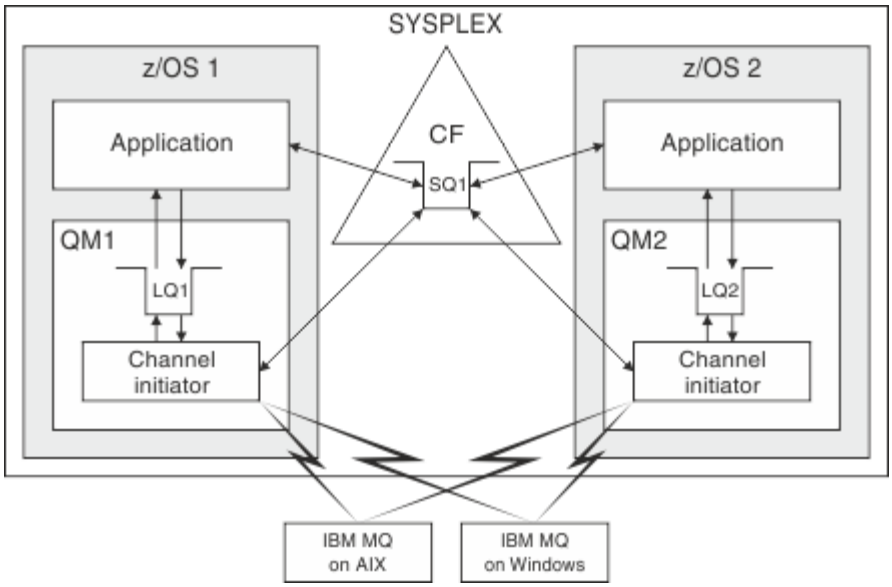
z/OS Dağıtılmış kuyruğa alma ve kuyruk paylaşım grupları

Dağıtılmış kuyruğa alma ve kuyruk paylaşımı grupları, uygulama sistemlerinizin kullanılabilirliğini artırmak için kullanabileceğiniz iki tekniktir. Bu teknikler hakkında daha fazla bilgi edinmek için bu konuyu kullanın.

Paylaşılan kuyruklardaki iletilerin yüksek kullanılabilirliğini tamamlamak için, IBM MQ ' in dağıtılmış kuyruğa alma bileşeni aşağıdakileri sağlamak üzere ek işlevlere sahiptir:

- Ağ için daha yüksek kullanılabilirlik.
- Kuyruk paylaşım grubuna gelen ağ bağlantıları için artan kapasite.

Şekil 65 sayfa 199 içinde dağıtılmış kuyruğa alma ve kuyruk paylaşım grupları gösterilir. Her ikisi de aynı kuyruk paylaşım grubuna ait olan bir sistem şebekesindeki iki kuyruk yöneticisini gösterir. Her ikisi de SQ1paylaşılan kuyruğuna erişebilir. Ağdaki kuyruk yöneticileri (örneğin, AIX ve Windows üzerinde), her iki kuyruk yöneticisinin kanal başlatıcısı aracılığıyla bu kuyruğa ileti yerleştirebilirler. Her iki kuyruk yöneticisindeki eşkopyalanmış uygulamalar kuyruğa hizmet verir.



Şekil 65. Dağıtılmış kuyruğa alma ve kuyruk paylaşım grupları

İlgili kavramlar

“Paylaşılan kanallar” sayfa 199
IBM MQ for z/OSile paylaşılan kanalların kavramlarını ve bunların kullanımını anlamak için bu konuyu kullanın.

“Grup içi kuyruğa alma” sayfa 204
Bu bölümde, z/OS platformuna özgü bir IBM MQ for z/OS işlevi olan grup içi kuyruğa alma açıklanmaktadır. Bu işlev yalnızca, bir kuyruk paylaşım grubuna tanımlanan kuyruk yöneticileri için kullanılabilir.

“Kümeler ve kuyruk paylaşım grupları” sayfa 201
Kuyruk paylaşım gruplarını kümelerle nasıl kullanabileceğinizi anlamak 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.

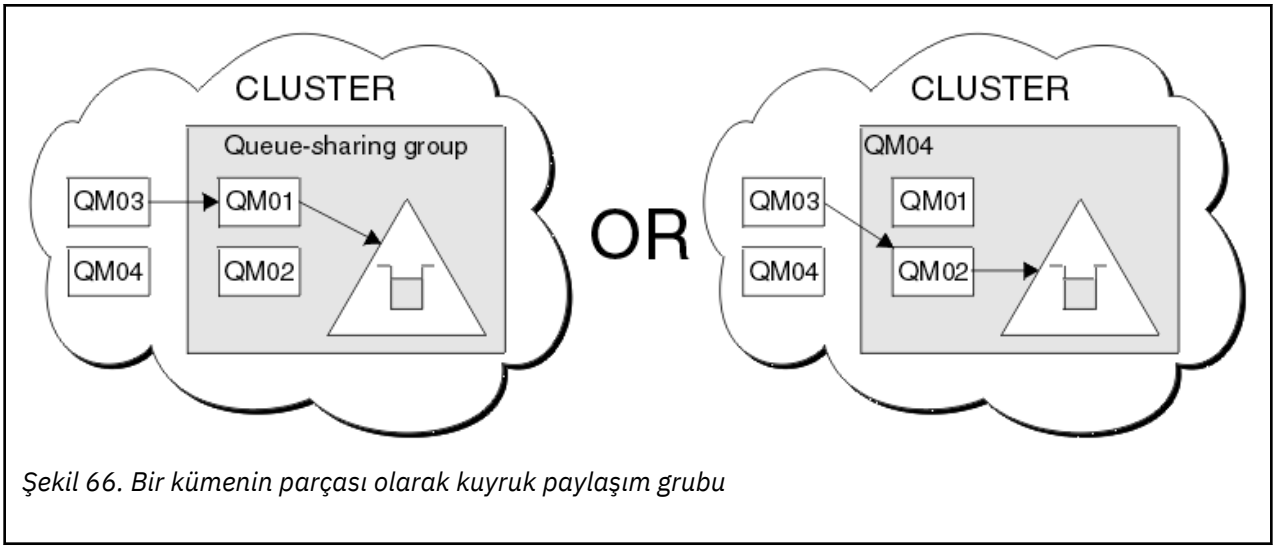
z/OS *Kümeler ve kuyruk paylaşım grupları*

Kuyruk paylaşım gruplarını kümelerle nasıl kullanabileceğinizi anlamak için bu konuyu kullanın.

Paylaşılan kuyruklarınızı tek bir tanımda bir küme için kullanılabilir kılabilirsiniz. Bunu yapmak için, paylaşılan kuyruğu tanımladığınızda kümenin adını belirtirsiniz.

Ağdaki kullanıcılar, paylaşılan kuyruğu, kuyruk paylaşım grubu içindeki her kuyruk yöneticisi tarafından barındırıldığını görürler (paylaşılan kuyruk, kuyruk paylaşım grubu tarafından barındırıldığı şeklinde duyurulmaz). İstemciler, iletileri aynı paylaşılan kuyruğa koymak için kuyruk paylaşım grubunun herhangi bir üyesiyle oturum başlatabilir.

Şekil 66 sayfa 202 , bir kümenin üyelerinin, kuyruk paylaşım grubunun herhangi bir üyesi aracılığıyla paylaşılan bir kuyruğa nasıl erişebileceğini gösterir.



Şekil 66. Bir kümenin parçası olarak kuyruk paylaşım grubu

z/OS Paylaşılan kuyruklar ile iş yükü dağıtımını etkileme

Bir kuyruk paylaşım grubundaki paylaşılan kuyruklar ile iş yükü dağıtımını etkileyen etkenleri anlamak için bu konuyu kullanın.

IBM MQ , paylaşılan kuyruklar için iş yükü dengelemesi sağlamaz. Ancak, bir kuyruk paylaşım grubundaki (QSG) iş yükü dağılımı *çekme tabanlı bir şekilde*etkilenebilir. Kuyruk yöneticisinin bir kuyruğa hizmet vermesinin (paylaşılan kuyruğa yazılan bir ileti alması) seçimi, kuyruk paylaşım grubundaki her kuyruk yöneticisinin kullanılabilir işleme kapasitesinden ve sistem şebekesinde tanımlanan iş yükü yönetimi hedeflerinden etkilenir.

Ancak, bir iletinin MQPUT değerini gerçekleştiren kuyruk yöneticisinin, iletiyi hangi kuyruk yöneticisinin aldığını belirmede büyük bir etkisi olabilir.

Yerel bir kuyruk yöneticisinin MQGET gerçekleştirme olasılığı daha yüksektir

MQPUT gerçekleştiren bir uygulama için, yerel kuyruk yöneticisinin uygulamanın bağlı olduğu kuyruk yöneticisi olduğu söylenir.

Bir alma uygulaması adına MQGET gerçekleştirerek bir iletiye ilişkin MQPUT ' yi tam olarak hangi kuyruk yöneticisinin hizmetinde hizmet verdiği aşağıdaki hususlardan etkilenir.

Bir ileti boş bir paylaşılan kuyruğa konduğunda, yerel kuyruk yöneticisi genellikle kuyruk paylaşım grubundaki diğer kuyruk yöneticilerine bildirilmeden önce gönderilir. Yerel kuyruk yöneticisi iletiyi işleyecek bir konumdaysa, QSG ' deki diğer kuyruk yöneticisinden önce bağlaşım olanağından (CF) bir liste geçişi bildirimi alır. (Liste geçişi bildirimi, paylaşılan kuyruğun boş durumundan boş durumuna değiştirildiğini bildiren bir bildirimdir.)

Bu durumda olası senaryolar şunlardır:

1. Kalıcı olmayan iletiye ilişkin MQPUT eşitleme noktası dışında ve *hızlı alıcı beklemeye alındı*.

   Kuyruğa ilişkin yerel kuyruk yöneticisinde *MQGET ve bekleme* içeren bir uygulama varsa, iletinin MQPUT doğrudan alma uygulamasının arabelleğine geçirilir ve kuyruğa yazılmaz. Bu, paylaşılan ve paylaşılmayan kuyruklar için geçerlidir. Bu özelliğe genellikle *bekleme alıcısına hızlı konma* mekanizması denir. Paylaşılan kuyruklar söz konusu olduğunda, boş kuyruktan boş olmayan kuyruğa geçiş olmadığı için QSG ' deki başka bir kuyruk yöneticisine bildirim gönderilmez. Bu, örneğin, bu kuyruk yöneticisinin bu uygulamadaki tüm girişlere hizmet verebileceği ve başka bir uygulamanın kuyruğa ileti koymadığı varsayılarak, kuyruk paylaşım grubundaki başka bir kuyruk yöneticisinin bu kuyruğun boşaltılmasına yardımcı olmadığı anlamına gelir. Ancak, yerel kuyruk yöneticisinde bekleme içeren bir MQGET yoksa ve paylaşılan kuyruğa bir ileti konursa, CF, liste geçişlerine ilişkin bildirimlere ilişkin kurallara göre kuyruk paylaşım grubundaki diğer kuyruk yöneticilerini bilgilendirecektir.
2. Kalıcı ya da zamanuyumlu bir iletiye ilişkin MQPUT.

Bu durumda, yerel kuyruk yöneticisinde *MQGET ve bekleme* içeren bir uygulama varsa, ileti paylaşılan kuyruğa konur ve CF, liste geçişlerinin bildirimlerine ilişkin kurallara göre kuyruk paylaşım grubundaki diğer kuyruk yöneticilerini bilgilendirir. Ancak, yerel kuyruk yöneticisi CF ' den bir geçiş bildirimi gelmesini beklemez, ancak önce yerel *MQGET with wait* komutunu kullanır ve genellikle, kuyruk paylaşım grubundaki herhangi bir kuyruk yöneticisi bir CF bildirimine yanıt vermeden önce uygulama adına bu iletinin alımını gerçekleştirir. Bu, yerel kuyruk yöneticisinin ne kadar meşgul olduğuna bağlıdır. Tersi durumda, boş kuyruğa ileti gelmesi nedeniyle CF tarafından bildirilen herhangi bir kuyruk yöneticisi önce alma için hizmet sunmaya çalışır. Yanıt veren ilk kuyruk yöneticisi yeni iletiyi işler.

3. Son olarak, CF ' nin kuyruk için boş durumundan boş olmayanlara bir durum değişikliği bildirimi gönderdiği kuyruk boşaltılmazsa, bağlı tüm kuyruk yöneticileri kuyruğun işlenmesine yardımcı olma fırsatına sahip olur. Bu durumda, iş yükünün *çekme tabanlı*olduğu söylenmektedir.

Bu tasarım, tamamen çekme tabanlı bir iş yükü dağıtımına göre daha yüksek performans sağlar. Bunun amacı, CF 'de tutulan kuyruklar tarafından sunulan yüksek kullanılabilirliğin avantajlarından yararlanmak ve mümkünse, kuyruk yöneticisinin CF' ye başvurmasına gerek kalmadan MQGET ' i gerçekleştirmesine ve böylece ileti iş yükünü mümkün olduğunca verimli bir şekilde işlemesine izin vermektir.

İş yükü dengesinin vurgulanması, daha önce açıklanan performans geliştirmelerinden daha önemli olduğu durumlarda alternatif yaklaşımlar benimsenebilir. Örneğin, alma uygulamalarının hiçbirinin, koyma uygulamasının bağlı olduğu aynı kuyruk yöneticisine bağlanmadığından emin olun. Bu tasarım kullanılarak, tüm iletiler kuyruğa yerleştirilir ve kuyruk, bu tür geçişleri işlemek için CF algoritmasına uygun olarak boş olandan boş olmayanlara hareket ettiğinde QSG ' deki tüm kuyruk yöneticileri bilgilendirilir. Buna ek olarak, *hızlı bekleme alıcı* mekanizması uygulanamaz.

z/OS Paylaşılan kuyruklar ve kuyruk paylaşım gruplarıyla ilgili daha fazla bilgi nereden bulunacak?

IBM MQ for z/OS ' in paylaşılan kuyrukları ve kuyruk paylaşım gruplarını nasıl kullandığına ilişkin daha fazla bilgi edinmek için bu konudaki tabloyu kullanın.

Çizelge 19. Paylaşılan kuyruklar ve kuyruk paylaşım gruplarıyla ilgili daha fazla bilgi nereden bulunacak?

Konu	**Nereye bakıyorsunuz?**
Kuyruk paylaşım grubu kurtarma	“z/OS üzerinde kurtarma ve yeniden başlatma” sayfa 242
Kuyruk paylaşım grubu güvenliği	“IBM MQ for z/OS içindeki güvenlik kavramları” sayfa 258
Özel ve genel nesne tanımlamaları Komutlar farklı kuyruğa yönlendiriliyor Yöneticiler	z/OS
Bağlaşım tesisinizin planlanması ortam	Bağlaşım olanağı kaynaklarının tanımlanması
SMDS ortamınızın planlanması	Paylaşılan ileti veri kümesi (SMDS) ortamınızın planlanması
Planlama Db2 Ortam	Db2 ortamınızın planlanması
Paylaşılan kuyruklarınızın ayarlanması Sistem değiştirbeleri	“Paylaşılan kuyruklar ve kuyruk paylaşım grupları” sayfa 160

Çizelge 19. Paylaşılan kuyruklar ve kuyruk paylaşım gruplarıyla ilgili daha fazla bilgi nereden bulunacak? (devamı var)	
Konu	**Nereye bakıyorsunuz?**
Yardımcı programlar Kuyruklar geçiriliyor	IBM MQ yardımcı programlar da cı larda bakın unun larda unun unun z/OS
Konsol iletileri	IBM MQ for z/OS ile ilgili iletiler
MQSC komutları	MQSC komutları
IBM MQ Kümeler	Kuyruk yöneticisi kümesinin yapılandırılması
IBM MQ dağıtılmış kuyruğa alma Kanal adları	Dağıtılmış kuyruk yönetimine giriş
Uygulamalar yazılıyor	Uygulama tasarımına genel bakış
MQCONNX çağrısı	MQCONNX

z/OS Grup içi kuyruğa alma

Bu bölümde, z/OS platformuna özgü bir IBM MQ for z/OS işlevi olan grup içi kuyruğa alma açıklanmaktadır. Bu işlev yalnızca, bir kuyruk paylaşım grubuna tanımlanan kuyruk yöneticileri için kullanılabilir.

Kuyruk paylaşım gruplarına ilişkin bilgi için bkz. “Paylaşılan kuyruklar ve kuyruk paylaşım grupları” sayfa 160.

Grup içi kuyruğa alma kavramları

Kanal tanımlamadan, bir kuyruk paylaşım grubundaki kuyruk yöneticileri arasında hızlı ileti aktarımı gerçekleştirebilirsiniz. Bu, SYSTEM.QSG.TRANSMIT.QUEUE. Kuyruk paylaşım grubundaki her kuyruk yöneticisi, kuyruk yöneticisine yönlendirilen iletilerin bu kuyruğa gelmesini bekleyen grup içi kuyruğa alma aracısı adı verilen bir görev başlatır. Böyle bir ileti saptandığında, kuyruktan kaldırılır ve doğru hedef kuyruğa yerleştirilir.

Standart ad çözme kuralları kullanılır, ancak grup içi kuyruğa alma (IGQ) etkinleştirildiyse ve hedef kuyruk yöneticisi kuyruk paylaşım grubu içindeyse, SYSTEM.QSG.TRANSMIT.QUEUE , iletiyi bir iletim kuyruğu ve kanal kullanmak yerine doğru hedef kuyruk yöneticisine aktarmak için kullanılır.

Bir kuyruk yöneticisi özniteliği aracılığıyla grup içi kuyruğa alma etkinleştirilir. Grup içi kuyruğa alma, kalıcı olmayan iletileri eşitleme noktası dışına ve eşitleme noktası içindeki kalıcı iletileri taşır. İletileri hedef kuyruğa teslim etme konusunda bir sorun bulursa, grup içi kuyruğa alma işlemi bunları teslim edilmeyen iletiler kuyruğuna yerleştirmeyi dener. Gitmeyen iletiler dolu ya da tanımlanmamışsa, kalıcı olmayan iletiler atılır, ancak kalıcı iletiler yedeklenir ve SYSTEM.QSG.TRANSMIT.QUEUEve IGQ aracısı, başarılı oluncaya kadar iletileri teslim etmeye çalışır.

Kuyruk paylaşım grubundaki farklı bir kuyruk yöneticisinde kuyruğa gönderilecek bir iletiyi alan bir gelen paylaşılan kanal, iletiyi doğru hedefe *atlamak* için grup içi kuyruğa alma özelliğini kullanabilir.

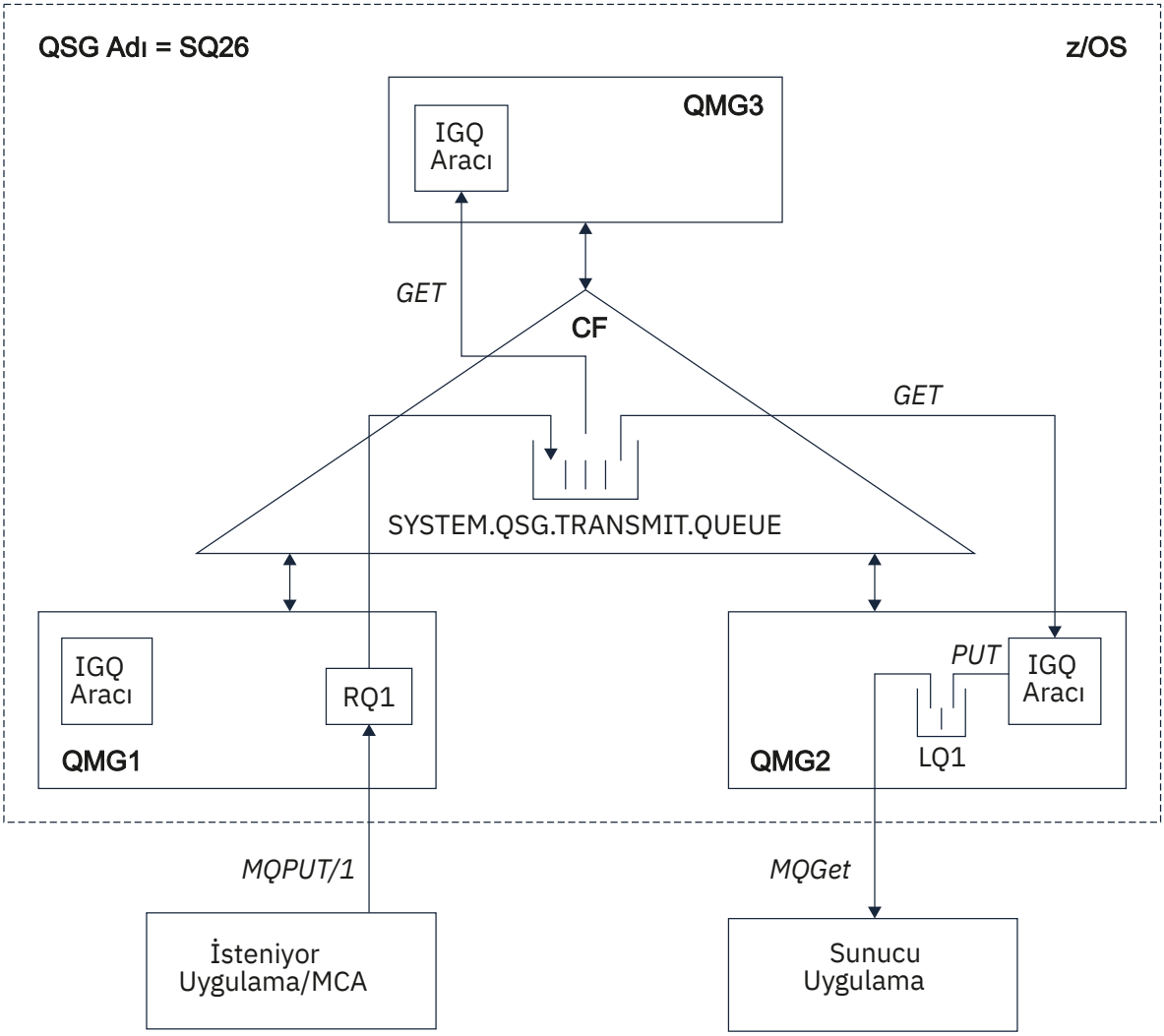
Hedef kuyruk önce hedef kuyruk yöneticisine aktarılmak yerine, hedef kuyruk paylaşılan bir kuyruksa, yerel kuyruk yöneticisinin bir iletiyi doğrudan hedef kuyruğa koymasını istediğiniz zamanlar olabilir. Bunu denetlemek için SQQMNAME kuyruk yöneticisi özniteliğini kullanabilirsiniz. SQQMNAME değerini USE olarak ayarlarsanız, MQOPEN komutu **ObjectQMgrName**ile belirtilen kuyruk yöneticisinde gerçekleştirilir.

Ancak, hedef kuyruk paylaşılan bir kuyruksa ve SQQMNAME değerini IGNORE olarak ayarlarsanız ve **ObjectQMgrName** kuyruk paylaşım grubundaki başka bir kuyruk yöneticisiyse, paylaşılan kuyruk yerel kuyruk yöneticisinde açılır. Yerel kuyruk yöneticisi hedef kuyruğu açamazsa ya da kuyruğa bir ileti yerleştiremezse, ileti belirtilen **ObjectQMgrName** ' e IGQ ya da IBM MQ kanalı aracılığıyla aktarılır.

Grup içi kuyruğa alma, daha verimli, küçük iletileri bir kuyruk paylaşım grubu içindeki uzak kuyruk yöneticilerine teslim etmek için kullanılabilir. Grup içi kuyruğa alma da büyük iletileri destekler; en büyüğü iletim kuyruğu üstbilgisinin uzunluğu 100 MB *eksi* .

Not: Bu özelliği kullanırsanız, kullanıcıların kuyruk paylaşım grubundaki her kuyruk yöneticisindeki kuyruklara aynı erişimleri olmalıdır.

Aşağıdaki çizge, grup içi kuyruğa almanın tipik bir örneğini göstermektedir.



Şekil 67. Grup içi kuyruğa alma örneği

Çizge şunları gösterir:

- SQ26adlı bir kuyruk paylaşım grubuna tanımlanan üç kuyruk yöneticinde (QMG1, QMG2ve QMG3) çalışan IGQ aracıları.
- SYSTEM.QSG.TRANSMIT.QUEUE .
- QMG1kuyruk yöneticisinde tanımlanan uzak kuyruk tanımlaması.
- QMG2kuyruk yöneticisinde tanımlı bir yerel kuyruk.

- İstekte bulunan bir uygulama (bu uygulama, QMG1kuyruk yöneticisine bağlı bir Message Channel Agent (MCA) olabilir.
- QMG2kuyruk yöneticisine bağlı bir sunucu uygulaması.
- SYSTEM.QSG.TRANSMIT.QUEUE.

Grup içi kuyruğa alma ve grup içi kuyruğa alma aracısı

Kuyruk yöneticisi kullanıma hazırlanırken bir IGQ aracısı başlatılır. Uygulamalar açıldığında ve uzak kuyruklara ileti yerleştirdiğinde, yerel kuyruk yöneticisi ileti aktarımı için grup içi kuyruğa alma kullanılıp kullanılmayacağını belirler. Grup içi kuyruğa alma kullanılacaksa, yerel kuyruk yöneticisi iletiyi SYSTEM.QSG.TRANSMIT.QUEUE. Hedef uzak kuyruk yöneticisindeki IGQ aracısı iletiyi alır ve hedef kuyruğa yerleştirir.

Grup içi kuyruğa alma terminolojisi

Terminolojinin açıklamaları: grup içi kuyruğa alma, grup içi kuyruğa alma ve grup içi kuyruğa alma aracısıyla kullanılmak üzere paylaşılan iletim kuyruğu.

Grup içi kuyruğa alma

Grup içi kuyruğa alma, bir kuyruk paylaşım grubundaki kuyruk yöneticileri arasında potansiyel olarak hızlı ve daha ucuz ileti aktarımını kanalları tanımlamaya gerek olmadan etkileyebilir.

Grup içi kuyruğa alma tarafından kullanılmak üzere paylaşılan iletim kuyruğu

Her kuyruk paylaşım grubunun SYSTEM.QSG.TRANSMIT.QUEUE . Grup içi kuyruğa alma etkinleştirildiyse, SYSTEM.QSG.TRANSMIT.QUEUE görüntülenir. Uygulamalar (İleti Kanal Aracıları (MCA 'lar da içinde olmak üzere) iletileri uzak bir kuyruğa koyduğunda, yerel kuyruk yöneticisi iletilerin hızlı aktarma için uygunluğunu belirler ve bunları SYSTEM.QSG.TRANSMIT.QUEUE.

Grup içi kuyruğa alma aracısı

IGQ aracısı, SYSTEM.QSG.TRANSMIT.QUEUE. IGQ aracısı, bu kuyruktan uygun iletileri alır ve hedef kuyruklara teslim eder.

Grup içi kuyruğa alma, kuyruk yöneticisinin kendi iç işlemesi için kendisi tarafından kullanıldığından, her kuyruk yöneticisi için IGQ aracısı her zaman başlatılır.

z/OS Grup içi kuyruğa almanın yararları

Grup içi kuyruğa almanın yararları şunlardır: azaltılmış sistem tanımları, azaltılmış sistem yönetimi, artırılmış performans, geçişi destekler ve kuyruk paylaşım grubundaki kuyruk yöneticileri arasında çoklu geçiş yapılırken iletilerin teslimini destekler.

Grup içi kuyruğa almanın yararları şunlardır:

Azaltılmış sistem tanımları
Grup içi kuyruğa alma, bir kuyruk paylaşım grubundaki kuyruk yöneticileri arasında kanal tanımlama gereksinimini ortadan kaldırır.

Azaltılmış sistem yönetimi
Bir kuyruk paylaşım grubundaki kuyruk yöneticileri arasında tanımlı kanal olmadığından, kanal yönetimi için herhangi bir gereksinim yoktur.

İyileştirilmiş performans
Bir iletinin hedef kuyruğa gönderilmesi için yalnızca bir IGQ aracısı (iki ara gönderen ve alıcı aracısı yerine) gerektiği için, grup içi kuyruğa alma kullanan iletilerin gönderilmesi, kanalları kullanan iletilerin gönderilmesinden daha ucuz olabilir. Grup içi kuyruğa alma içinde yalnızca bir alıcı bileşen vardır; gönderme bileşeni gereksinimi kaldırılmıştır. Bunun nedeni, yerel kuyruk yöneticisindeki koyma işlemi tamamlandığında ve eşitleme noktası kapsamına konan iletiler söz konusu olduğunda, iletinin

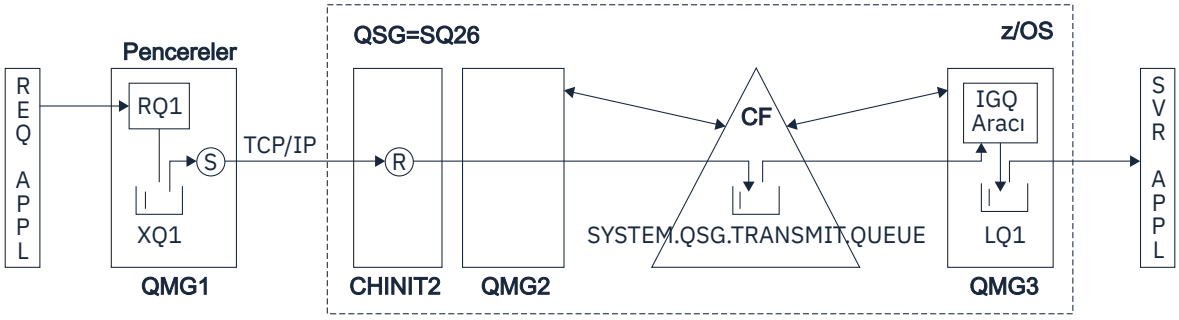
hedef kuyruk yöneticisindeki IGQ aracısı tarafından hedef kuyruğa teslim edilmek üzere kullanılabilir olmasıdır.

Geçişin desteklenmesini sağlar

Bir kuyruk paylaşım grubu dışındaki uygulamalar, kuyruk paylaşım grubundaki belirli bir kuyruk yöneticisine bağlanırken, iletileri kuyruk paylaşım grubundaki herhangi bir kuyruk yöneticisinde bulunan bir kuyruğa teslim edebilir. Bunun nedeni, uzak kuyruk yöneticisindeki bir kuyruğa yönlendirilen bir alıcı kanalına gelen iletilerin, grup içi kuyruğa alma kullanılarak hedef kuyruğa şeffaf bir şekilde gönderilebilir olmasıdır. Bu olanak, kuyruk paylaşım grubu dışındaki sistemlerin değiştirilmesine gerek kalmadan uygulamaların kuyruk paylaşım grubu arasında konuşlandırılmasına olanak sağlar.

Tipik bir yapılandırma aşağıdaki çizgeyle gösterilir:

- QMG1 kuyruk yöneticisine bağlı istekte bulunan bir uygulamanın, QMG3kuyruk yöneticisindeki yerel bir kuyruğa ileti göndermesi gerekir.
- QMG1 kuyruk yöneticisi yalnızca QMG2kuyruk yöneticisine bağlanır.
- Daha önce kanallar kullanılarak bağlanan kuyruk yöneticileri QMG2 ve QMG3artık SQ26kuyruk paylaşım grubunun üyeleridir.



Şekil 68. Geçiş desteği örneği

İşlem akışı aşağıdaki gibidir:

1. İstekte bulunan uygulama, yerel kuyruk LQ1 için QMG3uzak kuyruk yöneticisine RQ1uzak kuyruk tanımlamasına bir ileti yerleştirir.
2. Windows NT iş istasyonunda çalışan kuyruk yöneticisi QMG1, iletiyi XQ1iletim kuyruğuna yerleştirir.
3. QM1 üzerindeki Gönderici MCA 'ları, iletiyi TCP/IP kullanarak CHINIT2kanal başlatıcısında alıcı MCA' ya (R) iletir.
4. CHINIT2 adlı kanal başlatıcıdaki alıcı MCA (R), iletiyi SYSTEM.QSG.TRANSMIT.QUEUE.
5. QMG3 kuyruk yöneticisindeki IGQ aracısı, SYSTEM.QSG.TRANSMIT.QUEUE ve bunu hedef yerel kuyruğa LQ1yerleştirir.
6. Sunucu uygulaması iletiyi hedef yerel kuyruktan alır ve işler.

Bir kuyruk paylaşım grubundaki kuyruk yöneticileri arasında çoklu sıçrama sırasında iletilerin teslimi

Geçişin desteklendiği önceki çizge, bir kuyruk paylaşım grubundaki kuyruk yöneticileri arasında birden çok sıçrama yapıldığında iletilerin teslimini de gösterir. Kuyruk paylaşım grubu içindeki bir kuyruk yöneticisine gelen, ancak kuyruk paylaşım grubundaki başka bir kuyruk yöneticisine yönlendirilmiş iletiler, grup içi kuyruğa alma kullanılarak hedef kuyruk yöneticisindeki hedef kuyruğa kolayca iletilebilir.

z/OS Grup içi kuyruğa alma sınırlamaları

Grup içi kuyruğa alma sınırlamaları şunlardır: Grup içi kuyruğa alma kullanılarak aktarılmaya uygun iletiler, kuyruk yöneticisi başına grup içi kuyruğa alma aracılarının sayısı ve grup içi kuyruğa alma aracısının başlatılması ve durdurulması.

Bu konuda, grup içi kuyruğa alma sınırlamaları açıklanmaktadır.

Grup içi kuyruğa alma kullanılarak aktarılmaya uygun iletiler
Grup içi kuyruğa alma, bağlaşım olanağı (CF) içinde tanımlanan bir paylaşılan iletim kuyruğunu kullandığından, grup içi kuyruğa alma, paylaşılan kuyruklar için desteklenen ileti uzunluğu üst sınırını bir iletim kuyruğu üstbilgisinin (MQXQH) uzunluğunu çıkararak iletilerle sınırlıdır.

Kuyruk yöneticisi başına grup içi kuyruğa alma aracılarının sayısı
Bir kuyruk paylaşım grubunda kuyruk yöneticisi başına yalnızca bir IGQ aracısı başlatılır.

Grup içi kuyruğa alma aracısının başlatılması ve durdurulması
IGQ aracısı, kuyruk yöneticisi kullanıma hazırlanırken başlatılır ve kuyruk yöneticisi kapatılırken sonlandırılır. Uzun süren, kendini kurtaran (olağandışı sonlandırma durumunda) bir görev olacak şekilde tasarlanmıştır. SYSTEM.QSG.TRANSMIT.QUEUE (örneğin, bu kuyruk engellenirse) IGQ aracısı yeniden denemeye devam eder. IGQ aracısı, kuyruk yöneticisi hala etkinken aracının olağan şekilde sonlandırılmasıyla sonuçlanan bir hatayla karşılaşırsa, ALTER QMGR IGQ (ENABLED) komutu verilerek yeniden başlatılabilir. Bu komut, kuyruk yöneticisini geri dönüştürme gereksinmesini ortadan kaldırır.

Kuyruk yöneticisi özniteliği IGQ ' nun ENABLED ya da DISABLED olarak ayarlanması
Kuyruk yöneticisi özniteliği IGQ ENABLED ya da DISABLED olarak ayarlanırsa, var olan nesne tanıtıcıları MQRC_OBJECT_CHANGED neden koduyla geçersiz kılınabilir. Daha fazla bilgi için bkz. Grup içi kuyruğa alma ile çalışmaya başlama .

z/OS Grup içi kuyruğa alma ile çalışmaya başlama

Bu konuda açıklandığı gibi grup içi kuyruğa almayı etkinleştirebilir, devre dışı bırakabilir ve kullanabilirsiniz.

Grup içi kuyruğa alma etkinleştiriliyor
Kuyruk yöneticilerinizde grup içi kuyruğa alma özelliğini etkinleştirmek için aşağıdakileri yapmanız gerekir:

- SYSTEM.QSG.TRANSMIT.QUEUE. Bu kuyruğun tanımı, kuyruk paylaşım gruplarına ilişkin SYSTEM nesneleri için thlqual.SCSQPROC(CSQ4INSS) CSQINP2 örneğinde bulunabilir. Grup içi kuyruğun düzgün çalışması için bu kuyruk, thlqual.SCSQPROC(CSQ4INSS) içinde belirtildiği gibi doğru özniteliklerle tanımlanmalıdır.
- IGQ aracısı her zaman kuyruk yöneticisi kullanıma hazırlanırken başlatıldığından, grup içi kuyruğa alma her zaman gelen ileti işleme için kullanılabilir. IGQ aracısı, SYSTEM.QSG.TRANSMIT.QUEUE. Ancak, giden işleme için grup içi kuyruğa alma özelliğini etkinleştirmek için, kuyruk yöneticisi özniteliği IGQ ENABLED olarak ayarlanmalıdır.

  Önemli: Kuyruk yöneticisi özniteliği IGQ ENABLED olarak ayarlanırsa, var olan nesne tanıtıcıları MQRC_OBJECT_CHANGED neden koduyla geçersiz kılınabilir. Ek bilgi için bkz. “Grup içi kuyruğa almanın belirli özellikleri” sayfa 215 . Bu neden koduna ilişkin 'Programcı yanıtı' bölümünde açıklandığı gibi, uygulamaların bu durumu ele almak için kodlanmaları gerekir (daha fazla ayrıntı için bkz. 2041 (07F9) (RC2041): MQRC_OBJECT_CHANGED).

Buna ek olarak, IGQ, başlatma sırasında başlayan ve kapatma ile sona eren uzun süreli ve kendi kendini kurtaran bir görev olarak tasarlandıkça, daha fazla bilgi için bkz. “Grup içi kuyruğa alma sınırlamaları” sayfa 208 .

Grup içi kuyruğa alma devre dışı bırakılıyor

Giden ileti aktarımı için grup içi kuyruğa almayı devre dışı bırakmak üzere kuyruk yöneticisi özniteliği IGQ ' yu DISABLED olarak ayarlayın. Belirli bir kuyruk yöneticisi için grup içi kuyruğa alma geçersiz kılındıysa, o kuyruk yöneticisindeki IGQ aracısı SYSTEM.QSG.TRANSMIT.QUEUE , giden aktarımı için grup içi kuyruğa alma özelliği etkinleştirilmiş bir kuyruk yöneticisi tarafından.

Önemli: Kuyruk yöneticisi özniteliği IGQ ENABLED olarak ayarlanırsa, var olan nesne tanıtıcıları MQRC_OBJECT_CHANGED neden koduyla geçersiz kılınabilir. Ek bilgi için bkz. “Grup içi kuyruğa almanın belirli özellikleri” sayfa 215 . Bu neden koduna ilişkin 'Programcı yanıtı' bölümünde açıklandığı gibi, uygulamaların bu durumu ele almak için kodlanmaları gerekir (daha fazla ayrıntı için bkz. 2041 (07F9) (RC2041): MQRC_OBJECT_CHANGED).

Buna ek olarak, IGQ, başlatma sırasında başlayan ve kapatma ile sona eren uzun süreli ve kendi kendini kurtaran bir görev olarak tasarlandıkça, daha fazla bilgi için bkz. “Grup içi kuyruğa alma sınırlamaları” sayfa 208 .

Grup içi kuyruğa alma özelliğinin kullanılması

Grup içi kuyruğa alma etkinleştirildikten sonra, kullanılabilir ve bir kuyruk yöneticisi mümkün olduğunda bunu kullanır. Başka bir deyişle, bir uygulama uzak kuyruk tanımına, tam olarak nitelenmiş bir uzak kuyruğa ya da bir küme kuyruğuna bir ileti yerleştirdiğinde, kuyruk yöneticisi iletinin grup içi kuyruğa alma kullanılarak teslim edilmeye uygun olup olmadığını belirler ve uygunsa, iletiyi SYSTEM.QSG.TRANSMIT.QUEUE. Kuyruk yöneticisinin SYSTEM.QSG.TRANSMIT.QUEUE(Kuyruk), başka bir iletim kuyruğuna tercih edilir.

z/OS Grup içi kuyruğa alma yapılandırmaları

Tipik grup içi kuyruğa alma yapılandırmasına ek olarak, diğer yapılandırmalar da olabilir.

“Grup içi kuyruğa alma kavramları” sayfa 204 içinde tipik yapılandırma açıklanmaktadır.

İlgili kavramlar

“Grup içi kuyruğa alma ile dağıtılmış kuyruğa alma (birden çok teslim yolu)” sayfa 209
Kısa iletileri işleyen uygulamalar için, grup içi kuyruğa alma işlemini yalnızca bir kuyruk paylaşım grubundaki kuyruk yöneticileri arasında ileti teslim etmek için yapılandırmak uygun olabilir.

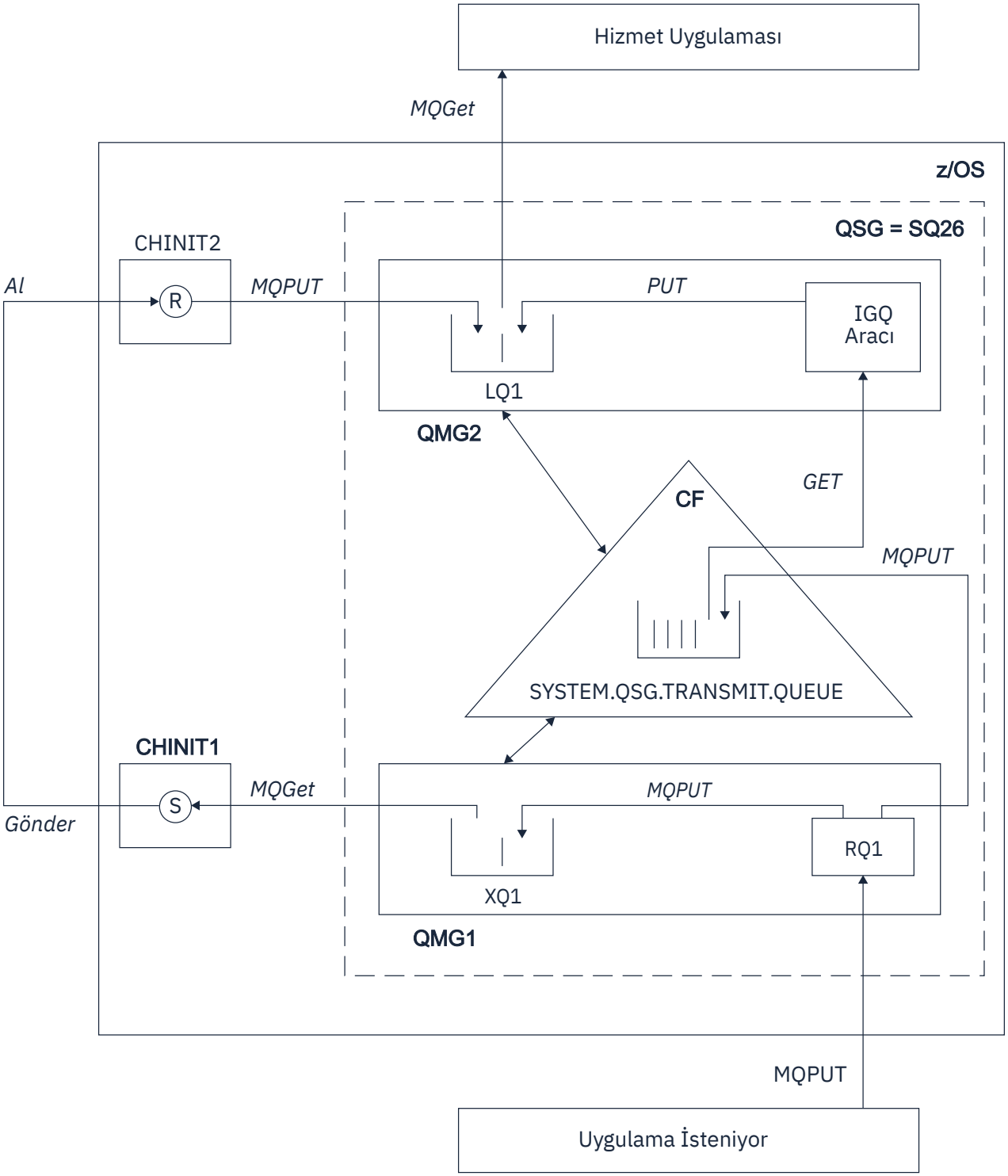
“Grup içi kuyruğa alma ile kümeleme (birden çok teslim yolu)” sayfa 211
Kuyruk yöneticileri, bir kümede ve bir kuyruk paylaşım grubunda olacak şekilde yapılandırılabilir.

“Kümeleme, grup içi kuyruğa alma ve dağıtılmış kuyruğa alma” sayfa 213
Bir kümenin üyesi olan ve bir kuyruk paylaşım grubu olan ve bir gönderen/alıcı kanal çifti kullanılarak dağıtılmış bir kuyruk yöneticisine bağlı olan bir kuyruk yöneticisi yapılandırılabilir.

z/OS *Grup içi kuyruğa alma ile dağıtılmış kuyruğa alma (birden çok teslim yolu)*

Kısa iletileri işleyen uygulamalar için, grup içi kuyruğa alma işlemini yalnızca bir kuyruk paylaşım grubundaki kuyruk yöneticileri arasında ileti teslim etmek için yapılandırmak uygun olabilir.

Kanal iletişimi üzerinden grup içi kuyruğa alma seçeneği, CFSTRUCT tip düzeyi tarafından denetlenebilir. (4 ya da 5 yerine 3). SYSTEM.QSQ.TRANSMIT.QUEUE.



Şekil 69. Örnek bir yapılandırma

Açma/Koy işlemi

1. İstekte bulunan uygulama RQ1uzak kuyruğunu açtığında, hem paylaşılmayan iletim kuyruğu XQ1 , hem de paylaşılan iletim kuyruğu SYSTEM.QSG.TRANSMIT.QUEUE.
2. İstekte bulunan uygulama, kuyruk yöneticisinde ve ileti özelliklerinde giden aktarımı için grup içi kuyruğa alma özelliğinin etkinleştirilip etkinleştirilmediğine bağlı olarak uzak kuyruğa bir ileti yerleştirdiğinde, ileti iletim kuyruğuna XQ1ya da iletim kuyruğuna SYSTEM.QSG.TRANSMIT.QUEUE.

Kuyruk yöneticisi, tüm büyük iletileri XQ1iletim kuyruğuna ve tüm küçük iletileri SYSTEM.QSG.TRANSMIT.QUEUE.

3. XQ1 iletim kuyruğu doluysa ya da kullanılamıyorsa, büyük iletilere ilişkin koyma istekleri zamanuyumlu olarak uygun bir dönüş ve neden koduyla başarısız olur. Ancak, küçük iletilere ilişkin istekleri başarılı olmaya devam eder ve SYSTEM.QSG.TRANSMIT.QUEUE.
4. SYSTEM.QSG.TRANSMIT.QUEUE dolu ya da konulamıyor, küçük iletilere ilişkin istekler uygun bir dönüş ve neden koduyla zamanuyumlu olarak başarısız oluyor. Ancak, büyük ileti koyma istekleri başarılı olmaya devam eder ve XQ1iletim kuyruğuna yerleştirilir. Bu durumda, küçük iletileri bir iletim kuyruğuna koyma girişiminde bulunulmaz.

Büyük iletilere ilişkin akış

1. İstekte bulunan uygulama, büyük iletileri RQ1uzak kuyruğuna koyar.
2. Kuyruk yöneticisi QMG1 , iletileri XQ1iletim kuyruğuna koyar.
3. QMG1 kuyruk yöneticisindeki Gönderici MCA ' ları iletileri XQ1 iletim kuyruğundan alır ve QMG2kuyruk yöneticisine gönderir.
4. QMG2 kuyruk yöneticisindeki Günlük Nesnesi MCA (R) iletileri alır ve LQ1hedef kuyruğuna yerleştirir.
5. Sunan uygulama, iletileri LQ1kuyruğundan alır ve işler.

Küçük iletiler için akış

1. İstekte bulunan uygulama, RQ1uzak kuyruğuna küçük iletiler koyar.
2. Kuyruk yöneticisi QMG1 , iletileri SYSTEM.QSG.TRANSMIT.QUEUE.
3. QMG2 kuyruk yöneticisindeki IGQ, iletileri alır ve LQ1hedef kuyruğuna yerleştirir.
4. Hizmet veren uygulama, iletileri LQ1kuyruğundan alır.

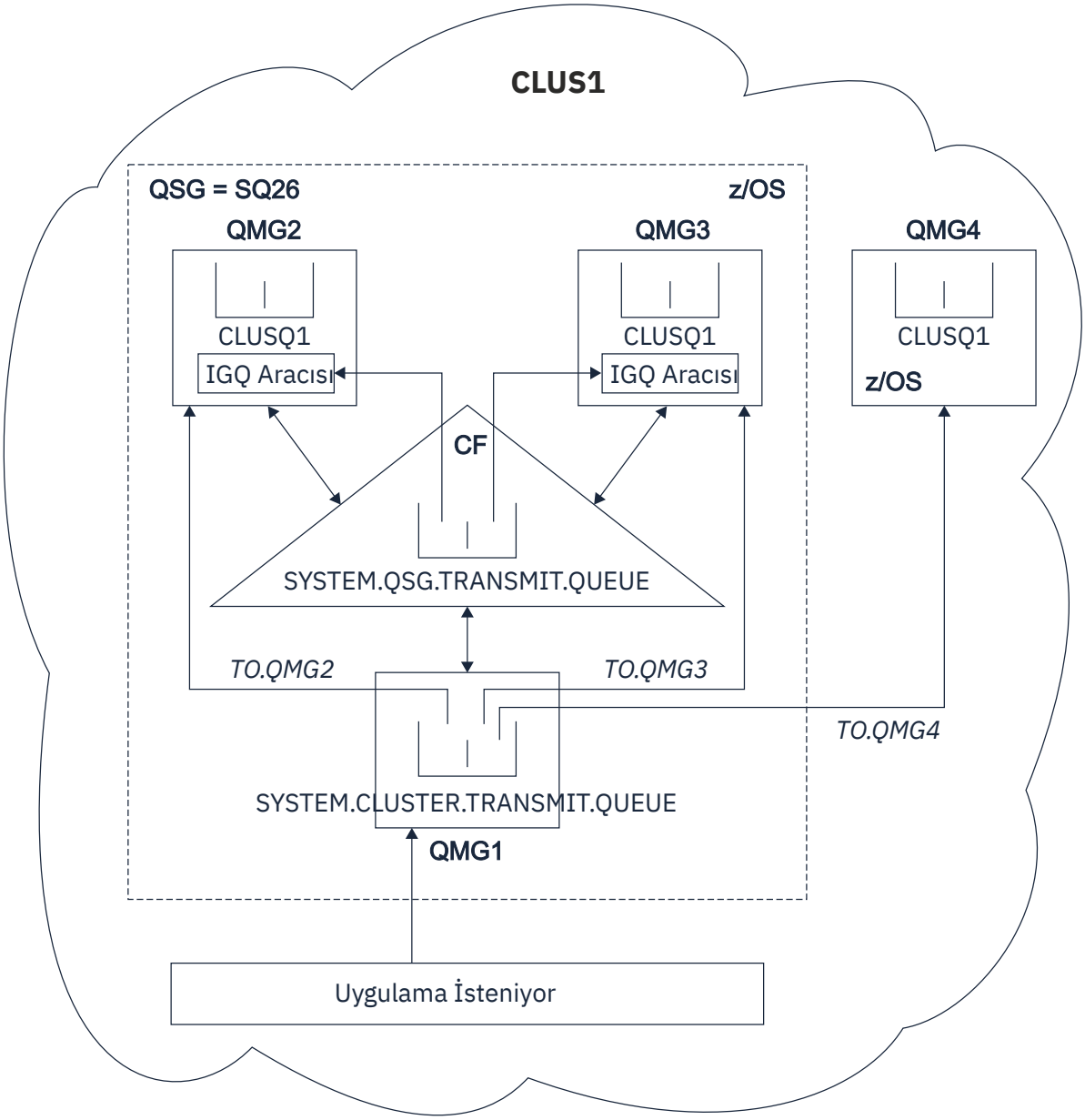
Nota işaret eder

1. İstekte bulunan uygulamanın, iletilerin teslim edilmesinde kullanılan temel mekanizmayı bilmesine gerek yoktur.
2. Küçük iletiler için potansiyel olarak daha hızlı bir ileti sağlama mekanizması elde edilebilir.
3. İleti teslimi için birden çok yol vardır (normal kanal rotası ve grup içi kuyruğa alma rotası).
4. Grup içi kuyruğa alma rotası, olası olarak daha hızlı, normal kanal rotasına göre seçilir. İleti özelliklerine bağlı olarak, ileti teslimi iki yola bölünebilir. Bu nedenle, iletiler sırasız olarak teslim edilebilir (ancak bu teslim, iletiler yalnızca normal kanal rotası kullanılarak teslim edilirse de mümkündür).
5. Bir rota seçildiğinde ve iletiler iletim kuyruklarına yerleştirildiğinde, ileti teslimi için yalnızca seçilen rota kullanılır. SYSTEM.QSG.TRANSMIT.QUEUE , XQ1iletim kuyruğuna yönlendirilmedi.

z/OS *Grup içi kuyruğa alma ile kümeleme (birden çok teslim yolu)*

Kuyruk yöneticileri, bir kümede ve bir kuyruk paylaşım grubunda olacak şekilde yapılandırılabilir.

İletiler bir küme kuyruğuna gönderildiğinde ve yerel ve uzak hedef kuyruk yöneticileri aynı kuyruk paylaşım grubunda olduğunda, küçük iletilerin teslimi için (SYSTEM.QSG.TRANSMIT.QUEUE) ve grup içi kuyruğa alma, iletinin boyutunu destekliyorsa, büyük iletilerin teslimini sağlar. Ayrıca, SYSTEM.CLUSTER.TRANSMIT.QUEUE , iletilerin kümedeki, ancak kuyruk paylaşım grubunun dışındaki herhangi bir kuyruk yöneticisine teslim edilmesi için kullanılır. Aşağıdaki şemada bu yapılandırma gösterilir (kanal başlatıcıları gösterilmez).



Şekil 70. Grup içi kuyruğa alma ile kümeleme örneği

Çizge şunları gösterir:

- CLUS1kümesinde konfigürasyonu tanımlanmış dört z/OS kuyruk yöneticisi QMG1, QMG2, QMG3ve QMG4 .
- Kuyruk yöneticileri QMG1, QMG2ve QMG3 , SQ26kuyruk paylaşım grubunda yapılandırıldı.
- QMG2 ve QMG3kuyruk yöneticileriyle çalışan IGQ aracıları.
- Yerel SYSTEM.CLUSTER.TRANSMIT.QUEUE , QMG1içinde tanımlandı.

  Not: Netlik için, SYSTEM.CLUSTER.TRANSMIT.QUEUE gösterilmiyor.
- CFLEVEL (3) RECOVER (YES) özniteliğiyle yapılandırılmış bir IBM MQ yapısında bulunan CF ' de tanımlanan paylaşılan SYSTEM.QSG.TRANSMIT.QUEUE .
- Küme kanalları TO.QMG2 (QMG1 ile QMG2arasında bağlantı kuruluyor), TO.QMG3 (QMG1 ile QMG3arasında bağlantı kuruluyor) ve TO.QMG4 (QMG1 ile QMG4arasında bağlantı kuruluyor).
- CLUSQ1 küme kuyruğu QMG2, QMG3ve QMG4kuyruk yöneticilerine yerleştiriliyor.

İstekte bulunan uygulamanın küme kuyruğunu MQOO_BIND_NOT_FIXED seçeneğiyle açtığını, böylece küme kuyruğuna ilişkin hedef kuyruk yöneticisinin koyma sırasında seçildiğini varsayın.

Seçilen hedef kuyruk yöneticisi QMG2: ise

- İstekte bulunan uygulama tarafından konan tüm büyük iletiler şunlardır:
  - SYSTEM.CLUSTER.TRANSMIT.QUEUE on QMG1, çünkü SYSTEM.QSG.TRANSMIT.QUEUE , CFLEVEL (3) yapısında olduğundan, yalnızca 63 KB ' ye kadar olan iletileri destekler.
  - Küme kanalı kullanılarak CLUSQ1 QMG2 üzerinde TO.QMG2 küme kuyruğuna aktarıldı
- İstekte bulunan uygulama tarafından konan tüm küçük iletiler
  - SYSTEM.QSG.TRANSMIT.QUEUE. Bu kuyruk, RECOVER (YES) özniteliğiyle yapılandırılmış bir yapıdadır; bu nedenle, hem kalıcı, hem de kalıcı olmayan küçük iletiler için kullanılır.
  - QMG2 üzerinde IGQ aracısı tarafından alındı
  - QMG2 üzerindeki CLUSQ1 küme kuyruğuna koy

Seçilen hedef kuyruk yöneticisi QMG4: ise:

- QMG4 , SQ26kuyruk paylaşım grubunun bir üyesi olmadığından, istekte bulunan uygulama tarafından konan tüm iletiler
  - SYSTEM.CLUSTER.TRANSMIT.QUEUE - QMG1
  - Küme kanalı kullanılarak CLUSQ1 QMG4 üzerinde TO.QMG4 küme kuyruğuna aktarıldı

Nota işaret eder

- İstekte bulunan uygulamanın, iletilerin teslim edilmesinde kullanılan temel mekanizmayı bilmesine gerek yoktur.
- Bir kuyruk paylaşım grubundaki kuyruk yöneticileri arasında (aynı kuyruk yöneticileri bir kümede olsa bile) küçük, kalıcı olmayan iletilerin aktarılması için potansiyel olarak daha hızlı bir teslim mekanizması elde edilir.
- İleti teslimi için birden çok yol vardır (küme rotası ve grup içi kuyruğa alma rotası).
- Grup içi kuyruğa alma rotası, küme rotasına göre daha hızlı olabilir. İleti özelliklerine bağlı olarak, ileti teslimi iki yola bölünebilir. Bu nedenle, iletiler sıra dışı olarak teslim edilebilir. Bu teslimatın, uygulama tarafından belirtilen MQOO_BIND_ * seçeneğine bakılmaksızın yapılabileceğine dikkat edilmelidir. Grup içi kuyruğa alma, MQOO_BIND_NOT_FIXED, MQOO_BIND_ON_OPEN, MQOO_BIND_ON_GROUP ya da MQOO_BIND_AS_Q_DEF ' nin açılışta belirtilip belirtilmediğine bağlı olarak, iletileri kümelemeyle aynı şekilde dağıtır.
- Bir rota seçildiğinde ve iletiler iletim kuyruklarına yerleştirildiğinde, ileti teslimi için yalnızca seçilen rota kullanılır. SYSTEM.QSG.TRANSMIT.QUEUE , SYSTEM.CLUSTER.TRANSMIT.QUEUE.

z/OS *Kümeleme, grup içi kuyruğa alma ve dağıtılmış kuyruğa alma*

Bir kümenin üyesi olan ve bir kuyruk paylaşım grubu olan ve bir gönderen/alıcı kanal çifti kullanılarak dağıtılmış bir kuyruk yöneticisine bağlı olan bir kuyruk yöneticisi yapılandırılabilir.

Bu yapılandırma, grup içi kuyruğa alma ve kümeleme ile grup içi kuyruğa alma işlevlerinin bir birleşimidir.

Grup içi kuyruğa alma “Grup içi kuyruğa alma ile dağıtılmış kuyruğa alma (birden çok teslim yolu)” sayfa 209içinde açıklanır.

Grup içi kuyruğa alma ile kümeleme “Grup içi kuyruğa alma ile kümeleme (birden çok teslim yolu)” sayfa 211içinde açıklanmıştır.

z/OS Grup içi kuyruğa alma iletileri

Bu bölümde, SYSTEM.QSG.TRANSMIT.QUEUE.

İleti yapısı
İletim kuyruklarına konan diğer tüm iletiler gibi, SYSTEM.QSG.TRANSMIT.QUEUE kuyruğunun başına iletim kuyruğu üstbilgisi (MQXQH) eklenir.

İleti kalıcılığı

IBM WebSphere MQ 5.3 ve üstünde, paylaşılan kuyruklar hem kalıcı hem de kalıcı olmayan iletileri destekler.

IGQ aracısı kalıcı olmayan iletileri işlerken kuyruk yöneticisi sona ererse ya da IGQ aracısı, iletilerin işlendiği ortadayken olağandışı bir şekilde sonlandıysa, işlenmekte olan kalıcı olmayan iletiler kaybolabilir. Uygulamalar, kurtarma gerekliyse, kalıcı olmayan iletilerin kurtarılmasına ilişkin düzenlemeler yapmalıdır.

IGQ aracısı tarafından yayınlanan kalıcı olmayan bir ileti için bir koyma isteği beklenmedik bir şekilde başarısız olursa, işlenmekte olan ileti kaybolur.

İletilerin teslim edilmesi
IGQ aracısı, eşitleme noktası kapsamı dışındaki tüm kalıcı olmayan iletileri ve eşitleme noktası kapsamı içindeki tüm kalıcı iletileri alır ve teslim eder. Bu durumda, IGQ aracısı eşitleme noktası koordinatörü olarak hareket eder. Bu nedenle IGQ aracısı, bir ileti kanalında hızlı, kalıcı olmayan iletilerin işlenmesi gibi kalıcı olmayan iletileri işler. Bkz. Hızlı, kalıcı olmayan iletiler.

İletilerin toplu olarak toplenmesi
IGQ aracısı, 50 iletiden oluşan sabit bir toplu iş boyutu kullanır. Bir toplu iş içinde alınan kalıcı iletiler, 50 ileti aralıklarla kesinleştirilmiştir. Aracı, SYSTEM.QSG.TRANSMIT.QUEUE.

İleti büyüklüğü
SYSTEM.QSG.TRANSMIT.QUEUE , paylaşılan kuyruklar için desteklenen ileti uzunluğu üst sınırı eksi iletim kuyruğu üstbilgisi (MQXQH) uzunluğudur.

Varsayılan ileti kalıcılığı ve varsayılan ileti önceliği
SYSTEM.QSG.TRANSMIT.QUEUE , açılışta belirlenen kuyruk adı çözme yolunda yer alır; daha sonra, varsayılan kalıcılık ve varsayılan önceliğe (ya da varsayılan kalıcılık ya da varsayılan önceliğe) sahip iletiler için, kullanılan varsayılan öncelik ve kalıcılık değerlerine sahip kuyruk seçiminde normal kurallar uygulanır. (Kuyruk seçimine ilişkin kurallar hakkında daha fazla bilgi için IBM MQ iletiler bölümüne bakın).

İlgili kavramlar

“Teslim edilmeyen/işlenmeyen iletiler” sayfa 214
Bu konuda, SYSTEM.QSG.TRANSMIT.QUEUE.

“Rapor iletileri-Grup İçi Kuyruğa Alma” sayfa 215
Bu konuda rapor iletileri açıklanır: Geliş onayı, teslim onayı, süre bitimi raporu ve kural dışı durum raporu.

z/OS *Teslim edilmeyen/işlenmeyen iletiler*

Bu konuda, SYSTEM.QSG.TRANSMIT.QUEUE.

Bir IGQ aracısı hedef kuyruğa bir ileti teslim edemezse, IGQ aracısı:

- MQRO_DISCARD_MSG rapor seçeneğini destekler (teslim edilmeyen ileti için MQMD ' nin Rapor seçenekleri alanı bunun gerekli olduğunu gösteriyorsa) ve teslim edilmeyen iletiyi atar.
- İleti henüz atılmadıysa, teslim edilmeyen iletiyi hedef kuyruk yöneticisine ilişkin gitmeyen ileti kuyruğuna yerleştirmeyi dener. IGQ aracısı, iletinin başına bir ileti kuyruğu üstbilgisi (MQDLH) ekler.

Teslim edilmeyen iletiler kuyruğu tanımlanmadıysa ya da teslim edilmeyen iletiler teslim edilmeyen iletiler kuyruğuna konamazsa ve teslim edilmeyen iletiler şu şekilde olursa:

- kalıcı, IGQ aracısı işlediği geçerli kalıcı ileti kümesini geri çevirir ve yeniden deneme durumuna girer. Daha fazla bilgi için bkz “Grup içi kuyruğa almanın belirli özellikleri” sayfa 215.
- kalıcı olmayan IGQ aracısı iletiyi atar ve sonraki iletiyi işlemeye devam eder.

Bir kuyruk paylaşım grubundaki bir kuyruk yöneticisi, ilişkili IGQ aracısının tüm iletilerini işlemek için zamanı olmadan sonlandırılırsa, işlenmemiş iletiler SYSTEM.QSG.TRANSMIT.QUEUE . IGQ aracısı daha sonra iletileri alır ve hedef kuyruklara iletir.

Bağlaşım olanağı, SYSTEM.QSG.TRANSMIT.QUEUE işlendi, işlenmemiş kalıcı olmayan iletiler kaybolur.

IBM , uygulamaların iletileri doğrudan iletim kuyruklarına koymamalarını önerir. Bir uygulama iletileri doğrudan SYSTEM.QSG.TRANSMIT.QUEUE, IGQ aracısı bu iletileri işleyemeyebilir ve bunlar SYSTEM.QSG.TRANSMIT.QUEUE. Kullanıcılar daha sonra bu işlenmemiş iletilerle başa çıkmak için kendi yöntemlerini kullanmak zorunda kalırlar.

z/OS *Rapor iletileri-Grup İçi Kuyruğa Alma*

Bu konuda rapor iletileri açıklanır: Geliş onayı, teslim onayı, süre bitimi raporu ve kural dışı durum raporu.

Varış onayı (COA) /teslim onayı (COD) rapor iletileri
COA ve COD iletileri, grup içi kuyruğa alma kullanıldığında kuyruk yöneticisi tarafından oluşturulur.

Süre bitimi raporu iletileri
Süre bitim raporu iletileri kuyruk yöneticisi tarafından oluşturulur.

Kural dışı durum raporu iletileri

Teslim edilmeyen iletiye ilişkin ileti tanımlayıcısının *Rapor seçenekleri* alanında belirtilen MQRO_EXCEPTION_ * rapor seçeneğine bağlı olarak, IGQ aracısı gerekli kural dışı durum raporunu oluşturur ve bunu belirtilen yanıt kuyruğuna yerleştirir. Grup içi kuyruğa alma, kural dışı durum raporunu hedef yanıt kuyruğuna teslim etmek için kullanılabilir.

Rapor iletisinin sürekliliği, teslim edilmeyen iletinin sürekliliğiyle aynıdır. IGQ aracısı hedef yanıt kuyruğunun adını çözemezse ya da yanıt iletisini bir iletim kuyruğuna (sonraki hedef yanıt kuyruğuna aktarma için) yerleştiremezse, kural dışı durum raporunu, rapor iletisinin oluşturulduğu kuyruk yöneticisinin kullanılmayan ileti kuyruğuna yerleştirmeyi dener. Mümkün değilse, teslim edilmeyen ileti:

- kalıcı, IGQ aracısı kural dışı durum raporunu atar, geçerli ileti kümesini geri alır ve yeniden deneme durumuna girer. Daha fazla bilgi için bkz "Grup içi kuyruğa almanın belirli özellikleri" sayfa 215.
- Kalıcı olmayan IGQ aracısı, kural dışı durum raporunu atar ve SYSTEM.QSG.TRANSMIT.QUEUE.

z/OS Grup içi kuyruğa alma güvenliği

Bu konuda, grup içi kuyruğa alma için güvenlik düzenlemeleri açıklanmaktadır.

IGQ aracısı hedef kuyrukları açtığında gerçekleştirilen güvenlik denetimi düzeyini denetlemek için kuyruk yöneticisi öznitelikleri IGQAUT (IGQ yetkisi) ve IGQUSER (IGQ aracısı kullanıcı kimliği) ayarlanabilir.

Grup içi kuyruğa alma yetkisi (IGQAUT)

IGQAUT özniteliği, gerçekleştirilecek güvenlik denetimlerinin tipini gösterecek şekilde ayarlanabilir ve bu nedenle, IGQ aracısı hedef kuyruğa ileti koyma yetkisi verdiğinde IGQ aracısı tarafından kullanılacak kullanıcı kimliklerini belirlemek için kullanılabilir.

IGQAUT özniteliği, kanal tanımlarında bulunan PUTAUT özniteliğine benzer.

Grup içi kuyruğa alma kullanıcı kimliği (IGQUSER)

IGQUSER özniteliği, iletileri hedef kuyruğa koyma yetkisi verdiğinde IGQ aracısı tarafından kullanılacak bir kullanıcı kimliğini aday göstermek için kullanılabilir.

IGQUSER özniteliği, kanal tanımlarında bulunan MCAUSER özniteliğine benzer.

z/OS Grup içi kuyruğa almanın belirli özellikleri

Bu kısımda, grup içi kuyruğa alma aracısının nesne tanıtıcılarının geçersizliğini denetleme, kendi kendine kurtarma ve yeniden deneme yeteneği ve grup içi kuyruğa alma aracısı ve diziselleştirme de içinde olmak üzere, grup içi kuyruğa alma işleminin belirli özellikleri açıklanmaktadır.

Nesne tanıtıcılarının geçersiz kılınması (MQRC_OBJECT_CHANGED)

Nesne açıldıktan sonra bir nesnenin özniteliklerinin değiştiği bulunursa, kuyruk yöneticisi nesne tanıtıcısını sonraki kullanımında MQRC_OBJECT_CHANGED ile geçersiz kılar.

Grup içi kuyruğa alma, nesne tanıtıcısı geçersiz kılma için aşağıdaki kuralları sunar:

- SYSTEM.QSG.TRANSMIT.QUEUE açık işleme sırasında ad çözme yolunda yer aldıysa, çünkü grup içi kuyruğa alma açık zamanda ETKINLEşTIRILMIşTI, ancak grup içi kuyruğa alma işleminin yerleştirme sırasında DEVRE Dışı olduğu saptandıysa, kuyruk yöneticisi nesne tanıtıcısını geçersiz kılar ve MQRC_OBJECT_CHANGED ile koyma isteğini başarısız olur.
- SYSTEM.QSG.TRANSMIT.QUEUE , açık işleme sırasında ad çözme yoluna dahil edilmedi; bunun nedeni, grup içi kuyruğa alma işleminin açık zamanda DISABLED olması, ancak grup içi kuyruğa alma işleminin koyma sırasında ENABLED olması, kuyruk yöneticisinin nesne tanıtıcısını geçersiz kılması ve MQRC_OBJECT_CHANGED ile koyma isteğini başarısız olmasıdır.
- SYSTEM.QSG.TRANSMIT.QUEUE , açık işleme sırasında grup içi kuyruğa alma etkinleştirildiği, ancak SYSTEM.QSG.TRANSMIT.QUEUE tanımlamasının koyma süresine göre değiştiği için ad çözme yoluna eklendiyse, kuyruk yöneticisi nesne tanıtıcısını geçersiz kılar ve MQRC_OBJECT_CHANGED ile koyma isteğini başarısız olur.

Grup içi kuyruğa alma aracısının kendi kendine kurtarılması

IGQ aracısı olağandışı sona ererse, CSQM067E iletisi yayınlanır ve IGQ aracısı yeniden başlar.

Grup içi kuyruğa alma aracısının yeniden deneme yeteneği

IGQ aracısı SYSTEM.QSG.TRANSMIT.QUEUE (örneğin, tanımlanmadığı ya da yanlış özniteliklerle tanımlandığı ya da Gets için ya da başka bir nedenle engellendiği için), IGQ aracısı yeniden deneme durumuna geçer.

IGQ aracısı, kısa ve uzun yeniden deneme sayılarını ve aralıklarını gözlemler. Değiştirilemeyen bu sayılara ve aralıklara ilişkin değerler aşağıdaki gibidir:

Çizelge 20. 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 dak

Grup içi kuyruğa alma aracısı ve diziselleştirme

Eşdüzey kurtarma devam ederken IGQ aracısının paylaşılan kuyruklara erişimi diziselleştirme girişimi başarısız olabilir.

IGQ aracısı, paylaşılan bir kuyruktaki ya da kuyruklardaki kesinleştirilmemiş iletilerle uğraşırken bir kuyruk paylaşım grubunda bir kuyruk yöneticisinde hata oluşursa, IGQ aracısı sona erer ve başarısız olan kuyruk yöneticisi için paylaşılan kuyruk eşdüzey kurtarma işlemi gerçekleşir. Paylaşılan kuyruk eşdüzey kurtarma zamanuyumsuz bir etkinlik olduğundan, bu, başarısız olan kuyruk yöneticisi ve bu kuyruk yöneticisine ilişkin IGQ aracısının, paylaşılan kuyruk eşdüzey kurtarma işlemi tamamlanmadan önce yeniden başlatılması olasılığını bırakır. Bu da, iletilerin hala kurtarılmaya devam edilirken, kesinleştirilmiş iletilerin sırasız ve sırasız olarak işlenmesi olasılığını ortadan kalkar. İletilerin sırasız olarak işlenmemesini sağlamak için IGQ aracısı, MQCONNX API çağrısını yayınlayarak paylaşılan kuyruklara erişimi diziselleştirir.

Eşdüzey kurtarma devam ederken IGQ aracısının paylaşılan kuyruklara erişimi diziselleştirme girişimi başarısız olabilir. Bir hata iletisi yayınlandı ve IGQ aracısı yeniden deneme durumuna alındı. Örneğin, kuyruk yöneticisi eşdüzey kurtarma işlemi tamamlandığında, IGQ aracısı başlatılabilir.

z/OS z/OS üzerinde 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 kullandığına ilişkin daha fazla ayrıntı sağlar.

İlgili kavramlar

"IBM MQ for z/OS için sayfa kümeleri" sayfa 217
IBM MQ for z/OS ' nın iletileri saklamak için sayfa kümelerini nasıl kullandığını anlamak için bu konuyu kullanın.

"IBM MQ for z/OS için depolama sınıfları" sayfa 218
Depolama sınıfı, kuyruk yöneticisinin kuyrukları sayfa kümeleriyle eşlemesine olanak sağlayan bir IBM MQ for z/OS kavramıdır. Hangi veri kümelerinin hangi kuyruklar tarafından kullanılacağını denetlemek için depolama sınıflarını kullanabilirsiniz.

"IBM MQ for z/OS için arabellekler ve arabellek havuzları" sayfa 219
IBM MQ for z/OS , verileri geçici olarak önbelleğe almak için arabellekleri ve arabellek havuzlarını kullanır. Arabelleklerin nasıl düzenlendiğini ve kullanıldığını daha iyi anlamak için bu konuyu kullanın.

İlgili başvurular

"IBM MQ for z/OS için depolama yönetimi hakkında daha fazla bilgi nerede bulunur?" sayfa 221
IBM MQ for z/OSdepolama yönetimi hakkında daha fazla bilgi edinmek için bu konuyu başvuru olarak kullanın.

z/OS IBM MQ for z/OS için sayfa kümeleri

IBM MQ for z/OS ' nın 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 kümeidir. Çoğu iletiyi ve nesne tanımlamasını saklamak için sayfa kümeleri kullanılır.

Bunun kural dışı durumları, Db2üzerindeki paylaşılan bir havuzda saklanan genel tanımlamalar ve paylaşılan kuyruklardaki iletilerdir. 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 bkz. Özel ve genel tanımlar.

IBM MQ sayfa kümelerinin boyutu en çok 64 GB olabilir. Her 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 yöneticisiyle ilgili diğer önemli bilgileri saklamak için sayfa kümesi sıfır (PSID=00) değerini kullanır. IBM MQ ' in olağan çalışması için, sıfır sayfa kümesinin dolu olmaması önemlidir, bu nedenle iletileri saklamak için bunu kullanmayın.

Sisteminizin performansını artırmak için kısa ömürlü iletileri farklı sayfa kümelerine yerleştirerek uzun ömürlü iletilerden ayırmanız da gerekir.

Sayfa kümelerini biçimlendirmeniz gerekir ve IBM MQ bunun için bir FORMAT yardımcı programı sağlar; bkz. Sayfa kümelerini biçimlendirme (FORMAT). Sayfa kümeleri IBM MQ altsistemine de tanımlanmalıdır.

IBM MQ for z/OS , bir sayfa kümesi dolduğunda dinamik olarak genişletilecek şekilde yapılandırılabilir. IBM MQ , yeterli disk depolama alanı varsa, 123 mantıksal kapsam bulununcaya kadar sayfa kümesini genişletmeye devam eder. Doğrusal veri kümesi bu şekilde tanımlanırsa kapsamlar birimlere yayılabilir, ancak IBM MQ sayfa kümelerini 64 GB ' den fazla genişletemez.

Farklı bir IBM MQ kuyruk yöneticisinde bir IBM MQ kuyruk yöneticisindeki sayfa kümelerini kullanamaz ya da kuyruk yöneticisi adını değiştiremezsiniz. Verileri bir kuyruk yöneticisinden diğerine aktarmak

istiyorsanız, tüm nesneleri ve iletileri ilk kuyruk yöneticisinden kaldırıp başka bir kuyruk yöneticisine yeniden yüklemeniz gerekir.

V6'dan önceki bir yayın düzeyini çalıştıran bir kuyruk yöneticisinde 4 GB' den büyük sayfa kümeleri kullanılamaz. Geçiş süresi boyunca, önceki bir kod yayınına geri dönmeniz gerekebilir:

- Sayfa kümesi 0 değerini 4 GB ' den büyük olacak şekilde değiştirmeyin.
- Önceki bir yayın düzeyiyle bir kuyruk yöneticisi yeniden başlatılırken, 4 GB ' den büyük diğer sayfa kümeleri çevrimdışı bırakılır.

4 GB 'den fazla genişletme yeteneğine sahip var olan sayfa kümelerinin geçirilmesiyle ilgili daha fazla bilgi için 4 GB' den büyük bir sayfa kümesi tanımlamabaşlıklı konuya bakın.

Bir denetimci, çalışmakta olan bir kuyruk yöneticisine devingen olarak sayfa kümeleri ekleyebilir ya da çalışmakta olan bir kuyruk yöneticisinden (sayfa kümesi sıfır dışında) sayfa kümelerini kaldırabilir. DEFINE PSID komutu, kuyruk yöneticisinin yeniden başlatılması tamamlandıktan sonra, yalnızca komut DSN anahtar sözcüğünü içeriyorsa çalıştırılabilir.

z/OS IBM MQ for z/OS için depolama sınıfları

Depolama sınıfı, kuyruk yöneticisinin kuyrukları sayfa kümeleriyle eşlemesine olanak sağlayan bir IBM MQ for z/OS kavramıdır. Hangi veri kümelerinin hangi kuyruklar tarafından kullanılacağını denetlemek için depolama sınıflarını kullanabilirsiniz.

Depolama sınıfları tanıtılıyor

Depolama sınıfı , bir ya da daha çok kuyruğu bir sayfa kümesiyle eşleyen bir sınıftır. Bu, o kuyruğa ilişkin iletilerin o sayfa kümesinde saklandığı anlamına gelir.

Depolama sınıfları, yönetim, veri kümesi alanı ve yük yönetimi ya da uygulama yalıtma amaçları için paylaşılmayan ileti verilerinin nerede saklandığını denetlemenizi sağlar. IMS köprüsünü kullanıyorsanız, bir IMS bölgesinin XCF grubunu ve üye adını tanımlamak için depolama sınıflarını da kullanabilirsiniz (bkz. “IBM MQ ve IMS” sayfa 273).

Paylaşılan kuyruklar, sayfa kümesi eşlemesi elde etmek için depolama sınıflarını kullanmaz; bu kuyruklardaki iletiler sayfa kümelerinde saklanmaz.

Depolama sınıfları nasıl çalışır?

- DEFINE STGCLASS komutunu kullanarak, bir sayfa kümesi tanıtıcısı (PSID) belirterek bir depolama sınıfı tanımlarsınız.
- Bir kuyruk tanımladığınızda, STGCLASS özniteliğinde depolama sınıfını belirtirsiniz.

Aşağıdaki örnekte, QE5 yerel kuyruğu, ARC2depolama sınıfı aracılığıyla sayfa kümesi 21 ile eşlenmiştir.

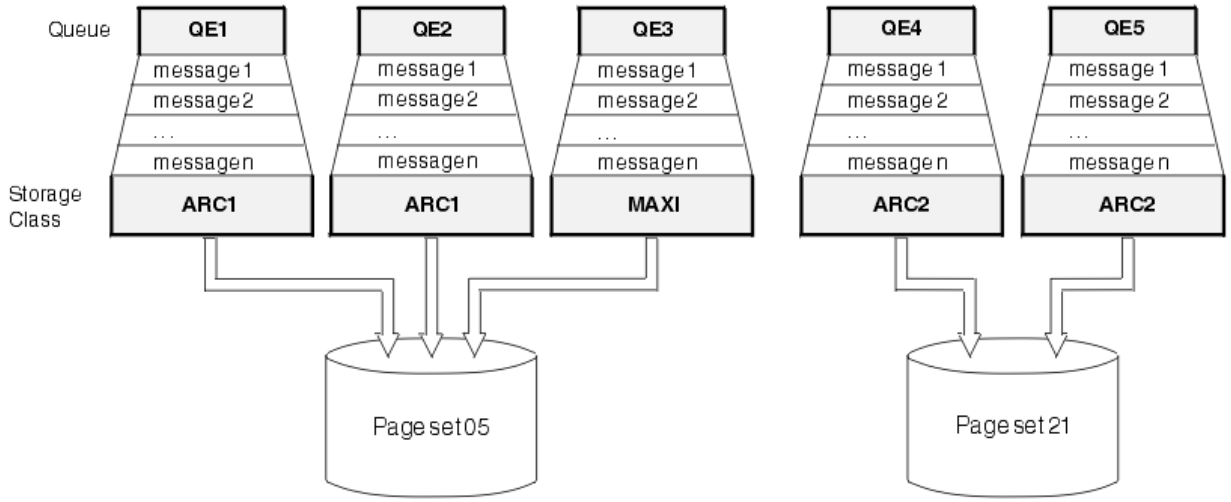
```
DEFINE STGCLASS(ARC2) PSID(21)
DEFINE QLOCAL(QE5) STGCLASS(ARC2)
```

Bu, QE5 kuyruğuna konan iletilerin sayfa kümesi 21 'de (kuyrukta DASD ' ye yazılacak kadar uzun kalırlarsa) saklanacağı anlamına gelir.

Aynı depolama sınıfını birden çok kuyruk kullanabilir ve istediğiniz kadar depolama sınıfı tanımlayabilirsiniz. Örneğin, önceki örneği aşağıdaki gibi daha fazla depolama sınıfı ve kuyruk tanımı içerecek şekilde 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 71 sayfa 219içinde, ARC1 ve MAXI depolama sınıfları sayfa kümesi 05 ile ilişkilendirilir. Bu nedenle, QE1, QE2ve QE3 kuyrukları sayfa kümesi 05 ile eşlenir. Benzer şekilde, ARC2 depolama sınıfı QE4 ve QE5 kuyruklarını sayfa kümesi 21 ile ilişkilendirir.



Şekil 71. Depolama sınıfları aracılığıyla kuyrukların sayfa kümeleriyle eşlenmesi

Depolama sınıfı belirtmeden bir kuyruk tanımlarsanız, IBM MQ varsayılan bir depolama sınıfı kullanır.

Var olmayan bir depolama sınıfını belirten bir kuyruğa ileti konursa, uygulama bir hata alır. Kuyruk tanımlamasını var olan bir depolama sınıfı adı verecek şekilde değiştirmeli ya da kuyruk tarafından adlandırılan depolama sınıfını yaratmalısınız.

Bir depolama sınıfını yalnızca aşağıdaki zamanlarda değiştirebilirsiniz:

- Bu depolama sınıfını kullanan tüm kuyruklar boş ve kesinleştirilmemiş etkinliği yok.
- Bu depolama sınıfını kullanan tüm kuyruklar kapatılır.

z/OS IBM MQ for z/OS için arabellekler ve arabellek havuzları

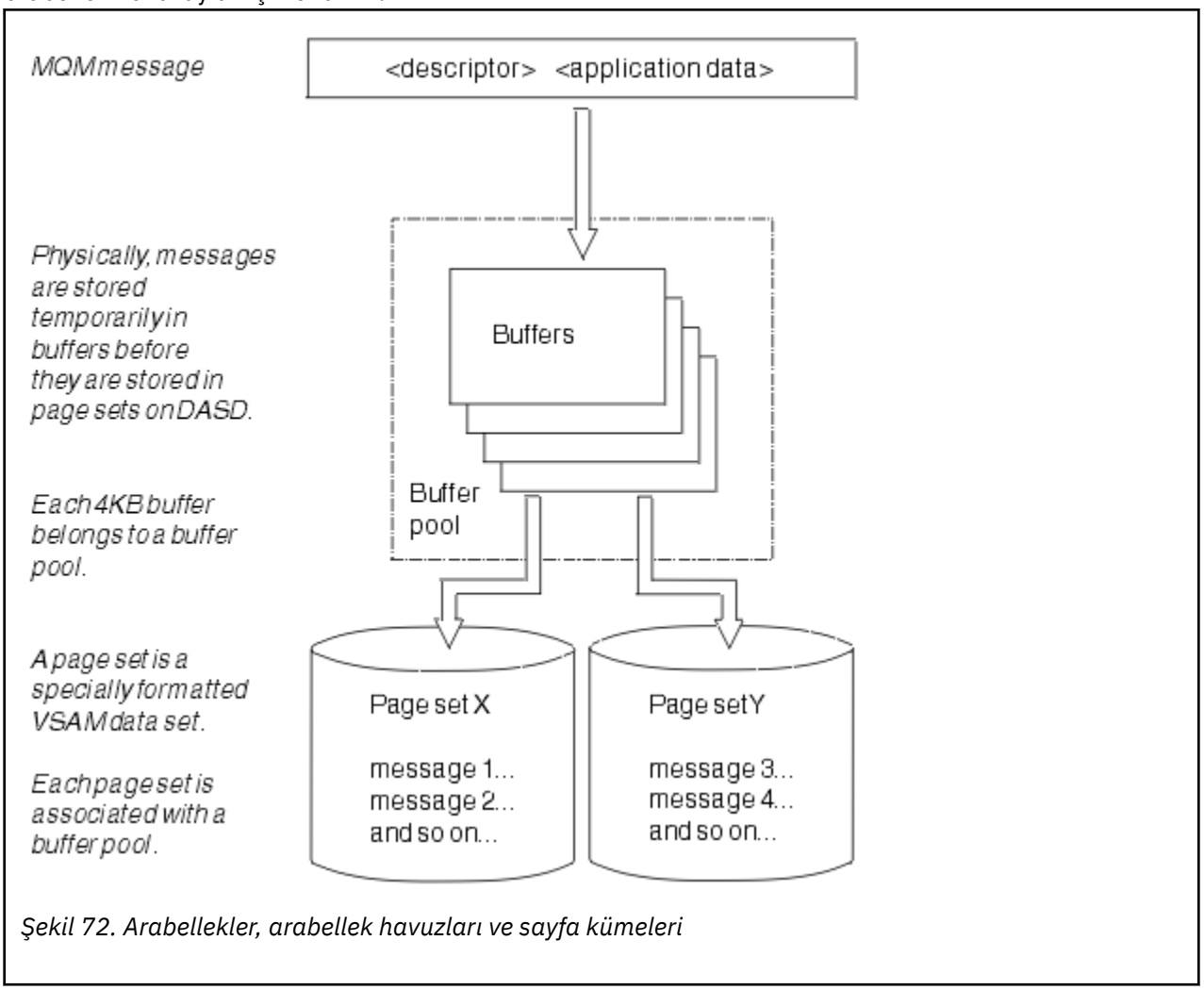
IBM MQ for z/OS , verileri geçici olarak önbelleğe almak için arabellekleri ve arabellek havuzlarını kullanır. Arabelleklerin nasıl düzenlendiğini ve kullanıldığını daha iyi anlamak için bu konuyu kullanın.

Verimlilik için IBM MQ , iletilerin (ve nesne tanımlarının) DASD ' deki sayfa kümelerinde saklanmadan önce geçici olarak arabelleklerde saklandığı bir önbelleğe alma biçimi kullanır. Kısa ömürlü iletiler (yani, alındıktan kısa bir süre sonra kuyruktan alınan iletiler) yalnızca arabelleklerde saklanabilir. Bu önbelleğe alma etkinliği, IBM MQ' un bir bileşeni olan bir arabellek yöneticisi tarafından denetlenir.

Arabellekler *arabellek havuzları*içinde düzenlenir. Her kuyruk yöneticisi için en çok 100 arabellek havuzu (0-99) tanımlayabilirsiniz.

Şekil 72 sayfa 220içinde özetlenen nesne ve ileti tipi ayrımı ve uygulamanızın sahip olabileceği veri yalıtımı gereksinimleri ile tutarlı olarak en az sayıda arabellek havuzu kullanmanız önerilir. Her arabellek 4 KB uzunlukta. Arabellek havuzları varsayılan olarak 31 bit saklama alanı kullanır; bu kipte arabellek sayısı üst sınırı, kuyruk yöneticisi adres alanında kullanılabilir 31 bit saklama alanı miktarına göre belirlenir; arabellekler için yaklaşık %70 'i geçmez. Alternatif olarak, arabellek havuzu saklama alanı ayırma 64 bit saklama alanından yapılabilir (**DEFINE BUFFPOOL** komutunun LOCATION özniteliğini kullanın). 64 bit depolamanın kullanılması için LOCATION (ABOVE) kullanılması iki yarar sağlar. İlk olarak, çok daha fazla 64 bit saklama alanı vardır, bu nedenle arabellek havuzları çok daha büyük olabilir ve ikinci olarak, 31 bit saklama alanı diğer işlevler tarafından kullanılabilir hale getirilir. Genellikle, arabellekleriniz ne kadar çok olursa, arabelleğe alma o kadar verimli olur ve IBM MQbaşarımı o kadar yüksek olur.

Şekil 72 sayfa 220 içinde iletiler, arabellekler, arabellek havuzları ve sayfa kümeleri arasındaki ilişki gösterilir. Bir arabellek havuzu bir ya da daha çok sayfa kümesiyle ilişkilendirilir; her sayfa kümesi tek bir arabellek havuzuyla ilişkilendirilir.



Şekil 72. Arabellekler, arabellek havuzları ve sayfa kümeleri

ALTER BUFFPOOL komutunu kullanarak, arabellek havuzu büyüklüğünü ve konumunu değiştirmek için dinamik olarak komut verebilirsiniz. Sayfa kümeleri, **DEFINE PSID** komutu kullanılarak dinamik olarak eklenebilir ya da **DELETE PSID** komutu kullanılarak silinebilir.

Bir arabellek havuzu çok küçükse, IBM MQ CSQP020Eiletisini yayınlar. Bundan sonra, etkilenen arabellek havuzuna devingen olarak daha fazla arabellek ekleyebilirsiniz (bunu yapmak için diğer arabellek havuzlarından arabellekleri kaldırmanız gerekebilir).

DEFINE BUFFPOOL komutuyla bir havuzdaki arabellek sayısını belirtirsiniz ve arabellek havuzlarını **ALTER BUFFPOOL** komutuyla dinamik olarak yeniden boyutlandırabilirsiniz. Bir havuzdaki yürürlükteki arabellek sayısını, **DISPLAY USAGE** komutunu kullanarak arabellek havuzunu kullanan bir sayfa kümesini dinamik olarak görüntüleyerek saptarsınız.

Başarımla ilgili nedenlerle, iletileri ve nesne tanımlamalarını aynı arabellek havuzuna koymayın. Nesne tanımlamalarının saklandığı sayfa kümesi sıfır için tek bir arabellek havuzu (örneğin sıfır) kullanın. 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 başlatmadan sonra **DEFINE BUFFPOOL** komutu kullanılamaz. Bunun yerine, bir **DEFINE PSID** komutu DSN anahtar sözcüğünü kullanıyorsa, tanımlı olmayan bir arabellek havuzunu belirtik olarak belirleyebilir. Bu yeni arabellek havuzu yaratılır.

z/OS IBM MQ for z/OS için depolama yönetimi hakkında daha fazla bilgi nerede bulunur?

IBM MQ for z/OSdepolama yönetimi hakkında daha fazla bilgi edinmek için bu konuyu başvuru olarak kullanın.

Bu bölümdeki konularla ilgili daha fazla bilgiyi aşağıdaki kaynaklardan bulabilirsiniz:

Çizelge 21. Depolama yönetimi hakkında daha fazla bilgi nerede bulunur?

Konu	Nereye bakıyorsunuz?
Ne kadar depolama alanına ihtiyacınız var?	z/OS üzerinde depolama ve performans gereksinimlerinizin planlanması
Sayfa kümelerinizi ne kadar büyük yapacağınızı ve arabellek havuzları	Sayfa kümelerinizi ve arabellek havuzlarınızı planlayın
Sayfa kümelerini yönetme	Sayfa kümelerini yönetme
MQSC komutları	MQSC komutları

z/OS oturum açmaIBM MQ for z/OS

IBM MQ veri değişikliklerinin ve önemli olayların *günlüklerini* , bunlar oluştuğunda tutar. Bu günlükler, gerektiğinde verileri önceki bir duruma kurtarmak için kullanılabilir.

Önyükleme veri kümesi (BSDS), günlükleri içeren veri kümeleriyle ilgili bilgileri saklar.

Günlük, istatistik, izleme ya da performans değerlendirmesi için bilgi içermiyor. IBM MQ ' in topladığı istatistiksel bilgiler ve izleme bilgileri hakkında daha fazla ayrıntı için bkz. İzleme ve istatistikler.

Günlüğe kaydetme hakkında daha fazla bilgi için aşağıdaki konulara bakın:

- “IBM MQ for z/OS içindeki günlük dosyaları” sayfa 222
- “Günlüğün nasıl yapılandırıldığını” sayfa 226
- “IBM MQ for z/OS günlüklerinin yazılması” sayfa 226
- “Daha büyük günlük göreli byte adresi” sayfa 229
- “Önyükleme veri kümesi” sayfa 230

İlgili görevler

Günlük kaydı ortamınızın planlanması

Sistem parametre modülü kullanılarak günlüklerin ayarlanması

z/OS Yönetmez/OS

İlgili başvurular

z/OS IBM MQ for z/OS ile ilgili iletiler

z/OS IBM MQ for z/OS içindeki günlük dosyaları

Günlük dosyaları, hareket kurtarma için gerekli bilgileri içerir. Günlük verilerini uzun bir süre saklayabilmeniz için etkin günlük dosyaları arşivlenebilir.

Günlük dosyası nedir?

IBM MQ , önemli olayların tümünü bir *etkin günlüğe*kaydeder. Günlük, kurtarmak için gereken bilgileri içerir:

- Kalıcı iletiler
- IBM MQ nesneleri; örneğin, kuyruklar
- IBM MQ kuyruk yöneticisi

Etkin günlük, döngüsel olarak kullanılan veri kümelerinden (310 'a kadar) oluşur.

Etkin bir günlük bir arşiv veri kümesinde bir kopya doldurması için günlük arşivlemeyi etkinleştirebilirsiniz. Arşivlemenin kullanılması, günlük verilerini uzun bir süre saklamanızı sağlar. Arşivlemeyi kullanmıyorsanız, günlüklerin kaydırılması ve önceki verilerin üzerine yazılır. Bir sayfa kümesini kurtarmak ya da CF yapısındaki verileri kurtarmak için, sayfa kümesinin ya da yapının yedeklenmesinden itibaren günlük verilerine gereksinim duyarsınız. Diskte ya da manyetik bantta bir arşiv günlüğü yaratılabilir.

Arşivleme

Etkin günlüğün sabit bir boyutu olduğu için IBM MQ , her günlük veri kümesinin içeriğini düzenli olarak bir *arşiv günlüğüne*kopyalar; bu, genellikle doğrudan erişimli bir depolama aygıtında (DASD) ya da bir manyetik bantta bulunan bir veri kümesidir. Bir altsistem ya da hareket hatası varsa, IBM MQ etkin günlüğü ve gerekiyorsa, kurtarma için arşiv günlüğünü kullanır.

Arşiv günlüğü en çok 1000 sıralı veri kümesi içerebilir. Her veri kümesini, z/OS tümleşik katalog olanağını (ICF) kullanarak kataloğa alabilirsiniz.

Arşivleme, IBM MQ kurtarmasının önemli bir bileşenidir. Bir kurtarma birimi uzun süreli bir birimse, o kurtarma birimindeki günlük kayıtları arşiv günlüğünde bulunabilir. Bu durumda, kurtarma işlemi için arşiv günlüğünden veri alınması gerekir. Ancak, arşivleme devre dışı bırakılırsa, yeni günlük kayıtlarıyla etkin günlük kayarak önceki günlük kayıtlarının üzerine yazılır. Bu, IBM MQ ' in kurtarma birimini geri alamayabileceği ve iletilerin kaybolabileceği anlamına gelir. Kuyruk yöneticisi daha sonra olağandışı sona erer.

Bu nedenle, bir üretim ortamında **arşivlemeyi hiçbir zaman kapatma**. Bunu yaparsanız, bir sistem ya da işlem hatasından sonra veri kaybı riskini çalıştırırsınız. Yalnızca bir test ortamında çalışıyorsanız, arşivlemeyi kapatmayı düşünebilirsiniz. Bunu yapmanız gerekiyorsa, CSQ6LOGP makrosunu kullanın; bu makroyu CSQ6LOGPiçinde bulabilirsiniz.

Planlanmamış uzun süreli iş birimlerinde sorunların önlenmesine yardımcı olmak için IBM MQ bir ileti yayınlar (CSQJ160I ya da CSQJ161I) etkin günlük boşaltma işlemi sırasında uzun süreli bir iş birimi saptanırsa.

İkili günlük kaydı

İkili günlük kaydında, her günlük kaydı, yeniden başlatma sırasında veri kaybı sorunları olasılığını en aza indirmek için iki farklı etkin günlük veri kümesine yazılır.

IBM MQ ' yi *tek günlük kaydı* ya da *ikili günlük kaydı*ile çalışacak şekilde yapılandırabilirsiniz. Tek günlük kaydı ile, günlük kayıtları etkin günlük veri kümesine bir kez yazılır. Her etkin günlük veri kümesi, tek kapsam VSAM doğrusal veri kümesidir (LDS). İkili günlük kaydı ile, her günlük kaydı iki farklı etkin günlük veri kümesine yazılır. İkili günlük kaydı, yeniden başlatma sırasında veri kaybı sorunları olasılığını en aza indirir.

Günlük shunting

Günlük kaydı, bazı iş birimlerine ilişkin günlük kayıtlarının günlüğe daha sonra yazılmasına neden olur. Bu, uzun süreli ya da uzun süreli belirsiz iş birimleri için kuyruk yöneticisi yeniden başlatıldığında okunması gereken günlük verisi miktarını azaltır.

Bir iş birimi uzun olarak değerlendirildiğinde, her günlük kaydının bir gösterimi günlüğün daha aşağısına yazılır. Bu teknik, *shunting*(shunting) olarak bilinir. Tüm iş birimi işlendiğinde, iş birimi *shunted* (uzak) durumunda olur. Uzak iş birimiyle ilgili herhangi bir yedekleme ya da yeniden başlatma etkinliği, özgün iş birimi günlük kayıtlarını kullanmak yerine, gizli günlük kayıtlarını kullanabilir.

Uzun süreli bir iş biriminin saptanması, denetim noktası işleminin bir işlevidir. Denetim noktası zamanında, her etkin iş birimi, geri çekilmesi gerekip gerekmediğini belirlemek için denetlenir. İş birimi, yaratıldığından bu yana iki önceki denetim noktasından geçtiyse ya da en son geri alındığından bu yana, iş birimi yok edilmeye uygundur. Bu, tek bir iş biriminin birden çok kez geri çekilebileceği anlamına gelir. Bu, *çok yönlü* iş birimi olarak bilinir.

Her üç kontrol noktasında bir iş birimi geri çekilir. Ancak, denetim noktası günlük anahtarında zamanuyumsuz olarak gerçekleştirilir (ya da LOGLOAD işleminin aşılmasına neden olan günlük kaydının yazılması).

Bir kerede yalnızca tek bir denetim noktası var, bu nedenle bir denetim noktası tamamlanmadan önce birden çok günlük anahtarı olabilir.

Bu, yeterli sayıda etkin günlük yoksa ya da çok küçükse, tüm günlükler doldurulmadan önce büyük bir iş biriminin atılması tamamlanmayabilir anlamına gelir.

Shunting tamamlanamıyorsa, CSQR027I iletisi görüntülenir.

Günlük arşivleme kapalıysa, 00D1032A nedeniyle ABEND 5C6 ortaya çıkar; bunun nedeni, geri çevirme işleminin başarısız olduğu iş birimini geri alma girişiminde bulunulmasıdır. Bu sorunu önlemek için OFFLOAD=YES kullanmalısınız.

Günlük kaydetme her zaman etkindir ve günlük arşivlemenin etkinleştirilip etkinleştirilmediğini çalıştırır.

Not: Bir iş birimine ilişkin tüm günlük kayıtları dışlansa da, her kaydın tüm içeriği dışlanmaz, yalnızca geriletme için gerekli olan kısım kapatılır. Bu, yazılan günlük verisi miktarının minimum düzeyde tutulduğu ve bir sayfa kümesi hatası oluştuğunda gizli kayıtların kullanılamayacağı anlamına gelir. Uzun süre çalışan bir iş birimi, üçten fazla kuyruk yöneticisi denetim noktası için çalışan bir birimdir.

Günlük ayırmayla ilgili daha fazla bilgi için Günlüklerin yönetilmesibaşlıklı konuya bakın.

Günlük sıkıştırması

Günlük kayıtlarını, günlük veri kümesinden yazılırken ve okunurken sıkıştırmak ve açmak için IBM MQ for z/OS ' i yapılandırabilirsiniz.

Günlük sıkıştırma, özel kuyruklardaki kalıcı iletiler için günlüğe yazılan veri miktarını azaltmak için kullanılabilir. Ulaşılan sıkıştırma miktarı, iletilerde bulunan veri tipine bağlıdır. Örneğin, Run Length Encoding (RLE), yapılandırılmış ya da kayıt odaklı veriler için iyi sonuçlar verebilecek yinelenen bayt örneklerini sıkıştırarak çalışır.

Uyarı: Paylaşılan bir kuyruğa konan kalıcı iletiler günlük sıkıştırmasına tabi değildir.

Ne kadar veri sıkıştırması elde edildiğini izlemek için Sistem Yönetimi Olanağı 115 (SMF) kayıtlarının Günlük Yöneticisi bölümündeki alanları kullanabilirsiniz. SMF hakkında daha fazla bilgi için bkz. Sistem Yönetimi Olanağının Kullanılması ve Muhasebe ve istatistik iletileri.

Günlük sıkıştırma, sistemin işlemci kullanımını artırır. Yalnızca kuyruk yöneticinizin verimi, günlük veri kümelerine yazma GÇ bant genişliği tarafından kısıtlandıysa ya da günlük veri kümelerini tutmak için gereken disk depolama alanı tarafından kısıtlandıysanız, sıkıştırmayı kullanmayı düşünmeniz gerekir. Paylaşılan kuyruklar kullanıyorsanız, kuyruk paylaşım grubuna ek kuyruk yöneticileri eklenerek ve iş yükü daha fazla kuyruk yöneticilerine dağıtılarak GÇ bant genişliği kısıtlamaları hafifletilir.

Günlük sıkıştırma seçeneği, kuyruk yöneticisini durdurma ve yeniden başlatma gereksinimi olmadan gerektiği şekilde etkinleştirilebilir ve devre dışı bırakılabilir. Kuyruk yöneticisi, yürürlükteki günlük sıkıştırma ayarından bağımsız olarak, sıkıştırılmış günlük kayıtlarını okuyabilir.

Kuyruk yöneticisi, günlük sıkıştırma için 3 ayarı destekler.

YOK
Günlük verileri sıkıştırması kullanılmaz. Bu varsayılan değerdir.

RLE
Günlük veri sıkıştırması, çalıştırma uzunluğu kodlaması (RLE) kullanılarak gerçekleştirilir.

Fark Etmez
Kuyruk yöneticisinin en yüksek günlük kaydı sıkıştırma derecesini veren sıkıştırma algoritmasını seçmesini sağlayın. Bu seçenek RLE sıkıştırmayla sonuçlanır.

Günlük kayıtlarının sıkıştırmasını aşağıdakilerden birini kullanarak denetleyebilirsiniz:

- MQSC ' de SET ve DISPLAY LOG komutları; bkz. SET LOG ve DISPLAY LOG
- PCF arabiriminde Set Log and Inquire Log işlevleri; Set log ve Inquire log başlıklı konuya bakın.
- Sistem değiştirgesi modülündeki CSQ6LOGP makrosu; bkz. CSQ6LOGP

Buna ek olarak, CSQ1LOGP günlük yazdırma yardımcı programı, sıkıştırılmış günlük kayıtlarının genişletilmesini de destekler.

Günlük verileri

Günlük en çok 18 milyon ($1.8 * 10^{19}$) içerebilir bayt. Her bayt, günlüğün başından itibaren göreli konumuyla adreslenir ve bu göreli konum, *göreli bayt adresi* (RBA) olarak bilinir.

RBA 'ya, 6 baytlık ya da 8 baytlık günlük RBA' larının kullanımda olup olmadığına bağlı olarak, toplam adreslenebilir 2^{48} bayt ya da 2^{64} bayt veren 6 baytlık ya da 8 baytlık bir alan tarafından başvurulur.

Ancak IBM MQ , kullanılan aralığın F00000000000 (6 baytlık RNA 'lar kullanılıyorsa) ya da FFFF800000000000 (8 baytlık günlük RNA' ları kullanılıyorsa) ötesinde olduğunu saptadığında, CSQI045, CSQI046, CSQI047ve CSQJ032 iletileri yayınlanır ve günlük RBA 'sini sıfırlamanız için sizi uyarır.

RBA değeri FFF800000000 (6 baytlık günlük RNA 'ları kullanıyorsa) ya da FFFFFFC000000000 (8 baytlık günlük RNA' ları kullanıyorsa) değerine ulaşırsa, kuyruk yöneticisi sonlandırılır. Neden kodu 00D10257.

Kullanılan günlük aralığına ilişkin uyarı iletileri yayınlandıktan sonra, kuyruk yöneticisinin 8 baytlık günlük RNA ' larını kullanacak şekilde dönüştürülebileceği ya da günlüğün ilk durumuna getirilebileceği bir kuyruk yöneticisi kesintisi planlamanız gerekir. Günlüğün ilk durumuna getirilmesine ilişkin yordam Kuyruk yöneticisinin günlüğünün ilk durumuna getirilmesibaşlıklı kısımda belgelenmiştir.

Kuyruk yöneticiniz 6 baytlık günlük RNA 'ları kullanıyorsa, Daha büyük günlük Göreli Bayt Adresinin Uygulanmasıbaşlıklı kısımda açıklanan yordamı izleyerek, kuyruk yöneticisini kuyruk yöneticisinin günlüğünü sıfırlamak yerine 8 baytlık günlük RNA' larını kullanacak şekilde dönüştürmeyi düşünün.

Günlük, her biri tek bir birim olarak işlem gören bir günlük verileri kümesi olan *günlük kayıtlarından*oluşur. Günlük kaydı, üstbilgisinin ilk baytındaki RBA ile ya da günlük kaydı sıra numarasıyla (LRSN) tanımlanır. RBA ya da LRSN, günlükte belirli bir noktadan başlayan bir kaydı benzersiz olarak tanımlar.

Günlük noktalarını tanımlamak için RBA ya da LRSN kullanıp kullanmayacağınız, kuyruk paylaşım gruplarını kullanıp kullanmadığınıza bağlıdır. Bir kuyruk paylaşım ortamında, birden çok kuyruk yöneticisi aynı kuyruğu aynı anda güncelleyebildiğinden ve her birinin kendi günlüğü olduğundan, bir günlük noktasını benzersiz olarak tanımlamak için göreli bayt adresini kullanamazsınız. Bunu çözmek için, günlük kaydı sıra numarası bir zaman damgası değerinden türetilir ve günlük içindeki günlük kaydının fiziksel yer değiştirmesini göstermez.

Her günlük kaydının tipi, kaydı yapan IBM MQ alt bileşeni ve kurtarma kaydı birimi için bir kurtarma tanıtıcısı birimi veren bir üstbilgisi vardır.

Aşağıdaki başlıklar altında açıklanan dört tip günlük kaydı vardır:

- Kurtarma günlüğü kaydı 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 kısaltmak ve sistem kullanılabilirliğini artırmak için *geri alma/yineleme* ve *günlük kayıtlarını dengeleme* gibi özel günlük kaydı tekniklerini kullanır.

Bunun bir etkisi, yeniden başlatma zamanının sınırlı olması. Yeniden başlatma sırasında bir hata oluşursa, kuyruk yöneticisinin ikinci kez yeniden başlatılması gerekirse, ilk yeniden başlatmada başarısızlık noktasına kadar tamamlanan tüm kurtarma etkinliğinin ikinci yeniden başlatma sırasında yeniden uygulanması gerekmez. Bu, ardışık yeniden başlatmanın tamamlanma sürelerini aşamalı olarak daha uzun sürmeyeceği anlamına gelir.

Denetim noktası kayıtları

Yeniden başlatma süresini kısaltmak için, IBM MQ olağan çalışma sırasında periyodik denetim noktalarını alır. Bunlar aşağıdaki gibi oluşur:

- Önceden tanımlanmış sayıda günlük kaydı yazıldığında. Bu sayı, CSQ6SYSPsistem değiştirgesi makrosunun LOGLOAD adlı denetim noktası sıklığı işleneniyle tanımlanır; açıklamalar için bkz. CSQ6SYSP.
- Başarılı bir yeniden başlatma işleminin sonunda.
- Normal sonlandırmada.
- IBM MQ , döngüde bir sonraki etkin günlük verilerine geçtiğinde.

Bir denetim noktası alındığında, IBM MQ DISPLAY CONN komutunu verir (DISPLAY CONN başlıklı konuda açıklanmıştır). z/OS konsol günlüğüne şu anda kuşkulanmakta olan bağlantıların bir listesinin yazılması için dahili olarak.

Sayfa kümesi denetim kayıtları

Bu kayıtlar, her denetim noktasında IBM MQ kuyruk yöneticisi tarafından bilinen sayfa kümelerini ve arabellek havuzlarını kaydeder ve denetim noktası sırasında sayfa kümesinin ortam kurtarmasını gerçekleştirmek için gereken günlük aralıklarına ilişkin bilgileri kaydeder.

Sayfa kümelerinde ve arabellek havuzlarında yapılan bazı dinamik değişiklikler de sayfa kümesi denetim kayıtları olarak yazılır; böylece değişiklikler kurtarılabilir ve sonraki kuyruk yöneticisi yeniden başlatıldığında otomatik olarak yeniden yürürlüğe konabilir.

CF yapısı yedekleme kayıtları

Bu kayıtlar, BACKUP CFSTRUCT komutuna yanıt olarak bir bağlaşım olanağı liste yapısından okunan verileri içerir. Bir bağlaşım olanağı yapısının arızalanması durumunda, bu kayıtlar, kurtarma kayıtları birimiyle birlikte RECOVER CFSTRUCT komutuyla birlikte, bağlantı olanağı yapısının hata noktasına ortam kurtarma işlemini gerçekleştirmek için kullanılır.

İlgili görevler

Daha büyük günlük Göreli Bayt Adresi Uygulanıyor

z/OS Günlüğün nasıl yapılandırıldığını

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 ' lik bir denetim aralığıdır (CI). Her yapılandırma öğesi bir VSAM kaydı içerir.

Fiziksel ve mantıksal günlük kayıtları

Bir VSAM Yapılandırma Öğesi, *fiziksel* bir kayıttır. Belirli bir zamanda kaydedilen bilgiler, yapılandırma öğesindeki kullanılabilir alandan bağımsız olarak değişen uzunluğa sahip bir *mantıksal* kayıt oluşturur. Bir fiziksel kayıt şunları içerebilir:

- Birkaç mantıksal kayıt
- Bir ya da daha çok mantıksal kayıt ve başka bir mantıksal kaydın parçası
- Yalnızca bir mantıksal kaydın parçası

Günlük kaydı terimi, *mantıksal* kaydını depolamak için kaç *fiziksel* kayıt gerektiğinden bağımsız olarak gösterir.

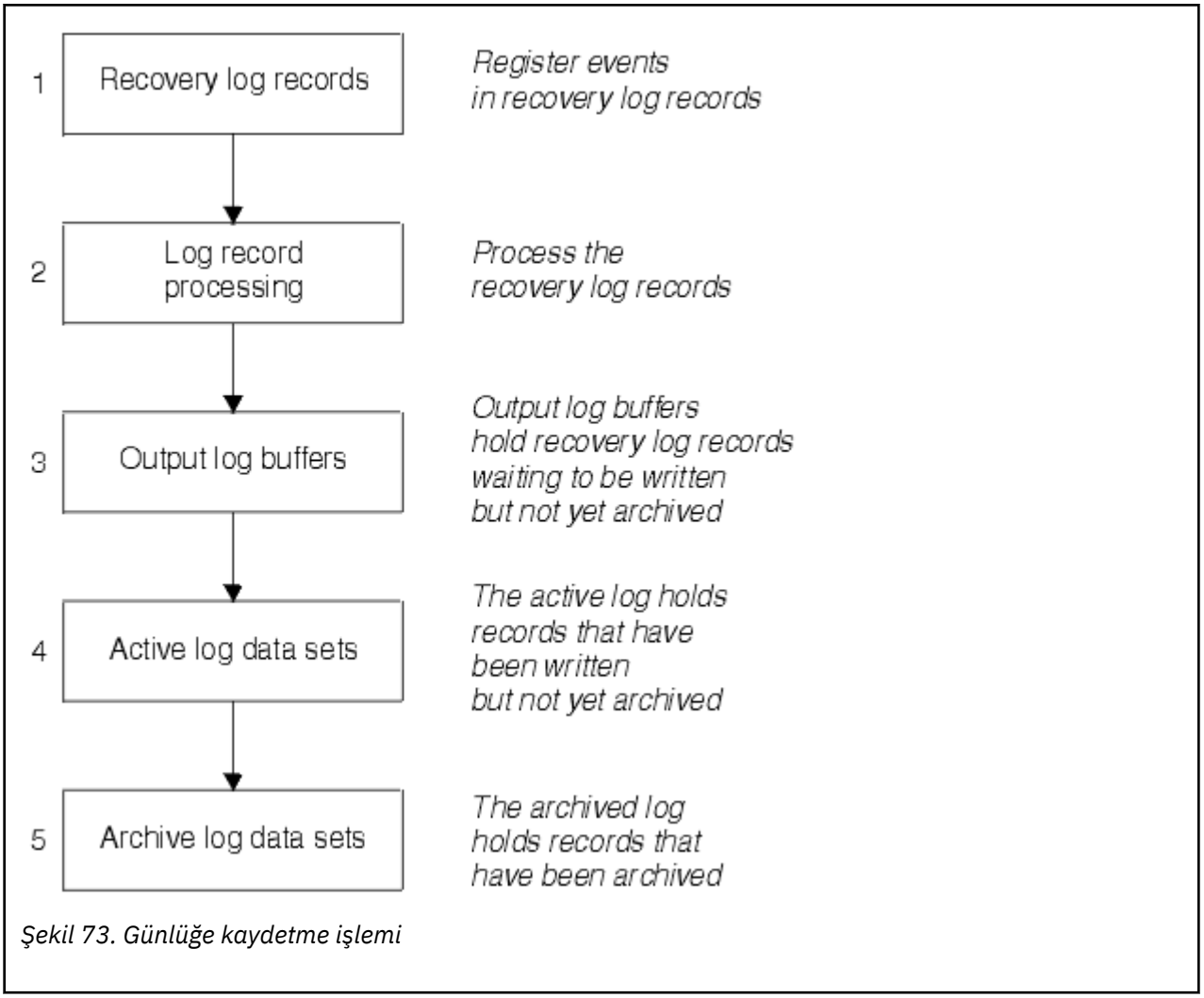
z/OS IBM MQ for z/OS günlüklerinin yazılması

IBM MQ ' ün günlük dosyası kayıtlarını nasıl işlediğini anlamak için bu konuyu kullanın.

IBM MQ , her günlük kaydını *etkin günlük*adlı bir DASD veri kümesine yazar. Etkin günlük dolduğunda IBM MQ , içeriğini *arşiv günlüğü*adlı bir DASD ya da manyetik bant veri kümesine kopyalar. Bu işleme *boşaltma*adı verilir.

Şekil 73 sayfa 227 içinde günlüğe kaydetme işlemi gösterilmektedir. Günlük kayıtları genellikle aşağıdaki döngüden geçer:

1. IBM MQ , kurtarma günlüğü kayıtlarındaki verilerde ve önemli olaylarda yapılan değişiklikleri not eder.
2. IBM MQ , kurtarma günlüğü kayıtlarını işler ve gerekirse bunları kesimlere ayırır.
3. Günlük kayıtları, VSAM Denetim Aralıkları (CI) olarak biçimlendirilen *çıkış günlüğü arabelleklerine*sırayla yerleştirilir. Her günlük kaydı, sıfır-2 ^{64} -1 aralığındaki göreli bir bayt adresiyle tanımlanır.
4. Yapılandırma Öğeleri, sıralı olarak kullanılan ve geri dönüştürülen bir dizi önceden tanımlanmış DASD etkin günlük veri kümesine yazılır.
5. Arşivleme etkinse, her etkin günlük veri kümesi dolduğunda, içeriği otomatik olarak yeni bir arşiv günlüğü veri kümesine boşaltılır.



Şekil 73. Günlüğe kaydetme işlemi

Etkin günlük yazıldığında

Depolama içindeki günlük arabellekleri, aşağıdakilerden herhangi biri oluştuğunda etkin bir günlük veri kümesine yazılır:

- Günlük arabellekleri doluyor.
- Yazma eşiğine ulaşılır (CSQ6LOGP makrosunda belirtildiği gibi).
- Kesinleştirme noktası gibi belirli önemli olaylar ya da IBM MQ BACKUP CFSTRUCT komutu verildiğinde ortaya çıkar.

Kuyruk yöneticisi kullanıma hazırlandığında, BSDS 'de adı belirtilen etkin günlük veri kümeleri, kuyruk yöneticisi tarafından dışlayıcı kullanım için dinamik olarak ayrılır ve kuyruk yöneticisi sona erinceye kadar özel olarak IBM MQ ' e ayrılır.

Günlük veri kümelerini dinamik olarak ekleme

Kuyruk yöneticisi çalışırken yeni etkin günlük veri kümelerini dinamik olarak tanımlamak mümkündür. Bu özellik, arşivleme geçici bir sorun nedeniyle etkin günlükleri boşaltamadığında 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 ayırırken Storage Management Subsystem (MVS/DFP SMS) depolama sınıflarını belirtmenizi sağlar. IBM MQ , günlük veri kümelerinin arşivlenmesini başlatır, ancak arşiv veri kümesinin ayrılmasını gerçekleştirmek için SMS ' yi kullanabilirsiniz.

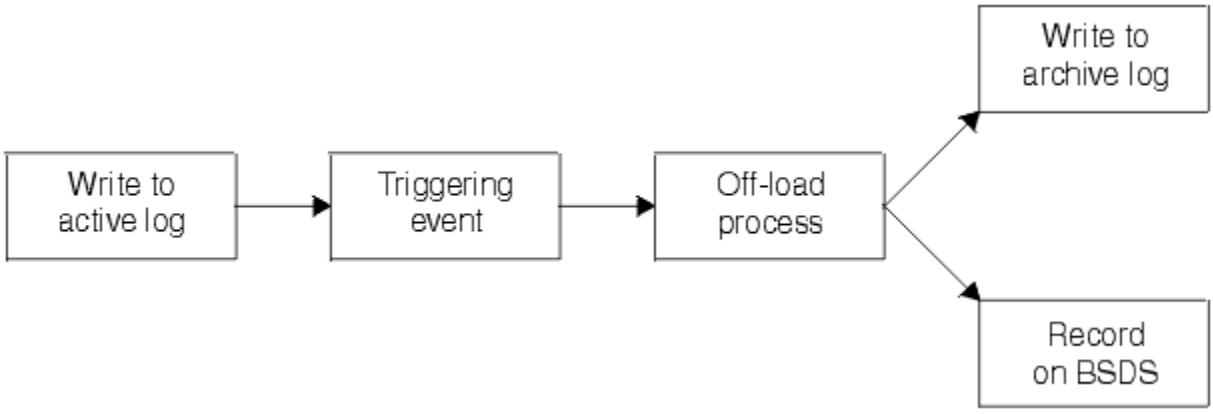
İlgili başvurular

"IBM MQ for z/OS arşiv günlüğü yazıldığında" sayfa 228
Etkin günlükleri arşiv günlüklerine kopyalama işlemini ve işlemin ne zaman oluştuğunu anlamak için bu konuyu kullanın.

z/OS *IBM MQ for z/OS arşiv günlüğü yazıldığında*

Etkin günlükleri arşiv günlüklerine kopyalama işlemini ve işlemin ne zaman oluştuğunu anlamak için bu konuyu kullanın.

Etkin günlükleri arşiv günlüklerine kopyalama işlemine *boşaltma*adı verilir. Diğer günlük kaydı olaylarıyla boşaltma ilişkisi, Şekil 74 sayfa 228içinde şematik olarak gösterilir.



Şekil 74. Boşaltma işlemi

Boşaltma işlemini tetikleme

Etkin bir günlüğün arşiv günlüğüne boşaltma işlemi birkaç olay tarafından tetiklenebilir. Örneğin:

- Etkin günlük veri kümesi dolduruluyor.
- MQSC ARCHIVE LOG komutu kullanılıyor.
- Etkin günlük veri kümesine yazılırken oluşan bir hata.

Veri kümesi, hata noktasından önce kesilir ve yazılmayan kayıt, yeni veri kümesinin ilk kaydı olur. Normal tam günlük veri kümesi olarak kesilen veri kümesi için boşaltma tetiklenir. İkili etkin günlükler varsa, iki kopya eşitlenmiş olarak kalacak şekilde her iki kopya da kesilir.

Kullanılabilir son etkin günlük %5 dolduğunda CSQJ110E iletisi yayınlanır ve bu ileti, kullanılmakta olan günlük kapasitesinin yüzdesini belirtir. Tüm etkin günlükler dolursa, IBM MQ boşaltma gerçekleşinceye kadar işlemeyi durdurur ve şu iletiyi verir:

```
CSQJ111A +CSQ1 OUT OF SPACE IN ACTIVE LOG DATA SETS
```

Boşaltma işlemi

Tüm etkin günlükler dolduğunda, IBM MQ boşaltma işlemini çalıştırır ve boşaltma işlemi tamamlanıncaya kadar işlemeyi durdurur. Etkin günlükler dolduğunda boşaltma işlemi başarısız olursa, IBM MQ olağandışı sonlanır.

Etkin bir günlük boşaltılmaya hazır olduğunda, z/OS konsol işletmenine bir manyetik bant takması ya da bir DASD birimi hazırlaması için bir istek gönderilir. ARCWTOR günlük kaydı seçeneğinin değeri (ek bilgi için bkz. CSQ6ARVP) isteğin alınıp alınmadığını belirler. Boşa almak için manyetik bant kullanıyorsanız, ARCWTOR=YES değerini belirtin. Değer YES ise, istekten önce bir WTOR gelir (ileti numarası CSQJ008E) işlecin ayrılacak arşiv günlüğü veri kümesini hazırlamasını sağlar.

İşletmenin bu iletiye hemen yanıt vermesi gerekmez. Ancak, yanıtın geciktirmesi boşaltma işlemini geciktirmektedir. İşletmen, IBM MQ ' in etkin günlükleri tükenecek kadar uzun süre yanıt geciktirmediği sürece IBM MQ performansını etkilemez.

İşletmen, boşaltma işlemini iptal ederek yanıt verebilir. Bu durumda, ayırma, ikili arşiv veri kümelerinin ilk kopyasına ilişkin ise, boşaltma işlemi yalnızca sonraki etkin günlük veri kümesi doluncaya kadar geciktirilir. Ayırma ikinci kopya içinse, arşiv işlemi yalnızca bu veri kümesi için tek kopyalama kipine geçer.

Boşaltılırken kesintiler ve hatalar oluştu

Boşaltma işlemi tamamlanıncaya kadar kuyruk yöneticisini durdurma isteği yürürlüğe girmez. Boşaltma işlemi devam ederken IBM MQ başarısız olursa, boşaltma işlemi kuyruk yöneticisi yeniden başlatıldığında yeniden başlar.

Boşaltma işlemi sırasında iletiler

Boşaltılan iletiler, z/OS konsoluna IBM MQ tarafından ve boşaltma işlemi tarafından gönderilir. Çeşitli günlük veri kümelerinde RBA aralıklarını bulmak için bu iletileri kullanabilirsiniz.

z/OS Daha büyük günlük göreli byte adresi

Bu işlev, günlüğü ilk durumuna getirmeden önce geçen süreyi artırarak kuyruk yöneticisinin kullanılabilirliğini artırır.

Kurtarma verileri, kuyruk yöneticisi yeniden başlatıldığında kalıcı iletilerin kullanılabilmesi için günlüğe yazılır. Günlük Günlüğü Göreli Bayt Adresi (günlük RBA) terimi, verilerin konumunu günlüğün başından itibaren bir göreli konum olarak belirtmek için kullanılır.

IBM MQ 8.0' den önce, 6 baytlık günlük RBA, 256 terabayta kadar verileri adresleyebiliyordu. Bu sayıda günlük kaydı yazılmadan önce, Kuyruk yöneticisinin günlüğünü ilk durumuna getirmebaşlıklı konuda belirtilen yordamı izleyerek kuyruk yöneticisinin günlüğünü ilk durumuna getirmeniz gerekir.

Kuyruk yöneticilerinin günlüklerinin sıfırlanması hızlı bir işlem değildir ve işlemin bir parçası olarak sayfa kümelerini sıfırlama gereksinimi nedeniyle genişletilmiş bir kesinti gerektirebilir. Yüksek kullanımlı bir kuyruk yöneticisi için bu işlem genellikle yılda bir kez yapılabilir.

IBM MQ 8.0'den itibaren günlük RBA 8 bayt uzunluğunda olabilir ve kuyruk yöneticisi artık günlük RBA' nın sıfırlanması gerekmeden önce 64,000 'den fazla veriyi (16 exabyte) adresleyebilir. Daha büyük günlük RBA ' nın kullanılmasının etkisi, yazılan günlük verilerinin boyutunun birkaç bayt artmasına yol açmaktadır.

Bu işlev ne zaman etkinleştirilir?

V 9.3.0 IBM MQ 9.3.0 ya da daha sonra yaratılan kuyruk yöneticilerinin bu işlevi zaten etkinleştirilmiş.

Geçerli günlük RBA günlük RBA aralığı sonuna yaklaşıyorsa, kuyruk yöneticisini kuyruk yöneticisinin günlüğünü sıfırlamak yerine 8 baytlık bir günlük RBA kullanacak şekilde dönüştürmeyi düşünün. Bir kuyruk yöneticisinin 8 baytlık günlük RNA ' larını kullanacak şekilde dönüştürülmesi, günlüğün ilk durumuna getirilmesinden daha kısa bir kesinti gerektirir ve günlüğü ilk durumuna getirmeden önce geçen süreyi önemli ölçüde artırır.

Kuyruk yöneticisi kullanıma hazırlanırken yayınlanan CSQJ034Iiletisi, kuyruk yöneticisi için yapılandırılan günlük RBA aralığının sonunu gösterir ve 6 baytlık ya da 8 baytlık günlük RNA ' larının kullanılıp kullanılmadığını belirlemek için kullanılabilir.

Bu işlev nasıl etkinleştirilir?

8 baytlık günlük RBA, kuyruk yöneticisi sürüm 2 biçimi BSDS ile başlatılarak etkinleştirilir. Özetle, bu işlem aşağıdaki şekilde gerçekleştirilir:

1. Kuyruk paylaşım grubundaki tüm kuyruk yöneticilerinin 8 baytlık günlük RBA ' nın etkinleştirilmesine ilişkin gereksinimleri karşıladığından emin olunması
2. Kuyruk yöneticisinin düzgün bir şekilde kapatılması
3. BSDS ' nin bir kopyasını sürüm 2 biçiminde yaratmak için BSDS dönüştürme yardımcı programı çalıştırılıyor.
4. Dönüştürülen BSDS ile kuyruk yöneticisi yeniden başlatılıyor.

Bir kuyruk yöneticisi 8 baytlık günlük RNA ' larını kullanacak şekilde dönüştürüldükten sonra, 6 baytlık günlük RBA kullanımına geri dönemez.

8 baytlık günlük RNA ' larını geçerli kılmaya ilişkin ayrıntılı yordam için Daha büyük günlük Göreli Bayt Adresinin Uygulanması başlıklı konuya bakın.

İlgili görevler

Adreslenebilir günlük aralığı üst sınırını artırmayı planlama

İlgili başvurular

BSDS dönüştürme yardımcı programı (CSQJUCNV)

z/OS Önyükleme veri kümesi

IBM MQ , günlük veri kümelerine ve günlük kayıtlarına başvurmak için bir mekanizma olarak önyükleme veri kümesini gerektirir. Bu bilgiler olağan işleme sırasında gereklidir ve kurtarmayı yeniden başlatın.

Önyükleme veri kümesinin ne için olduğunu

Bootstrap veri kümesi (BSDS), IBM MQiçin gerekli bilgileri tutan bir VSAM anahtar sıralı veri kümesidir (KSDS). Aşağıda belirtilenleri içerir:

- IBM MQtarafından bilinen tüm etkin ve arşivlenmiş günlük veri kümelerinin dökümü. IBM MQ aşağıdaki işlemleri yapmak için bu dökümü kullanır:
  - Etkin ve arşivlenmiş günlük veri kümelerini izle
  - Normal işleme sırasında günlük okuma isteklerini karşılayabilmesi için günlük kayıtlarını bulun
  - Yeniden başlatma işlemini işleyebilmesi için günlük kayıtlarını bul

  IBM MQ , her arşiv günlüğü veri kümesi tanımlandığında ya da etkin bir günlük veri kümesi yeniden kullanıldığında bilgileri dökümde saklar. Etkin günlükler için, stok hangilerinin dolu olduğunu ve hangilerinin yeniden kullanılabilir olduğunu gösterir. Döküm, o veri kümesinde tutulan günlüğün her bir bölümünün göreli bayt adresini (RBA) tutar.
- Tüm son IBM MQ etkinliğinin *sarma* dökümü. Kuyruk yöneticisini yeniden başlatmanız gerekiyorsa bu gereklidir.

Kuyruk yöneticisinde hata varsa ve yeniden başlatmanız gerekiyorsa, BSDS gereklidir. IBM MQ **bir BSDS ' ye sahip olmalıdır** . Yeniden başlatma sırasında sorun olasılığını en aza indirmek için, IBM MQ ' i her biri aynı bilgileri kaydeden ikili BSDS'lerle yapılandırabilirsiniz. İkili BSDSlerin kullanılması, *ikili kipte*çalışma olarak bilinir. Mümkünse, kopyaları ayrı birimlere yerleştirin. Bu, birim bozulursa ya da yok edilirse her ikisinin de kaybolma riskini azaltır. DASD ' ye ikili yazma yerine ikili BSD kullanın.

BSDS, IBM MQ özelleştirildiğinde ayarlanır ve değişiklik günlüğü stok yardımcı programını (CSQJU003) kullanarak dökümü yönetebilirsiniz. Bu yardımcı programla ilgili daha fazla bilgi için bkz. IBM MQ for z/OS Yönetimi. Kuyruk yöneticisi başlatma yordamındaki bir DD deyimi bu öğeye gönderme yapıyor.

Olağan durumda IBM MQ , BSDS ' nin yinelenen kopyalarını tutar. Bir G/Ç hatası oluşursa, hatalı kopyanın yerini ayırır ve tek bir BSDS ile devam eder. Çift kipli işlemi geri yükleyebilirsiniz; bu, IBM MQ for z/OS Yönetimiiçinde açıklanır.

Etkin günlükler ilk olarak IBM MQ kurulduğunda BSDS ' ye kaydedilir. Kuyruk yöneticisini sonlandırmadan ve yeniden başlatmadan etkin günlükleri değiştiremezsiniz.

Arşiv günlüğü veri kümeleri dinamik olarak ayrılır. Bir veri kümesi ayrıldığında, veri kümesi adı BSDS ' de kaydedilir. Arşiv günlüğü veri kümeleri listesi, arşivler eklendikçe genişler ve kullanıcı tarafından belirlenen sayıda girdiye ulaşıldığında kayar. Tek arşiv günlük kaydı için giriş sayısı üst sınırı 1000, ikili günlük kaydı için 2000 'dir.

Arşiv günlüğü veri kümelerini silmek için bir manyetik bant yönetim sistemi kullanabilirsiniz (IBM MQ otomatik bir yönteme sahip değildir). Bu nedenle, arşiv günlüğü veri kümesiyle ilgili bilgiler, arşiv günlüğü veri kümesi sistem yöneticisi tarafından silindikten çok sonra BSDS ' de olabilir.

Tersine, arşiv günlüğü veri kümesi sayısı üst sınırı aşılmış olabilir ve BSDS ' deki veriler, veri kümesi süre bitim tarihine ulaşmadan çok önce düşmüş olabilir.

Günlüğün kapsamını ve çeşitli ortam ya da kuyruk yöneticisi kurtarma tipleri için gerekli olan en eski günlük RBA kaydını tutan etkin ya da arşiv günlüğü veri kümesinin adını saptamak için aşağıdaki MQSC komutunu kullanabilirsiniz:

```
DISPLAY USAGE TYPE(DATASET)
```

Sistem parametre modülü, arşiv günlüğü veri kümelerinin ayrıldığında kataloğa alındığını belirtiyorsa, BSDS, sonraki ayırmalar için gerekli bilgiler için tümleşik katalog olanağı (ICF) kataloğunu işaret eder. Tersi durumda, her birim için BSDS girişleri, daha sonra ayırmalar için gerekli birim seri numarasını ve birim bilgilerini kaydeder.

BSDS sürümü

BSDS ' nin biçimi kendi sürümüne göre değişir. BSDS sürümünün artırılması, yeni özelliklerin kullanılmasına izin verir. Aşağıdaki BSDS sürümleri IBM MQtarafından desteklenir:

Sürüm 1
IBM MQ' in tüm yayınları tarafından desteklenir. Sürüm 1 BSDS, 6 baytlık günlük RBA değerlerini destekler.

Sürüm 2
IBM MQ 8.0 ve sonrası tarafından desteklenir. Sürüm 2 BSDS, 8 baytlık günlük RBA değerlerini ve her etkin günlük kopyasında en çok 310 veri kümesini etkinleştirir.

V 9.3.0 IBM MQ 9.3.0 ve daha sonra yaratılan kuyruk yöneticileri için varsayılan olarak etkinleştirilir.

Sürüm 3
IBM MQ 8.0 ve sonrası tarafından desteklenir. Etkin günlük kopyalarından birine 31 'den fazla veri kümesi eklendiğinde, BSDS otomatik olarak sürüm 3 'e dönüştürülür.

BSDS sürümünü, yazdırma günlüğü eşlemi yardımcı programını (CSQJU004)çalıştırarak saptayabilirsiniz. BSDS ' yi Sürüm 1 'den Sürüm 2 'ye dönüştürmek için BSDS dönüştürme yardımcı programını (CSQJUCNV)çalıştırın.

6 baytlık ve 8 baytlık günlük RNA ' larına ilişkin ek bilgi için bkz. “Daha büyük günlük göreli byte adresi” sayfa 229 .

Günlük veri kümelerini ve BSDS kopyalarını arşivle

Yeni bir arşiv günlüğü veri kümesi her yaratıldığında, BSDS ' nin bir kopyası da yaratılır. Arşiv günlüğü manyetik banttaysa, BSDS ilk çıkış birimindeki ilk veri kümesidir. Arşiv günlüğü DASD üzerdeyse, BSDS ayrı bir veri kümesidir.

Arşiv günlüğünün ve BSDS kopyasının veri kümesi adları 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

BSDS kopyalanırken okuma hatası oluşursa, kopya oluşturulmaz, CSQJ125E iletisi yayınlanır ve yeni arşiv günlüğü veri kümesine yapılan boşaltma BSDS kopyası olmadan devam eder.

z/OS z/OS üzerinde sistem tanımı

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

IBM MQ for z/OSiçinde bir sistem parametresi modülü, IBM MQ ' un işletiminde kullandığı günlüğe kaydetme, arşivleme, izleme ve bağlantı ortamlarını denetler. Sistem parametreleri, aşağıdaki gibi üç çevirici makroları tarafından belirlenir:

CSQ6SYSP
Bağlantı ve izleme ortamının ayarlanması da içinde olmak üzere sistem parametreleri.

CSQ6LOGP
Günlüğe kaydetme parametreleri.

CSQ6ARVP
Günlük arşivi parametreleri.

IBM MQ for z/OSile birlikte varsayılan parametre modülleri sağlanır. Bunlar kullanmak istediğiniz değerleri içermiyorsa, IBM MQile verilen örneği kullanarak kendi parametre modüllerinizi yaratabilirsiniz. Örnek: `thlqual.SCSQPROC(CSQ4ZPRM)`.

Bir kuyruk yöneticisi çalışırken bazı sistem parametrelerini değiştirebilirsiniz. MQSC komutlarıiçindeki SET SYSTEM, SET LOG ve SET ARCHIVE komutlarına bakın.

Tanımlama hakkında daha fazla bilgi için aşağıdaki konulara bakın:

- “IBM MQ for z/OS için sistem nesnelerinin tanımlanması” sayfa 232
- “IBM MQ for z/OS üzerinde kuyruk yöneticinizi ayarlama” sayfa 237
- “IBM MQ for z/OS ile verilen örnek tanımlar” sayfa 237

İlgili kavramlar
Örnek kullanıma hazırlama giriş veri kümelerinin uyarlanması
IBM MQ for z/OS üzerinde MQSC ve PCF komutlarını yayınlayabileceğiniz kaynaklar
İlgili görevler
Yönetmez/OS
Kümeleri yapılandırma
İzlemeIBM MQ

z/OS IBM MQ for z/OS için sistem nesnelerinin tanımlanması

IBM MQ for z/OS , yayınlama/abone olma uygulamaları, küme ve kanal denetimi ve diğer sistem yönetimi işlevleri için önceden tanımlanmış ek nesneler gerektirir.

IBM MQ for z/OS ' in gerektirdiği sistem nesneleri aşağıdaki kategorilere ayrılabilir:

- Nesneleri yayınla/abone ol
- Sistem varsayılan nesneleri

- Sistem komut nesneleri
- Sistem denetimi nesneleri
- Kanal kuyrukları
- Küme kuyrukları
- Kuyruk paylaşım grubu kuyrukları
- Depolama sınıfları
- Sistem nesnesinin ölmeyen iletiler kuyruğunun tanımlanması
- Varsayılan iletim kuyruğu
- İç kuyruklar
- “Kanal kimlik doğrulama kuyruğu” sayfa 236

Nesneleri yayınla/abone ol

IBM MQ for z/OSile yayınlama/abone olma uygulamalarını kullanmadan önce tanımlamanız gereken birkaç sistem nesnesi vardır. Örnek tanımlamalar, bu nesneleri tanımlamanıza yardımcı olmak için IBM MQ ile birlikte sağlanır. Bu örnekler CSQ4INSGiçinde açıklanmıştır.

Yayınlama/abone olma özelliğini kullanmak için aşağıdaki nesneleri tanımlamanız gerekir:

- SYSTEM.RETAINED.PUB.QUEUE. Her tam konu adının, bu kuyrukta saklanan en çok bir yayını olabilir. Uygulamalarınız birçok farklı konuda alıkonan yayınları kullanacaksa ya da alıkonan yayın iletileriniz büyük iletilerse, bu kuyruk için depolama gereksinimleri, depolama gereksinimleri büyükse, kendi sayfa kümesine atanması da dahil olmak üzere, dikkatli bir şekilde planlanmalıdır. Başarımı artırmak için, bu kuyruğu MSGID dizin tipiyle (sağlanan örnek kuyruk tanımında gösterildiği gibi) tanımlamanız gerekir.
- SYSTEM.DURABLE.SUBSCRIBER.QUEUE. Başarımı artırmak için, bu kuyruğu CORRELID dizin tipiyle (sağlanan örnek kuyruk tanımlamasında gösterildiği gibi) tanımlamanız gerekir.
- SYSTEM.DURABLE.MODEL.QUEUE.
- SYSTEM.NDURABLE.MODEL.QUEUE.
- SYSTEM.QPUBSUB.QUEUE.NAMELIST.
- SYSTEM.QPUBSUB.SUBPOINT.NAMELIST.
- SYSTEM.BASE.TOPIC.
- SYSTEM.BROKER.DEFAULT.STREAM.
- Kuyruğa alınan yayınlama/abone olma arabirimi tarafından kullanılan varsayılan RFH2 abonelik noktası olan SYSTEM.BROKER.DEFAULT.SUBPOINTadlı bir konu.
- SYSTEM.BROKER.ADMIN.STREAM, kuyruğa alınan yayınlama/abone olma arabirimi tarafından kullanılan yönetici akışıdır.
- SYSTEM.DEFAULT.SUB.

Sistem varsayılan nesneleri

Sistem varsayılan nesneleri, bir nesne tanımladığınızda varsayılan öznitelikleri sağlamak için kullanılır ve tanımlamanın dayandırılacağı başka bir nesnenin adını belirlemeyin.

Varsayılan sistem nesnesi tanımlarının 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 ad kuyrukları

- Uzak kuyruklar
- Süreçler
- Namelistler
- Kanallar
- Depolama sınıfları
- Kimlik doğrulama bilgileri

Paylaşılan kuyruklar özel bir yerel kuyruk tipidir; bu nedenle, paylaşılan bir kuyruk tanımladığınızda, tanımlama SYSTEM.DEFAULT.LOCAL.QUEUE. Varsayılan tanımda bir değer belirtilmediğinden, Bağlama Olanağı yapı adı için bir değer belirtmeyi unutmayın. Diğer bir seçenek olarak, paylaşılan kuyruklar için temel olarak kullanmak üzere kendi varsayılan paylaşılan kuyruk tanımlamanızı tanımlayabilirsiniz; böylece, bunların tümü gerekli öznitelikleri devralır. Yalnızca kuyruk paylaşım grubundaki bir kuyruk yöneticisinde paylaşılan kuyruk tanımlamanız gerektiğini unutmayın.

Sistem komut nesneleri

Sistem komut nesnelerinin adları SYSTEM.COMMANDKOMUTUNU YAZıN. Bir IBM MQ altsistemine komut vermek için IBM MQ işlemlerini ve denetim panolarını kullanmadan önce bu nesneleri tanımlamanız gerekir.

İki sistem komut nesnesi vardır:

1. Sistem komutu giriş kuyruğu, komutların IBM MQ komut işlemcisi tarafından işlenmeden önce yerleştirildiği yerel bir kuyruktur. SYSTEM.COMMAND.INPUTGIRIşI. SYSTEM.ADMIN.COMMAND.QUEUE , IBM MQ for Multiplatformsile uyumluluk için ve IBM MQ Console ve administrative REST APItarafından kullanım için de tanımlanmalıdır.
2. SYSTEM.COMMAND.REPLY.MODEL , sistem komutu yanıt kuyruğunu tanımlayan bir model kuyruğudur.

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 kuyruğudur.

Komutlar genellikle kalıcı olmayan iletiler kullanılarak gönderilir; böylece, her iki sistem komut nesnesinin de DEFPSIST (NO) özniteliği olmalıdır; böylece, bunları kullanan uygulamalar (yardımcı program ve işlemler ve denetim panoları gibi sağlanan uygulamalar da içinde olmak üzere) varsayılan olarak kalıcı olmayan iletileri alır. Komutlar için kalıcı iletiler kullanan bir uygulamanız varsa, yanıt iletileri kalıcı olduğundan, yanıt kuyruğu için DEFTYPE (PERMDYN) özniteliğini ayarlayın.

Sistem yönetimi nesneleri

Sistem denetimi nesnelerinin adları SYSTEM.ADMIN.

Yedi sistem yönetimi 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

Dağıtımlı kuyruğa alma özelliğini kullanmak için aşağıdaki nesneleri tanımlamanız gerekir:

- SYSTEM.CHANNEL.SYNCQ. Kanal başarımını artırmak için, bu kuyruğu MSGID dizin tipiyle (sağlanan örnek kuyruk tanımında gösterildiği gibi) tanımlamanız gerekir.
- SYSTEM.CHANNEL.INITQ.

Bu kuyrukları paylaşılan kuyruk olarak tanımlayamazsınız.

Küme kuyrukları

IBM MQ kümelerini kullanmak için aşağıdaki nesneleri tanımlamanız gerekir:

- SYSTEM.CLUSTER.COMMAND.QUEUE. Bu kuyruğa yazılan iletiler, havuzun yerel kopyasına uygulanacak havuz verilerine ilişkin güncellemeleri ya da havuz verilerine ilişkin istekleri içerir.
- SYSTEM.CLUSTER.REPOSITORY.QUEUE.
- SYSTEM.CLUSTER.TRANSMIT.QUEUE. Başarım nedenleriyle, bu kuyruğu CORRELID dizin tipiyle (örnek kuyruk tanımlamasında gösterildiği gibi) tanımlamanız gerekir.

Bu kuyruklar genellikle çok sayıda ileti içerir.

Bu kuyrukları paylaşılan kuyruk olarak tanımlayamazsınız.

Kuyruk paylaşım grubu kuyrukları

Paylaşılan kanalları ve grup içi kuyruğa alma özelliğini kullanmak için aşağıdaki nesneleri tanımlamanız gerekir:

- SYSTEM.QSG.CHANNEL.SYNCQ.
- SYSTEM.QSG.TRANSMIT.QUEUE. Bir kuyruk paylaşım grubunda çalışıyorsanız, grup içi kuyruğa alma kullanmıyor olsanız da bu kuyruğu tanımlamanız gerekir.

Depolama sınıfları

Aşağıdaki altı depolama sınıfını tanımlamanız önerilir. Bunlardan dördünü tanımlamanız gerekir, çünkü bunlar IBM MQtarafından gereklidir. Diğer depolama sınıfı tanımlamaları, örnek kuyruk tanımlarında kullanıldıkları için önerilir.

DEFAULT (gerekli)
Bu depolama sınıfı, performans açısından kritik olmayan ve diğer depolama sınıflarına uymayan tüm ileti kuyrukları için kullanılır. Kuyruk tanımlarken bir depolama sınıfı belirtmezseniz, bu sınıf, sağlanan varsayılan depolama sınıfıdır.

NODEFINE (gerekli)
Bir kuyruk tanımladığınızda belirtilen depolama sınıfı tanımlanmamışsa, bu depolama sınıfı kullanılır.

REMOTE (gerekli)
Bu depolama sınıfı, öncelikle kısa ömürlü performans açısından kritik iletileri olan, sistemle ilgili kuyruklar için kullanılır.

SYSLNGLV
Bu depolama sınıfı, uzun ömürlü, performans açısından kritik iletiler için kullanılır.

SYSTEM (gerekli)
Bu depolama sınıfı, performans açısından kritik, sistemle ilgili ileti kuyrukları (örneğin, SYSTEM.CHANNEL.SYNQ ve SYSTEM.CLUSTER.* kuyruklar.

SYSVOLAT
Bu depolama sınıfı, kısa ömürlü, performans açısından kritik iletiler için kullanılır.

Özniteliklerini değiştirebilir ve diğer depolama sınıfı tanımlamalarını gerektiği şekilde ekleyebilirsiniz.

Sistem nesnesi ileti kuyruğunun tanımlanması

İleti hedefi geçerli değilse, gitmeyen iletiler kuyruğu kullanılır. IBM MQ , bu tür iletileri, gönderilmeyen iletiler kuyruğu adı verilen yerel bir kuyruğa koyar. Bir gitmeyen iletiler kuyruğuna sahip olmak zorunlu olmasa da, özellikle dağıtılmış kuyruğa alma ya da IBM MQ köprülerinden birini kullanıyorsanız, bunu temel olarak kabul etmelisiniz.

Gitmeyen iletiler kuyruğunu paylaşılan kuyruk olarak **tanımlamayın** . Bir kuyruk yöneticisinde yerel kuyruğa konan bir koyma, gitmeyen iletiler kuyruğuna konabilir. Gitmeyen iletiler kuyruğu paylaşılan bir kuyruksa, farklı bir sistemdeki gitmeyen iletiler kuyruğu işleyicisi iletiyi işleyip aynı adı taşıyan bir kuyruğa yerleştirebilirse, ancak bu kuyruk farklı bir kuyruk yöneticisinde olduğu için yanlış kuyruk ya da farklı bir güvenlik profili olabilir. Kuyruk yoksa, yeniden işlenemez.

Bir gelmeyen iletiler kuyruğu tanımlamaya karar verirseniz, kuyruk yöneticisine adını da söylemeniz gerekir. Bunu yapmak için ALTER QMGR DEADQ (*kuyruk-adı*) komutunu kullanın. Daha fazla bilgi için bkz. Kuyruk yöneticisi özniteliklerini görüntüleme ve değiştirme.

Varsayılan iletim kuyruğu

Başka bir kuyruk yöneticisine ileti göndermek için uygun iletim kuyruğu yoksa, varsayılan iletim kuyruğu kullanılır. Varsayılan bir iletim kuyruğu tanımlarsanız, kuyruğa hizmet verecek bir kanal da tanımlamanız gerekir. Bunu yapmazsanız, varsayılan iletim kuyruğuna konan iletiler uzak kuyruk yöneticisine iletilmez ve kuyrukta 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 tanımlanan bir kuyruk, SYSTEM.PENDING.DATA.QUEUE, JMS yayınlama/abone olma ortamında sürekli aboneliklerin kullanımını destekler.
- **JMS 2.0 teslim gecikmesi hazırlama kuyruğu**
  - JMS 2.0 tarafından sağlanan teslim gecikmesi işlevselliği kullanılıyorsa, bir iç hazırlama kuyruğu olan SYSTEM.DDELAY.LOCAL.QUEUEtanımlanmalıdır. Bu kuyruk, teslim gecikmesi tamamlanıncaya ve ileti hedef hedefine yerleştirilinceye kadar, sıfır olmayan bir teslim gecikmesiyle gönderilen iletileri geçici olarak saklamak için kuyruk yöneticisi tarafından kullanılır. CSQ4INSG' de bu kuyruk için bir örnek tanımlama sağlanmıştır (açıklama satırı yapılmış).
  - SYSTEM.DDELAY.LOCAL.QUEUE kuyruğu, bir teslim gecikmesiyle gönderilecek beklenen ileti sayısı için STGCLASS, MAXMSGL ve MAXDEPTH özniteliklerini ayarlamanız gerekir. Buna ek olarak, SYSTEM.DDELAY.LOCAL.QUEUE kuyruğu, yalnızca kuyruk yöneticisinin iletileri bu kuyruğa koyabildiğinden emin olun. Hiçbir kullanıcı kimliğinin bu kuyruğa ileti koyma yetkisine sahip olmadığından emin olmak için dikkatli olunmalıdır.

Kanal kimlik doğrulama kuyruğu

Kanal kimlik doğrulamasının iç kullanımı için SYSTEM.CHLAUTH.DATA.QUEUE kuyruğu gerekli. Örnek tanımlamalar, bu nesneleri tanımlamanıza yardımcı olmak için IBM MQ ile birlikte sağlanır. Bu örnek, bazı varsayılan kuralları da tanımlayan CSQ4INSAiçinde açıklanmıştır.

z/OS IBM MQ for z/OS üzerinde kuyruk yöneticinizi ayarlama

Kuyruk yöneticinizin temel performans sorunlarını önlemek üzere ayarlandığından emin olmak için yapabileceğiniz birkaç basit adım vardır.

ALTER QMGR komutuyla belirlenen kuyruk yöneticisi öznitelikleriyle denetlenen kuyruk yöneticinizin başarımını artırmanın çeşitli yolları vardır. Bu kısım, kuyruk yöneticisinde izin verilen ileti sayısı üst sınırını ayarlayarak ya da kuyruk yöneticisinde 'temizlik' gerçekleştirerek bunu nasıl yapabileceğinize ilişkin bilgi içerir. IBM MQ SupportPac MP16 - z/OS Kapasite planlama ve ayarlama için IBM MQ , performans ve ayarlama hakkında daha fazla bilgi verir.

Eşitleme Noktaları

Kuyruk yöneticisinin rollerinden biri, bir uygulama içindeki syncpoint denetimidir. Bir uygulama, MQCMIT çağrısıyla sonlandırılan herhangi bir sayıda MQPUT ya da MQGET çağrısı içeren bir iş birimi oluşturur.

Bir MQCMIT kapsamındaki MQPUT ya da MQGET çağrılarının sayısı arttıkça, kesinleştirmenin performans maliyeti önemli ölçüde artar. Uygulamalar, genel olarak, tek bir eşitleme noktasında çok sayıda ileti MQPUT/MQGET ' i içermeyecek şekilde tasarlanmalıdır.

MAXUMSGS kuyruk yöneticisi özniteliğini kullanarak, herhangi bir tek eşitleme noktasındaki iletilerin sayısını denetimci olarak sınırlayabilirsiniz. Bir uygulama bu sınırı aşarsa, sınırı aşan MQPUT,MQPUT1ya da MQGET çağrısında MQRC_SYNCPOINT_LIMIT_ULAŞILDI değerini alır. Daha sonra uygulama MQCMIT ya da MQBACK komutunu uygun şekilde vermelidir.

MAXUMSGS varsayılan değeri 10000 'dir. Alt sınırı uygulamak istiyorsanız bu değer düşürülebilir, bu da döngülü uygulamalara karşı koruma sağlar. MAXUMSGS ürününü azaltmadan önce, var olan uygulamalarınızın sınırı aşmamasını sağlamak için bunları anladığınızdan emin olun ya da MQRC_SYNCPOINT_LIMIT_SIZE ulaşılan dönüş kodunu tolere edebildiğinizden emin olun

Süresi dolan iletiler

Süresi dolan iletiler, sonraki uygun MQGET çağrısıyla atılır. Ancak, böyle bir çağrı oluşmazsa, süresi dolan iletiler atılır ve bazı kuyruklar için, özellikle de ileti alma işleminin MessageId, CorrelIdya da GroupId tarafından yapıldığı ve kuyruk performans için dizinlendiği durumlarda, süresi dolan birçok ileti birikebilir. Kuyruk yöneticisi, süresi dolan iletiler için belirli aralıklarla herhangi bir kuyruğu tarayabilir ve daha sonra bu iletiler silinir. Bu taramayı hangi sıklıkta gerçekleşeceğini seçebilirsiniz. Bunu yapmanın iki yolu vardır:

Belirtik istek
Hangi kuyrukların taranacağını ve ne zaman taranacağını 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 kuyrukta süresi dolan iletilere ilişkin bilgileri saklar ve süresi dolan iletilere ilişkin taramanın ne zaman yararlı olacağını bilir. EXPRYINT aralığına her ulaşıldığında, kuyruk yöneticisi süresi dolan iletileri taramaya değer aday kuyrukları arar ve yalnızca değerli olduğunu düşündüğü kuyrukları tarar. Tüm kuyrukları taramaz. Bu, gereksiz taramalar için harcanan işlemci süresini önler.

Paylaşılan kuyruklar, kuyruk paylaşım grubunda yalnızca bir kuyruk yöneticisi tarafından taranır. Genellikle, yeniden başlatılır ilk kuyruk yöneticisi ya da EXPRYINT kümesi ilk kez tarama gerçekleştirir.

Not: Bir kuyruk paylaşım grubu içindeki tüm kuyruk yöneticileri için aynı EXPRYINT değerini ayarlamanız gerekir.

z/OS IBM MQ for z/OS ile verilen örnek tanımlar

Bu konuyu örnek JCL ve IBM MQ for z/OSile verilen kod için başvuru olarak kullanın.

Aşağıdaki örnek tanımlamalar, thlqual.SCSQPROC kitaplığında IBM MQ ile birlikte verilir. Bunları, sistem nesnelerini tanımlamak ve kendi nesnelerinizi uyarlamak için kullanabilirsiniz. Bunların bazılarını kullanıma hazırlama giriş veri kümelerine ekleyebilirsiniz (Başlatma komutlarında açıklanmıştır).

Çizelge 22. Sistem nesnelerine ilişkin IBM MQ örnek tanımlamaları

Başlangıç giriş verileri kümesi	**Örnek ad**
CSQINP1	CSQ4INP1 CSQ4INPR
CSQINP2	CSQ4INSA CSQ4INYS [1] CSQ4INSX CSQ4INSG CSQ4INSR CSQ4INSS CSQ4INSJ CSQ4INSM CSQ4INYG CSQ4INYR CSQ4INYC CSQ4INYD CSQ4INSC
CSQINPT	CSQ4INST CSQ4INYT
Diğer	CSQ4DISP CSQ4INPX CSQ4IVPQ CSQ4IVPG CSQ4MSTR CSQ4MSRR CSQ4QMIN

Not:

1. Bu örnek tanımlarının sırası önemlidir: INYS, INSX ve INSG yanlış sipariş edilirse bir hata oluşur.

CSQINP1 örnekleri

Her ileti sınıfı için bir sayfa kümesi kullanıyorsanız, örnek CSQINP1 veri kümesi thlqual.SCSQPROC(CSQ4INP1) ya da iletinin ana sınıfları için birden çok sayfa kümesi kullanırken thlqual.SCSQPROC(CSQ4INPR) değerini kullanın. Arabellek havuzlarına ilişkin tanımlamaları, arabellek havuzu ilişkilendirmeleri için sayfa kümesi ve ALTER SECURITY komutunu içerir. Örneği, kuyruk yöneticisi tarafından başlatılan görev yordamının CSQINP1 birleştirmesine ekleyin.

CSQINP2 örnekleri

CSQ4INSG sistem nesnesi örneği

Örnek CSQINP2 veri kümesi thlqual.SCSQPROC(CSQ4INSG), genel kullanım için aşağıdaki sistem nesnelerine ilişkin tanımlamaları içerir:

- Sistem varsayılan nesneleri
- Sistem komut nesneleri
- Sistem yönetimi nesneleri
- Sistem kullanımı için diğer nesneler

Bu örnekteki nesneleri tanımlamanız gerekir, ancak bunu altsistem ilk başlatıldığında yalnızca bir kez yapmanız gerekir. CSQINP2 veri kümesindeki tanımlamaların içermesi, bunu yapmanın en iyi yoludur. Bunlar, kuyruk yöneticisi kapatma ve yeniden başlatma işlemi boyunca korunur. Nesne adlarını değiştirmemelisiniz, ancak gerekiyorsa özniteliklerini değiştirebilirsiniz.

Aşağıdaki koşullar karşılandığında, SYSTEM.DURABLE.SUBSCRIBER.QUEUE kuyruğu (yayınlama aboneliği etkin olmasa bile):

- QMGR özniteliği PSMODE, DISABLED olarak ayarlandı
- CSQ4INST deyimi `DEFINE SUB('SYSTEM.DEFAULT.SUB')` örnek nesnesi var.

Bunu önlemek için `DEFINE SUB('SYSTEM.DEFAULT.SUB')` deyimini silin ya da açıklama satırı yapın.

JMS 2.0 teslim gecikmesi hazırlama kuyruğu, SYSTEM.DDELAY.LOCAL.QUEUE yalnızca JMS 2.0 teslim gecikmesi kullanılıyorsa tanımlanmalıdır. Varsayılan olarak, kuyruk tanımlaması açıklama satırı olur; gerekiyorsa, bu açıklama satırı olmaktan çıkarılabilir.

CSQ4INSA sistem nesnesi ve kimlik doğrulama örneği

Örnek CSQINP2 veri kümesi thlqual.SCSQPROC(CSQ4INSA), kanal doğrulama sistemi kuyruk tanımlamasını içerir. Bu kuyruk, kanal kimlik doğrulama kayıtlarını içerir. Ayrıca, varsayılan kanal kimlik doğrulama kurallarını da içerir.

Kuyruk yöneticisinde CHLAUTH ETKİN ise ve kanalları çalıştırmak istiyorsanız ya da SET ya da DISPLAY CHLAUTH kaydını kullanmak istiyorsanız, bu örnekteki nesneleri tanımlamanız gerekir. Altsistem ilk başlatıldığında bunları yalnızca bir kez tanımlamanız gerekir. CSQINP2 veri kümesindeki tanımlamaların içermesi, bunu yapmanın en iyi yoludur. Bunlar bir kuyruk yöneticisi kapatma ve yeniden başlatma işlemi boyunca tutulur, kuyruk adını değiştirmemelisiniz.

CSQ4INSS sistem nesnesi örneği

Kuyruk paylaşım gruplarını kullanıyorsanız, ek sistem nesneleri tanımlayabilirsiniz.

Örnek veri kümesi thlqual.SCSQPROC(CSQ4INSS), CF yapılarıyla kullanılmak üzere örnek komutlar ve paylaşılan kanallar ve grup içi kuyruğa alma için gereken sistem nesneleri için bir dizi tanımlama içerir.

Bu örneği olduğu gibi kullanamazsınız; kullanmadan önce özelleştirmeniz gerekir. Daha sonra bu üyeyi, kuyruk yöneticisi başlatma yordamının CSQINP2 DD birleştirmesine ekleyebilir ya da gerekli komutları vermek için CSQUTIL yardımcı programının COMMAND işlevine giriş olarak kullanabilirsiniz.

Grup ya da paylaşılan nesneler tanımlarken, bunları kuyruk paylaşım grubundaki tek bir kuyruk yöneticisi için CSQINP2 DD birleştirmesine eklemeniz gerekir.

CSQ4INSX sistem nesnesi örneği

Dağıtılmış kuyruğa alma ve kümeleme kullanıyorsanız, ek sistem nesneleri tanımlamanız gerekir.

Örnek veri kümesi thlqual.SCSQPROC(CSQ4INSX) gerekli kuyruk tanımlamalarını içerir. Bu üyeyi, kuyruk yöneticisi başlatma yordamının CSQINP2 DD birleştirmesine ekleyebilir ya da gerekli DEFINE komutlarını vermek için CSQUTIL yardımcı programındaki COMMAND işlevine giriş olarak kullanabilirsiniz.

İki tip nesne tanımlaması vardır:

- SYSTEM.CHANNEL.xx, dağıtılmış kuyruğa alma için gerekli
- SYSTEM.CLUSTER.xx, kümeleme için gerekli

CSQ4INSJ system JMS nesnesi örneği

JMS yayınlama/abone olma etki alanında kullanılan kuyrukları tanımlar.

CSQ4INSM sistem nesnesi örneği

Gelişmiş ileti güvenliği kullanıyorsanız, ek sistem nesneleri tanımlamanız gerekir. Örnek veri kümesi thlqual.SCSQPROC(CSQ4INSM) gerekli kuyruk tanımlamalarını içerir.

CSQ4INSR nesne örneği

WebSphere Application Server ve aracılar tarafından kullanılan kuyrukları tanımlar.

CSQ4INYD nesnesi örneği

Dağıtımlı kuyruklama kullanıyorsanız ve kendi kuyruklarınızı, süreçlerinizi ve kanallarınızı ayarlamanız gerekiyorsa.

Örnek veri kümesi thlqual.SCSQPROC(CSQ4INYD), dağıtılmış kuyruğa alma nesnelerinizi uyarlamak için kullanabileceğiniz örnek tanımlamaları içerir. Aşağıdakilerden oluşur:

- Gönderme ucuna ilişkin tanımlama kümesi
- Alma ucuna ilişkin tanımlama kümesi
- İstemcileri kullanmak için bir tanımlama kümesi

Bu örneği olduğu gibi kullanamazsınız-kullanmadan önce özelleştirmeniz gerekir. Daha sonra bu üyeyi, kuyruk yöneticisi başlatma yordamının CSQINP2 DD birleştirmesine ekleyebilir ya da gerekli DEFINE komutlarını vermek için CSQUTIL yardımcı programının COMMAND işlevine giriş olarak kullanabilirsiniz. (Bu, kuyruk yöneticisini her yeniden başlattığınızda bu nesneleri yeniden tanımlamanıza gerek olmadığı anlamına geldiğinden tercih edilir).

CSQ4INYC nesnesi örneği

Kümeleme kullanıyorsanız, gerektiğinde, dağıtılmış kuyruğa alma işleminin kanal tanımlarına ve uzak kuyruk tanımlarına eşdeğer tanımlar otomatik olarak yaratılır. Ancak, bazı el ile kanal tanımlamaları gerekir-küme için bir küme alıcı kanalı ve en az bir küme havuzu kuyruk yöneticisine bir küme gönderen tanımlaması.

Örnek veri kümesi: thlqual.SCSQPROC(CSQ4INYC), kümeleme nesnelerinizi uyarlamak için kullanabileceğiniz örnek tanımlamaları içerir:

- Kuyruk yöneticisine ilişkin tanımlar
- Alan kanala ilişkin tanımlar
- Gönderen kanala ilişkin tanımlar
- Küme kuyruklarına ilişkin tanımlar
- Küme listelerine ilişkin tanımlar

Bu örneği olduğu gibi kullanamazsınız-kullanmadan önce özelleştirmeniz gerekir. Daha sonra bu üyeyi, kuyruk yöneticisi başlatma yordamının CSQINP2 DD birleştirmesine ekleyebilir ya da gerekli DEFINE komutlarını vermek için CSQUTIL yardımcı programının COMMAND işlevine giriş olarak kullanabilirsiniz. Bu, IBM MQ' i her yeniden başlattığınızda bu nesneleri yeniden tanımlamanıza gerek olmadığı anlamına geldiği için tercih edilir.

CSQ4INYG nesnesi örneği

Örnek veri kümesi: thlqual.SCSQPROC(CSQ4INYG), genel kullanım için kendi nesnelerinizi uyarlamak üzere kullanabileceğiniz aşağıdaki örnek tanımlamaları içerir:

- Gönderilmeyen iletiler kuyruğu
- Varsayılan iletim kuyruğu
- CICS bağdaştırıcı nesneleri

Bu örneği olduğu gibi kullanamazsınız-kullanmadan önce özelleştirmeniz gerekir. Daha sonra bu üyeyi, kuyruk yöneticisi başlatma yordamının CSQINP2 DD birleştirmesine ekleyebilir ya da gerekli DEFINE komutlarını vermek için CSQUTIL yardımcı programının COMMAND işlevine giriş olarak kullanabilirsiniz. Bu, IBM MQ' i her yeniden başlattığınızda bu nesneleri yeniden tanımlamanıza gerek olmadığı anlamına geldiği için tercih edilir.

Buradaki örnek tanımlamalara ek olarak, sistem nesnesi tanımlamalarını kendi kaynak tanımlamalarınızın temeli olarak kullanabilirsiniz. Örneğin, SYSTEM.DEFAULT.LOCAL.QUEUE VE ADLANDıRıN MY.DEFAULT.LOCAL.QUEUE. Daha sonra bu kopyadaki parametrelerden herhangi birini gerektiği şekilde değiştirebilirsiniz. Daha sonra, bu tipte kaynak yaratma yetkiniz varsa, seçtiğiniz yönteme göre bir DEFINE komutu vermeniz gerekir.

Varsayılan iletim kuyruğu
Varsayılan bir iletim kuyruğu tanımlamak isteyip istemediğinize karar vermeden önce Varsayılan iletim kuyruğu tanımlamasını okuyun.

- Varsayılan bir iletim kuyruğu tanımlamak istediğinize karar verirseniz, bunu sunmak için bir kanal da tanımlamanız gerektiğini unutmayın.
- Tanımlamak istemediğinize karar verirseniz, örnekteki ALTER QMGR komutundan DEFXMITQ deyimini kaldırmayı unutmayın.

CICS bağdaştırıcı nesneleri
Örnek, CICS01.INITQ. Bu kuyruk, IBM MQ tarafından sağlanan CKTI işlemi tarafından kullanılır. Bu kuyruğun adını değiştirebilirsiniz; ancak, kuyruğun INITPARM deyimindeki CICS sistem kullanıma hazırlama çizelgesinde (SIT) ya da SYSIN geçersiz kılma değerinde belirtilen adla eşleşmesi gerekir.

CSQ4INYS/CSQ4INYR nesne örnekleri

Kullanmak için depolama sınıfı tanımlamaları:

- her ileti sınıfı için bir sayfa kümesi
- iletinin ana sınıfları için birden çok sayfa kümesi

Örneğin, SYSTEM.COMMAND.INPUT , STGCLASS ('SYSVOLAT') ve SYSTEM.CLUSTER.TRANSMIT.QUEUE KUYRUğU, STGCLASS ('REMOTE') KULLANıYOR. CSQ4INYS' de, bu depolama sınıflarının her ikisi de aynı sayfa kümesini kullanır. CSQ4INYR' de, bu depolama sınıfları iletim kuyruğu dolgusu etkisini azaltmak için farklı sayfa kümeleri kullanır.

CSQINPT örnekleri

CSQ4INST
Örnek veri kümesi: thlqual.SCSQPROC(CSQ4INST), sistem varsayılan aboneliğine ilişkin tanımı içerir.

Nesneyi bu örnekte tanımlamanız gerekir, ancak bunu yayınlama/abone olma altyapısı ilk kez başlatıldığında yalnızca bir kez yapmanız gerekir. Tanımlamanın CSQINPT veri kümesine eklenmesi, bunu yapmanın en iyi yoludur. Kuyruk yöneticisi kapatma ve yeniden başlatma işlemi boyunca korunur. Nesne adını değiştirmemelisiniz, ancak gerekirse özniteliklerini değiştirebilirsiniz.

CSQ4INYT
Örnek veri kümesi: thlqual.SCSQPROC(CSQ4INYT), yayınlama/abone olma altyapısı başlatıldığında çalıştırmak isteyebileceğiniz bir komut kümesi içerir. Bu örnek, Konu ve Abonelik bilgilerini görüntüler.

Diğer

CSQ4DISP görüntü örneği

Örnek veri kümesi: thlqual.SCSQPROC(CSQ4DISP), kuyruk yöneticinizde tanımlı tüm kaynakları görüntüleyen bir soysal DISPLAY komutları kümesi içerir. Bu, depolama sınıfları ve izleme gibi tüm IBM MQ nesnelerine ve tanımlamalarına ilişkin tanımlamaları içerir. Bu komutlar büyük miktarda çıkış

oluşturabilir. Bu örneği CSQINP2 veri kümesinde ya da CSQUTIL yardımcı programının COMMAND işlevine giriş olarak kullanabilirsiniz.

CSQ4INPX Sample

Örnek veri kümesi: thlqual.SCSQPROC(CSQ4INPX), kanal başlatıcısı her başlatıldığında yürütmek isteyebileceğiniz bir komut kümesi içerir. Bu örneği kullanmadan önce özelleştirmeniz gerekir; daha sonra kanal başlatıcısı için CSQINPX veri kümesine ekleyebilirsiniz.

CSQ4IVPQ ve CSQ4IVPG örnekleri

Örnek veri kümeleri: thlqual.SCSQPROC(CSQ4IVPQ) ve thlqual.SCSQPROC(CSQ4IVPG), kuruluş doğrulama programlarını (IVPs) çalıştırmak için gereken DEFINE komut kümelerini içerir.

Bu örnekleri CSQINP2 veri kümesine ekleyebilirsiniz. VPN ' leri başarıyla çalıştırdığınızda, kuyruk yöneticisi her yeniden başlatıldığında bunları yeniden çalıştırmanız gerekmez. Bu nedenle, bu örnekleri CSQINP2 birleşiminde kalıcı olarak saklamanız gerekmez.

CSQ4MSTR ve CSQ4MSRR örnekleri

Kuyruk yöneticisine ilişkin örnek başlatılan görev yordamları şunlardır: thlqual.SCSQPROC(CSQ4MSTR) ve thlqual.SCSQPROC(CSQ4MSRR).

CSQ4MSRR , önemli kuyrukların farklı sayfa kümelerine dağıtılması için CSQINP2 birleştirmesinde CSQ4INYR kullanır.

Gerekirse, yeni yaratılan kuyruk yöneticileri için CSQMINI kartını kullanabilmek üzere açıklamaları kaldırabilirsiniz.

CSQ4QMIN örneği

Örnek bir QMINI veri kümesi, thlqual.SCSQPROC(CSQ4QMIN).

QMINI veri kümesi ve **`TransportSecurity`** kısmına ilişkin ayrıntılar için bkz. QMINI veri kümesi .

z/OS z/OS üzerinde kurtarma ve yeniden başlatma

Yeniden başlatma ve kurtarma için IBM MQ for z/OS özellikleri hakkında bilgi edinmek üzere bu konudaki bağlantıları kullanın.

IBM MQ for z/OS , yeniden başlatma ve kurtarma için güçlü özelliklere sahiptir. Bir kuyruk yöneticisinin durduktan sonra nasıl kurtarılacağına ve yeniden başlatıldığında ne olacağına ilişkin bilgi için aşağıdaki alt konulara bakın:

- “IBM MQ for z/OS içindeki verilerde nasıl değişiklik yapılır?” sayfa 243
- “IBM MQ for z/OS içinde tutarlılığın nasıl korunduğu” sayfa 244
- “IBM MQ for z/OS içinde sona erdirme sırasında neler oluyor?” sayfa 246
- “IBM MQ for z/OS içinde yeniden başlatma ve kurtarma sırasında neler oluyor?” sayfa 248
- “Belirsiz kurtarma birimlerinin nasıl çözümlendiğini” sayfa 250
- “Paylaşılan kuyruk kurtarma” sayfa 252

İlgili kavramlar

z/OS IBM MQ for z/OS kurtarma işlemleri

İlgili görevler

Yedekleme ve kurtarma planlaması

z/OS Yönetmez/OS

İlgili başvurular

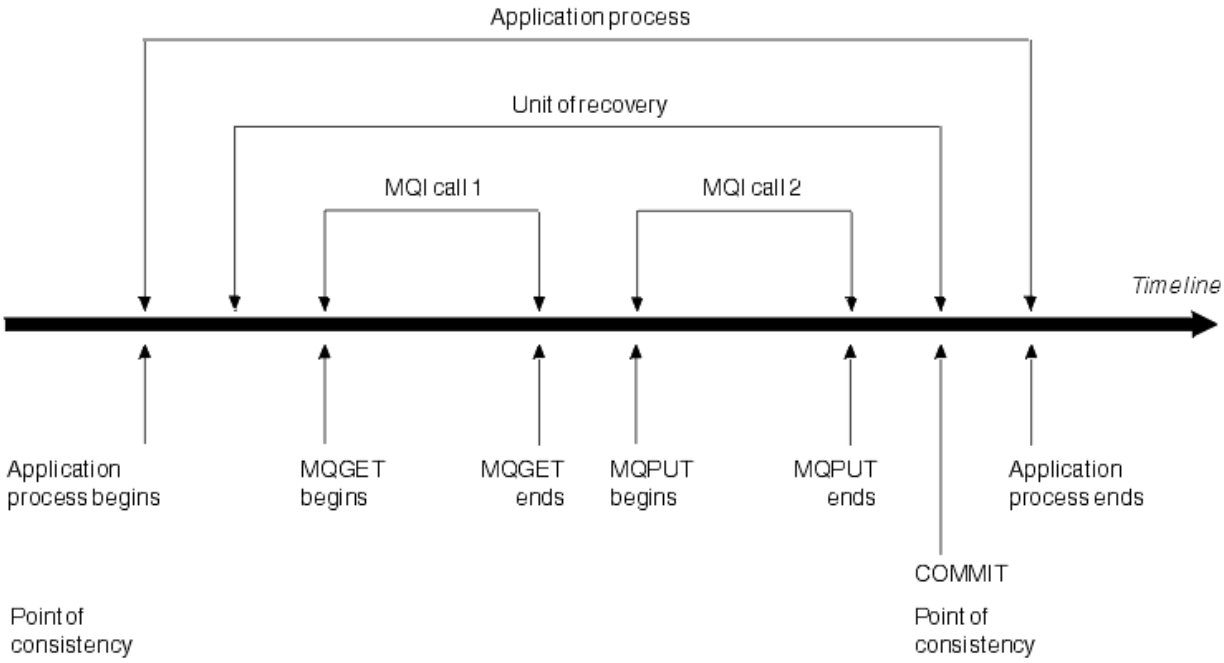
z/OS IBM MQ for z/OS ile ilgili iletiler

z/OS IBM MQ for z/OS içindeki verilerde nasıl değişiklik yapılır?

IBM MQ for z/OS , tüm verileri tutarlı tutmak için diğer altsistemlerle etkileşimde bulunmalıdır. Bu konuda, *kurtarma birimleri*, bunların ne oldukları ve *geri çıkışlarda*nasıl kullanıldıklarına ilişkin bilgiler yer alır.

Kurtarma birimleri

Kurtarma birimi , IBM MQ verilerini bir tutarlılık noktasından diğerine değiştiren bir uygulama programı için tek bir kuyruk yöneticisi tarafından yapılan işlemdir. *Tutarlılık noktası* (*syncpoint* ya da *kesinleştirme noktası* olarak da adlandırılır), bir uygulama programının eriştiği kurtarılabilir tüm verilerin tutarlı olduğu bir zaman noktasıdır.



Şekil 75. Bir uygulama programı içindeki kurtarma birimi

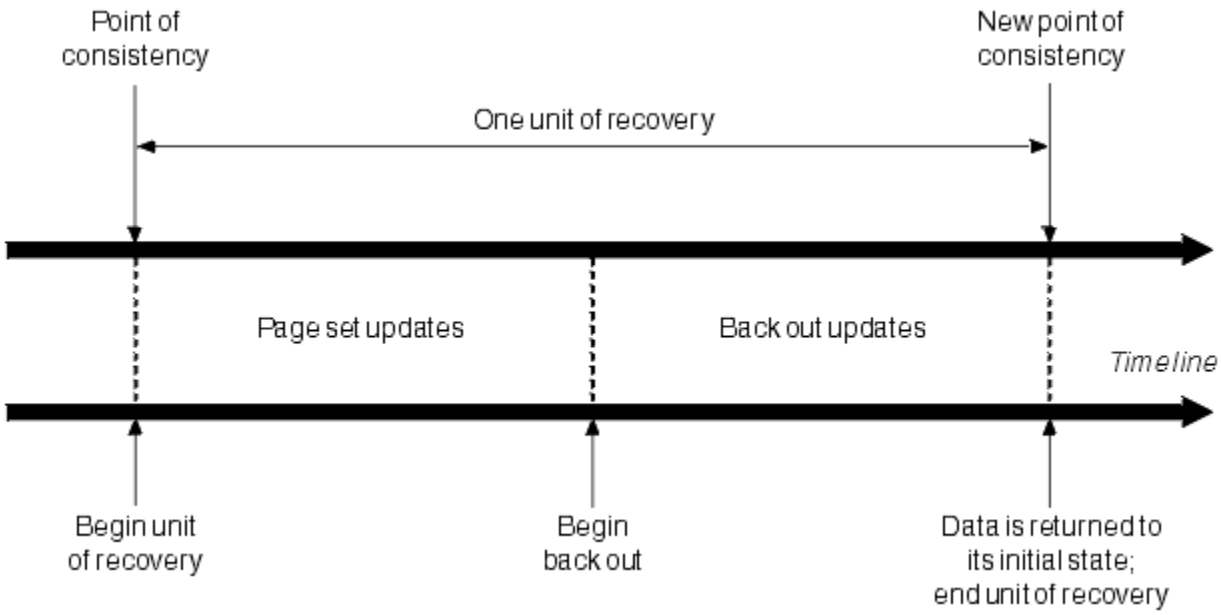
Bir kurtarma birimi, programın başlangıcından sonra ya da önceki tutarlılık noktasından sonra verilerde yapılan ilk değişiklikle başlar; daha sonra tutarlılık noktasıyla biter. Şekil 75 sayfa 243 içinde, kurtarma birimleri, tutarlılık noktası ve bir uygulama programı arasındaki ilişki gösterilir. Bu örnekte, uygulama programı MQI çağrıları 1 ve 2 aracılığıyla kuyruklarda değişiklik yapar. Uygulama programı birden çok kurtarma birimi ya da tek bir kurtarma birimi içerebilir. Ancak, tüm kurtarma birimleri bir kesinleştirme noktasında sona erer.

Örneğin, bir banka işlemi fonları bir hesaptan diğerine aktarır. İlk olarak, program ilk hesaptan, hesap A ' dan tutarı çıkarır. Sonra, ikinci hesaba, B ' ye miktarı ekler. Tutar A ' dan çıkarıldıktan sonra, iki hesap tutarsız olur ve IBM MQ kesinleştirilemez. Tutar B hesabına eklendiğinde tutarlılaşırlar. Her iki adım da tamamlandığında, program bir kesinleştirme yoluyla bir tutarlılık noktası duyurabilir ve değişiklikleri diğer uygulama programları tarafından görülebilir hale getirir.

Bir uygulama programının normal olarak sona erdirilmesi, otomatik olarak bir tutarlılık noktasına neden olur. CICS ve IMS programlarındaki bazı program istekleri de bir tutarlılık noktasına neden olur; örneğin, EXEC CICS SYNCPOINT.

İşi geri çekiyor

Bir kurtarma biriminde bir hata oluşursa, IBM MQ verilerde yapılan değişiklikleri kaldırır ve verileri kurtarma biriminin başlangıcında durumuna döndürerek; yani, IBM MQ işi geri alır. Olaylar Şekil 76 sayfa 244içinde gösterilir.



Şekil 76. Geri dönüş gösteren bir kurtarma birimi

z/OS IBM MQ for z/OS içinde tutarlılığın nasıl korunduğu

IBM MQ for z/OS içindeki veriler toplu iş, CICS, IMSya da TSO ile tutarlı olmalıdır. Birinde değiştirilen veriler, diğerinde yapılan bir değişikle eşleşmelidir.

Bir sistem, değiştirilen verileri kesinleştirmeden önce, diğer sistemin ilgili değişikliği yapabileceğini bilmelidir. Bu yüzden, sistemler iletişim kurmalı.

İki aşamalı kesinleştirme sırasında (örneğin, CICS altında), bir altsistem işlemi koordine eder. Bu altsisteme *koordinatör*adı verilir; diğeri *katılımcı*' dır. CICS ya da IMS , IBM MQile etkileşimlerde her zaman koordinatördür ve IBM MQ her zaman katılımcıdır. Toplu işte ya da TSO ortamında IBM MQ , z/OS RRS tarafından koordine edilen iki aşamalı kesinleştirme protokollerine katılabilir.

Tek fazlı kesinleştirme sırasında (örneğin TSO ya da toplu iş altında), IBM MQ her zaman etkileşimlerdeki eşgüdümcüdür ve kesinleştirme işlemini tam olarak denetler.

WebSphere Application Server ortamında, JMS oturum nesnesinin anlambilimi, tek fazlı mı, yoksa iki fazlı kesinleştirme eşgüdümünün mi kullanılacağını belirler.

CICS ya da IMS ile tutarlılık

IBM MQ ve CICS ya da IMS arasındaki bağlantı aşağıdaki eşitleme noktası iletişim kurallarını destekler:

- İki aşamalı kesinleştirme-birden çok kaynak yöneticisinin sahip olduğu kaynakları güncelleyen hareketler için.

  Bu, standart dağıtılmış syncpoint protokolüdür. Tek fazlı kesinleştirmekten daha fazla günlüğe kaydetme ve ileti akışı içerir.
- Tek aşamalı kesinleştirme-tek bir kaynak yöneticisine ait kaynakları güncelleyen işlemler için (IBM MQ).

  Bu protokol, günlüğe kaydetme ve ileti akışları için eniyilenmiştir.

- IBM MQ içeren, ancak kuyruk yöneticisinde bir eşitleme noktası gerektiren (örneğin, bir kuyruğa göz atma) hiçbir şey yapmayan hareketler için eşitleme noktasının atlanması.

Her bir durumda CICS ya da IMS , syncpoint yöneticisi olarak işlev görür.

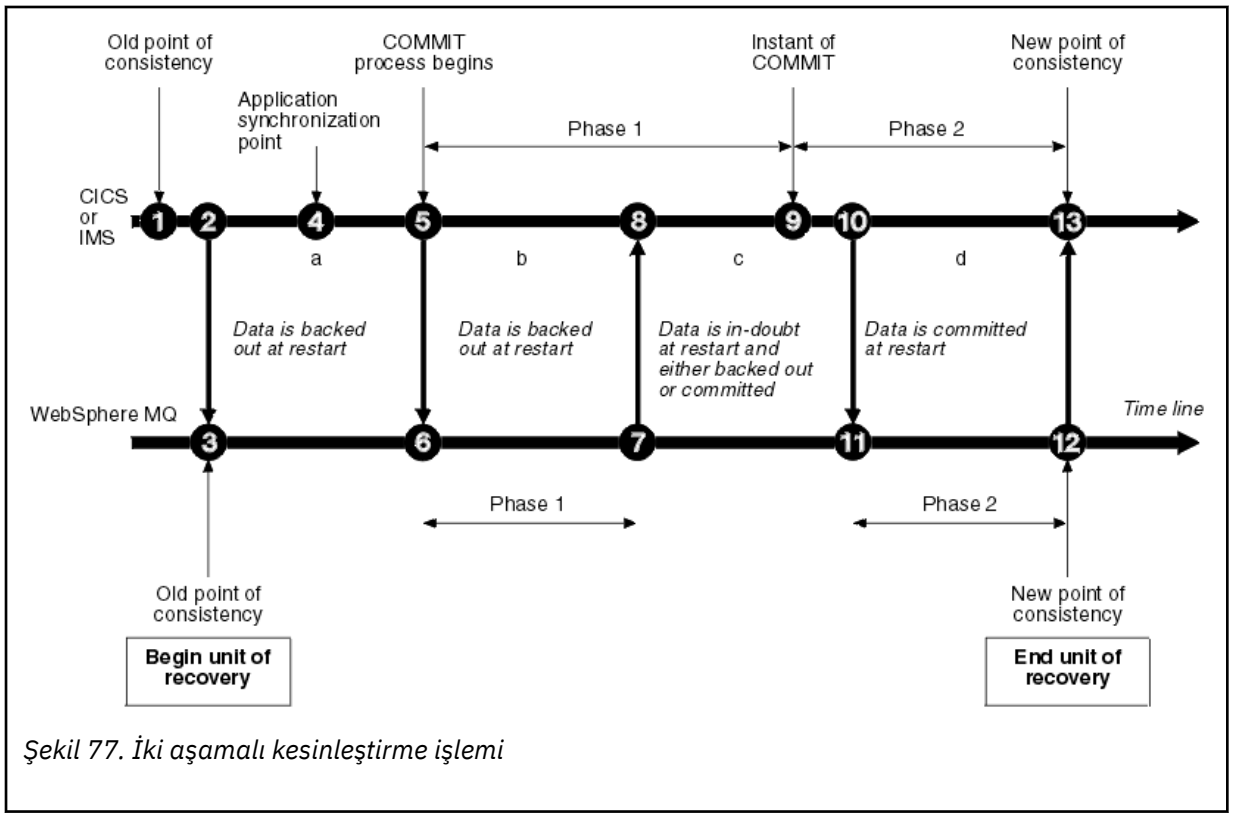
IBM MQ ' in CICS ya da IMS ile iletişim kurmak için kullandığı iki aşamalı kesinleştirme aşamaları şunlardır:

1. Aşama 1 'de, her sistem günlük kaydına yeterli kurtarma bilgisi kaydedip kaydetmediğini bağımsız olarak belirler ve işini kesinleştirebilir.

   Fazın sonunda, sistemler iletişim kurar. Kabul ederlerse, her biri bir sonraki aşamaya başlar.
2. 2. aşamada, değişiklikler kalıcı hale getirilir. Sistemlerden biri 2. aşama sırasında olağandışı sona ererse, işlem yeniden başlatma sırasında kurtarma işlemi tarafından tamamlanır.

İki fazlı kesinleştirme işleminin resmi

Şekil 77 sayfa 245 içinde iki aşamalı kesinleştirme işlemi gösterilmektedir. CICS ya da IMS eşgüdümcüsündeki olaylar üst çizgide, olaylar alt çizgideki IBM MQ içinde gösterilir.



Şekil 77. İki aşamalı kesinleştirme işlemi

Aşağıdaki bölümdeki numaralar, şekilde gösterilenlerle bağlantılıdır.

1. Eşgüdümcüdeki veriler tutarlılık noktasında.
2. Eşgüdümcüdeki bir uygulama programı, ileti ekleyerek bir kuyruğu güncellemek için IBM MQ ' i çağırır.
3. Bu, IBM MQiçinde bir kurtarma birimini başlatır.
4. Bir uygulama eşitleme noktasına ulaşılıncaya kadar eşgüdümcüde işlem devam eder.
5. Eşgüdümcü daha sonra kesinleştirme işlemini başlatır. CICS programları, kesinleştirmeyi başlatmak için bir SYNCPOINT komutu ya da olağan bir uygulama sonlandırması kullanır. IMS programları, bir CHKP çağrısı, SYNC çağrısı, IOPCB ' ye GET UNIQUE çağrısı ya da normal bir uygulama sonlandırması kullanarak kesinleştirmeyi başlatabilir. Kesinleştirme işleminin 1. aşaması başlıyor.
6. Eşgüdümcü 1. aşamayı işlemeye başladıkça, IBM MQda başlar.

7. IBM MQ , 1. aşamayı başarıyla tamamlar, bu durumu günlüğüne yazar ve eşgüdümcüya bildirir.
8. Eşgüdümcü bildirimi alır.
9. Eşgüdümcü, 1. evre işlemesini başarıyla tamamlar. Şimdi her iki altsistem de veri değişikliklerini kesinleştirmeyi kabul eder, çünkü her ikisi de 1. aşamayı tamamlamış ve herhangi bir hatadan kurtulabilmiştir. Eşgüdümcü, günlüğünde kesinleştirme anını kaydeder-iki altsistemin değişiklikleri yapması için geri alınamaz kararı.

   Koordinatör şimdi işlemin 2. aşamasına başlıyor-gerçek kesinleştirme.
10. Eşgüdümcü, IBM MQ ' e 2. aşamasını başlatmasını bildirir.
11. IBM MQ , 2. aşamanın başlangıcını günlüğe kaydeder.
12. 2. aşama başarıyla tamamlandı ve bu artık IBM MQiçin yeni bir tutarlılık noktasıdır. IBM MQ daha sonra, eşgüdümcüden 2. evre işlemesini tamamladığını bildirir.
13. Eşgüdümcü 2. evreyi işlemeyi bitirir. Her iki altsistem tarafından denetlenen veriler artık tutarlı ve diğer uygulamalar tarafından kullanılabilir.

Olağandışı bir sonlandırmadan sonra tutarlılık nasıl korunur

Olağandışı bir sonlandırma sonrasında bir kuyruk yöneticisi yeniden başlatıldığında, sona erdirme sırasında etkin olan kurtarma birimlerinin kesinleştirilip kesinleştirilmeyeceğini ya da geri çekilip çekilmeyeceğini belirlemelidir. Bazı kurtarma birimleri için, IBM MQ karar vermek için yeterli bilgiye sahiptir. Diğerleri için, bağlantı yeniden kurulduğunda eşgüdümcüden bilgi almaz ve almalıdır.

Şekil 77 sayfa 245 , iki aşama içinde dört dönemi gösterir: a, b, c ve d. Bir kurtarma biriminin durumu, sonlandırmanın gerçekleştiği süreye bağlıdır. Durum aşağıdakilerden biri olabilir:

Uçarken
Kuyruk yöneticisi, 1. aşama (a ya da b dönemi) tamamlanmadan sona erdi; yeniden başlatma sırasında IBM MQ güncellemeleri geri alır.

Belirsiz
Kuyruk yöneticisi 1. aşamayı bitirdikten sonra ve 2. aşamayı (c dönemi) başlatmadan önce sona erdi; hatanın kesinleştirmeden önce mi, yoksa sonra mı (9 noktası) olduğunu yalnızca eşgüdümcü bilir. Daha önce gerçekleşmişse, IBM MQ değişiklikleri geri almalıdır; daha sonra gerçekleşirse, IBM MQ değişiklikleri yapmalı ve bunları kesinleştirmelidir. Yeniden başlatma sırasında IBM MQ , bu kurtarma birimini işlemeden önce eşgüdümcüden bilgi bekler.

Kesinleştirmede
Kuyruk yöneticisi, kendi 2. aşama işlemesi (d dönemi) başladıktan sonra sona erdi; kesinleştirilmiş değişiklikler yaptı.

Arka planda
Bir kurtarma birimi yedeklenmeye başladıktan sonra kuyruk yöneticisi sona erdi, ancak yeniden başlatma sırasında işlem tamamlanmadan (şekilde gösterilmeden) önce IBM MQ değişiklikleri yedeklemeye devam eder.

z/OS IBM MQ for z/OS içinde sona erdirme sırasında neler oluyor?

STOP QMGR komutuna yanıt olarak bir kuyruk yöneticisi olağan şekilde sonlandırılır. Bir kuyruk yöneticisi başka bir nedenle durursa, sonlandırma olağandışı olur.

Kuyruk yöneticisi sonlandırılırken IBM MQ ' in komutu dahili olarak yayınladığını unutmayın.

```
DISPLAY CONN(*) TYPE(CONN) ALL WHERE (APPLTYPE NE SYSTEMAL)
```

Böylece, hangi iş parçacıklarının kuyruk yöneticisinin sona erdirme işlemini tamamlamasını engelleyebileceğini bilebilirsiniz.

SYSTEMAL, SYSTEM ya da CHINIT ' in APPLTYPES ile eşleşir; bu nedenle, DISPLAY CONN komutu süzgeç uygulama tipleri SYSTEMAL ile eşleşmez, olağan kapanmayı önleyebilecek iş parçacıklarıyla ilgili iş günlüğü bilgilerine döner.

Normal sonlandırma

Normal bir sonlandırmada IBM MQ , tüm etkinliği düzenli bir şekilde durdurur. IBM MQ ' i susturma, zorlama ya da yeniden başlatma kipini kullanarak durdurabilirsiniz. Etkiler Çizelge 23 sayfa 247içinde verilir.

Çizelge 23. QUIESCE, FORCE ve RESTART kullanılarak sonlandırma

İş parçacığı tipi	**QUIESCE**	**ZORLA**	**YENİDEN BAŞLATMA**
Etkin iş parçacıkları	Tamamlanmak üzere çalıştır	İptal Et	İptal Et
Yeni iş parçacıkları	Başlayabilir	İzin verilmez	İzin verilmez
Yeni bağlantılar	İzin verilmez	İzin verilmez	İzin verilmez

Uygulama hala bağlıyken bir sonlandırma gerçekleştiğinde toplu iş uygulamaları bilgilendirilir.

CICSile, geçerli bir iş parçacığı yalnızca kurtarma biriminin sonuna kadar çalışır. CICSile, susturma kipinde bir kuyruk yöneticisinin durdurulması CICS bağdaştırıcısını durdurur ve etkin bir görev birden çok kurtarma birimi içeriyorsa, görevin tamamlanıncaya kadar çalışması gerekmez.

Zorla ya da yeniden başlatma kipinde bir kuyruk yöneticisini durdurursanız, yeni iş parçacığı ayrılmaz ve bağlı iş parçacıkları üzerinde çalışma geriye işlenir. Bu kiplerin kullanılması, kesinleştirme işleme aşamaları arasında olan iş parçacıkları için belirsiz kurtarma birimleri yaratabilir. IBM MQ , denetleyici CICS, IMSya da RRS altsistemiyle yeniden bağlandığında bunlar çözülür.

Bir kuyruk yöneticisini herhangi bir kipte durdurduğunuzda, adımlar şunlardır:

1. Bağlantılar sona erdirilir.
2. IBM MQ komutu kabul etmeyi durdurur.
3. IBM MQ , sayfa kümelerinde bekleyen güncelleştirmelerin tamamlanmasını sağlar.
4. DISPLAY USAGE komutu, yeniden başlatma RBA 'sını z/OS konsol günlüğüne kaydetmek için IBM MQ tarafından dahili olarak verilir.
5. Kapatma denetim noktası alınır ve BSDS güncellenir.

Susturma kipini belirten sonlandırmalar, belirsiz kurtarma birimlerini etkilemez. Şüpheye düşen her birim şüphe içinde kalır.

Olağandışı sonlandırma

Olağandışı bir sonlandırma, verileri tutarsız durumda bırakabilir; örneğin:

- Bir kurtarma birimi, tutarlılık noktasına ulaşmadan önce kesildi.
- Kesinleştirilen veriler sayfa kümelerine yazılmadı.
- Kesinleştirilmemiş veriler sayfa kümelerine yazıldı.
- Bir uygulama programı, kesinleştirme işleminin 1. aşaması ile 2. aşaması arasında kesintiye uğradı ve kurtarma birimi şüpheli duruma geldi.

IBM MQ , yeniden başlatma ve kurtarma sırasında olağandışı sonlanmasından kaynaklanan veri tutarsızlıklarını çözer.

z/OS IBM MQ for z/OS içinde yeniden başlatma ve kurtarma sırasında neler oluyor?

IBM MQ , yeniden başlatıldığında nelerin kurtarılacağını belirlemek için kurtarma günlüğünü ve önyükleme veri kümesini (BSDS) kullanır. BSDS, etkin ve arşiv günlüğü veri kümelerini ve günlükteki en son IBM MQ denetim noktasının konumunu tanımlar.

Yeniden başlatma ve kurtarma işlemine giriş

IBM MQ başlatıldıktan sonra, kuyruk yöneticisi yeniden başlatma işlemi aşağıdaki gibi gerçekleşir:

- Günlük başlatma
- Geçerli durum yeniden oluşturma
- İleriye dönük günlük kurtarma
- Geriye doğru günlük kurtarma
- Kuyruk dizini yeniden oluşturuluyor

Kurtarma tamamlandığında:

- Kesinleştirilen değişiklikler verilere yansıtılır.
- Belirsiz etkinlik verilere yansıtılır. Ancak, veriler kilitlenir ve IBM MQ belirsiz bir kararı tanıyıp üzerinde işlem yapmadan kullanılamaz.
- Kesintiye uğrayan ve iptal edilmekte olan değişiklikler kuyruklardan kaldırıldı. İletiler tutarlı ve kullanılabilir.
- Yeni bir denetim noktası alındı.
- Kalıcı iletiler içeren dizinlenmiş kuyruklar için yeni dizinler oluşturulmuştur (açıklamalar için bkz. “Kuyruk dizinlerinin yeniden oluşturulması” sayfa 249).

İkili BSDS' ler kullanılıyorsa, IBM MQ BSDS ' deki zaman damgalarının tutarlılığını denetler:

- BSDS ' nin her iki kopyası da güncel ise, IBM MQ iki zaman damgalarının eşit olup olmadığını sınar. Değilse, IBM MQ CSQJ120E iletisini yayınlar ve sonlandırır. BSDS ' nin iki kopyası ayrı DASD birimlerinde tutulduğunda ve birimlerden biri, kuyruk yöneticisi durdurulurken geri yüklendiğinde bu durum oluşabilir. IBM MQ , yeniden başlatma sırasında durumu algılar.
- BSDS ' nin bir kopyası ayrılırsa ve günlük kaydı tek bir BSDS ile devam ederse, bir sorun ortaya çıkabilir. BSDS ' nin her iki kopyası da tek bir birimde tutulursa ve birim geri yüklendiyse ya da her iki BSDS kopyası da ayrı olarak geri yüklendiyse, IBM MQ geri yüklemeyi algılayamayabilir. Bu durumda, BSDS ' de belirtilmeyen günlük kayıtları sistem tarafından bilinmez.

Uygulama bir bağlantı isteğinde bulunduktan *sonra* yeniden başlatma gerçekleştiğinde toplu iş uygulamaları bilgilendirilmez.

Kurtarma için gereken günlük aralığının anlaşılması

Yeniden başlatma sırasında, okunması gereken günlük verileri aralığı birçok etkene bağlıdır:

- Olağandışı bir sona erdirme sırasında, genellikle sistemde birçok tamamlanmamış iş birimi vardır. Daha önce açıklandığı gibi, yeniden başlatma işlemi, sistemi tutarlılık durumuna getirir; bu, iş birimlerinin geri çekilmesini ya da belirsiz iş birimlerindeki kilitlerin kurtarılmasını içerebilir. İş birimi kurtarma işlemi, iş günlüğü kayıtlarının tümünün boşta durması, geri çekilmesi ve belirsiz iş birimleri için kullanılabilir olmasını gerektirir. IBM MQ , eski iş birimlerini 'shunt' yapacak, böylece iş birimi kurtarma işlemi çok daha küçük bir günlük verileri aralığı kullanılarak gerçekleştirilebilir.
- Olağandışı bir sonlandırma sırasında, genellikle yalnızca arabellek havuzu önbelleğinde tutulan birçok kalıcı güncelleme vardır. Henüz diske yazılmamışlar. Bu değişiklikler günlükten okunmalı ve sayfa kümelerinde tutulan verilere yeniden uygulanmalıdır. Denetim noktasındaki sayfa kümesi kurtarma RNA ' ları, sayfa kümelerini tutarlı bir duruma güncellemek için gereken en düşük günlük RBA 'sını açıklar.

- Eski sayfa kümeleri sisteme tanıtıldıysa, örneğin, bir ortam hatasından kurtulmak için bir sayfa kümesi yedeği sunulduysa, tüm değişikliklerin yedeklemenin alındığı zamandan itibaren günlükten okunması gerekir. Bu değişiklikler, kurtarılmakta olan sayfa kümesinde tutulan verilere yeniden uygulanır. Sayfa kümesinin 0. sayfasında tutulan sayfa kümesi kurtarma RNA ' ları, bir sayfa kümesinin ortam kurtarması için gereken en düşük günlük RBA 'sını açıklar.
- Paylaşılan kuyruklarda kalıcı iletiler kullanılıyorsa, kalıcı ileti tutan CFSTRUCT ' leri kurtarmak için bir günlük verileri aralığı gerekir. CFSTRUCT kurtarma işlemi için gerekli olan en eski günlük verileri, eski CFSTRUCT BACKUP işlemi zamanından kalma verilerdir.

Olağan çalışma sırasında, bu etmenlerle ilişkili kurtarma günlüğü aralığını görüntülemek için DISPLAY USAGE TYPE (DATASET) komutu kullanılabilir (eski sayfa kümelerinin yeniden tanıtılması nedeniyle bilgi sağlayamıyor). Olağandışı bir sonlandırma durumunda kuyruk yöneticisinin yeniden başlatılmasını uzatabilecek sorunları önlemek için, DISPLAY USAGE TYPE (veri kümesi) içindeki değer çıkışını düzenli olarak izleyin.

Ayrıca, kuyruk yöneticisi aşağıdaki etmenlerle ilgili bilgi iletileri verir:

- CSQJ160I ve CSQJ161I uzun süreli iş birimleriyle ilgili uyarı.
- CSQR026I ve CSQR027I , uzun süre çalışan bu iş birimlerinin başarıyla geri alınıp alınmadığına ilişkin bilgi sağlar.
- CSQE040I ve CSQE041E yapı yedeklerinin eskimiş olduğu konusunda uyarır ve sonuç olarak RECOVER CFSTRUCT işlemi uzun sürer.

Hangi uygulamanın uzun süreli bir iş birimine sahip olduğunun belirlenmesi

Uzun süreli iş birimine sahip uygulama belirlenebilir. Bunu yapmak için DISPLAY CONN komutunu kullanın.

DISPLAY CONN komutu, kuyruk yöneticisine bağlı tüm uygulamalara ilişkin bağlantı bilgilerini ve şu anda uzun süreli bir iş birimine sahip olan uygulamaları belirlemenize yardımcı olacak ek bilgileri döndürür. DISPLAY CONN komutunun döndürdüğü bilgiler, DISPLAY QSTATUS komutunun döndürdüğü bilgilere benzer; ancak ana fark, DISPLAY CONN ' un belirli bir nesneyle ilişkilendirilmiş bağlantılara ilişkin ayrıntılar yerine, nesnelere ilişkin bilgileri ve belirli bir bağlantıya ilişkin hareket bilgilerini görüntülediğidir.

Bağlı her uygulama için, DISPLAY CONN komutu aşağıdaki bilgileri döndürür:

- Bağlantı Tanıtıcısı ve PID de içinde olmak üzere temel bilgiler.
- Hareketin yaratıldığı tarih ve saat (yani, ilk MQGET/PUT ' in syncpoint altında yapıldığı tarih) ve hareketin günlüğe ilk yazıldığı tarih de içinde olmak üzere, bu bağlantıya ilişkin işlemsel bilgiler.
- Hangi uygulamanın uzun süre çalışan bir iş birimine sahip olduğunu gösteren günlük süresi bilgileri.
- Bağlantının şu anda açık olduğu tüm nesnelerin listesi. Her nesneye ilişkin ayrıntılar, anahtar olarak kullanılan Bağlantı Tanıtıcısı ile ayrı bir ileti olarak döndürülür. Kuyruklar ve kuyruk yöneticileri gibi farklı nesne tipleri olduğu için, nesneyle birlikte görüntülenen bilgiler nesne tipine özgüdür.

Kuyruk dizinlerinin yeniden oluşturulması

İletilerin sıralı olarak alınmadığı bir kuyruktaki MQGET işlemlerinin hızını artırmak için, IBM MQ ' in o kuyruktaki tüm iletiler için ileti ya da ilinti tanıtıcılarının ya da grup tanıtıcılarının bir dizinini tutmasını 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. Dizinlenen kuyruklarınız çok sayıda kalıcı ileti içeriyorsa, bu, kuyruk yöneticisinin yeniden başlatılması için geçen süreyi artırır.

CSQ6SYSP makrosunun QINDXBLD parametresini kullanarak, dizinlerin kuyruk yöneticisi başlatması için zamanuyumsuz olarak yeniden oluşturulmasını seçebilirsiniz. QINDXBLD=NOWAIT değerini ayarlarsanız, IBM MQ dizinlerin yeniden oluşturulmasını beklemeden yeniden başlatır.

z/OS Belirsiz kurtarma birimlerinin nasıl çözümlendiğini

IBM MQ başka bir kaynak yöneticisiyle bağlantısını kaybederse, genellikle yeniden başlatma sırasında tutarsız tüm nesneleri kurtarmayı dener.

IBM MQ , CICS, IMSya da RRS ile bağlantısını kaybederse, olağan durumda yeniden başlatma sırasında tüm tutarsız nesneleri kurtarmayı dener. Belirsiz kurtarma birimlerini çözmek için gereken bilgiler, eşgüdüm sisteminden gelmelidir. Sonraki kısımlarda, farklı ortamlar için çözüm süreci açıklanır.

- Belirsiz kurtarma birimlerinin CICS içinden nasıl çözümlendiğinden emin olma
- Belirsiz kurtarma birimlerinin IMS içinden nasıl çözümlendiğinden emin olma
- Belirsiz kurtarma birimlerinin RRS ' den çözümlenmesi
- GROUP kurtarma birimi içeren belirsiz kurtarma birimleri nasıl çözülür

Belirsiz kurtarma birimleri CICS ' den nasıl çözülür?

Bazı durumlarda, CICS belirsiz kurtarma birimlerini çözmek için IBM MQ işlemini çalıştıramaz. Bu durumda, IBM MQ aşağıdaki iletilerden birini gönderir:

- CSQC404E
- CSQC405E
- CSQC406E
- CSQC407E

ardından CSQC408Iiletisi gelir.

Bu iletilerin ne anlama olduğu hakkında ayrıntılı bilgi için IBM MQ for z/OS iletileri, tamamlama ve neden kodları elkitabına bakın.

Belirsiz birimlerin çözülmesi CICS kaynaklarını etkilemez. CICS , kurtarma koordinasyonunun kontrolündedir ve yeniden başlatıldığında, kesinleştirmenin başlangıcını işaretleyen bir günlük kaydı olup olmadığına bağlı olarak her birimi otomatik olarak kesinleştirir ya da geri alır. Belirsiz nesnelerin varlığı, IBM MQ yeniden bağlanırken CICS kaynaklarını kilitlemez.

CICS bağdaştırıcısının işlevlerinden biri, verileri CICS ile IBM MQarasında eşitlenmiş olarak tutmaktır. CICS' e bağlıyken bir kuyruk yöneticisi olağandışı sona ererse, CICS , IBM MQ farkında olmadan işi kesinleştirebilir ya da geri çekebilir. Kuyruk yöneticisi yeniden başlatıldığında, bu iş *şüphe içinde*olarak adlandırılır.

IBM MQ , CICS bağlantısı yeniden başlatılıncaya ya da yeniden bağlanıncaya kadar bu belirsiz kurtarma birimlerini çözemez (yani, IBM MQ kaynaklarında yapılan değişiklikleri kesinleştiremez ya da geri çekemez).

CICS bağdaştırıcısı başlatılırken, belirsiz kurtarma birimlerini çözme işlemi başlatılır. İşlem, bağdaştırıcı belirsiz kurtarma birimlerinin bir listesini istediğinde başlar. Sonra:

- Bağdaştırıcı, IBM MQadresinden bu bağlantı tanıtıcısı için belirsiz kurtarma birimlerinin bir listesini alır ve bunları çözüm için CICS ' e iletir.
- CICS , bu listedeki girdileri kendi günlüğündeki girdilerle karşılaştırır. CICS , her belirsiz kurtarma birimi için hangi işlemi gerçekleştirdiğini kendi listesinden belirler.

Tüm çözümlenen birimler için IBM MQ , kuyrukları gerektiği şekilde günceller ve karşılık gelen kilitleri serbest bırakır. Yeniden başlatmadan sonra çözümlenmemiş birimler kalabilir. Bunları IBM MQ for z/OS Yönetimiiçinde açıklanan yöntemlerle çözün.

Belirsiz kurtarma birimleri IMS ' den nasıl çözülür?

IMSiçinde belirsiz kurtarma birimlerinin çözülmesi DL/I kaynaklarını etkilemez. IMS , kurtarma koordinasyonunu kontrol eder ve yeniden başlatıldığında, tamamlanmamış DL/I işini otomatik olarak kesinleştirir ya da geri alır. Çevrimiçi bölgeler (hızlı olmayan yol) için kesinleştirme ya da geri çekilme

kararı, IMS günlük kaydı tipleri X'3730 've X'3801' var ya da yok. Belirsiz kurtarma birimlerinin varlığı, IBM MQ bağlanıncaya kadar DL/I kayıtlarının kilitli olduğu anlamına gelmez.

Kuyruk yöneticisinin yeniden başlatılması sırasında IBM MQ , belirsiz kurtarma birimlerinin bir listesini oluşturur. IMS , artık kurtarma girişlerinin (RRE) kendi listesini oluşturur. RRE ' ler tüm girdiler çözümleninceye kadar IMS denetim noktalarında günlüğe kaydedilir.

Bir IMS bölgesinin IBM MQile yeniden bağlanması sırasında IMS , IBM MQ ' e IBM MQ içinde şüpheli olarak işaretlenmiş iş birimlerinin kesinleştirilip kesinleştirilmeyeceğini ya da geri çekilip çekilmeyeceğini gösterir.

Belirsiz birimler çözüldüğünde:

1. IBM MQ bir girdiyi kesinleştirmek üzere işaretlediğini ve IMS bunu geriletilmek üzere işaretlediğini algılarsa, IBM MQ CSQQ010Eiletisini yayınlar. IBM MQ , IBM MQ ile IMSarasında bu tipteki tüm tutarsızlıklar için bu iletiyi yayınlar.
2. IBM MQ ' in kalan belirsiz birimleri varsa, bağdaştırıcı CSQQ008Iiletisini yayınlar.

Tüm çözümlenen birimler için IBM MQ , kuyrukları gerektiği şekilde günceller ve karşılık gelen kilitleri serbest bırakır.

IBM MQ , çözülmemiş belirsiz bir işte kilitleri tutar. Bu, önemli kilitler tutuluyorsa sistemde bir birikim günlüğüne neden olabilir. IMS RRE ' lerini çözebilmeniz için bağlantı etkin kalır. Belirsiz iş parçacıklarını IBM MQ for z/OS Yönetimiiçinde açıklanan yöntemlerle kurtarın.

IMS soğuk başlatma gibi yazılım ya da işletim sorunları olmadığı sürece, tüm belirsiz işler çözülmelidir. IMS denetim bölgesi tarafından belirsiz bir çözüm iki durumda gerçekleşir:

1. IBM MQile bağlantının başlangıcında, bu sırada çözüm zamanuyumlu olarak gerçekleştirilir.
2. Bir program olağandışı sona erdiğinde, bu sırada çözüm zamanuyumsuz olarak gerçekleştirilir.

Belirsiz kurtarma birimleri RRS ' den nasıl çözülür?

RRS bağdaştırıcısının işlevlerinden biri, IBM MQ ile diğer RRS katılan kaynak yöneticileri arasında verileri eşitlenmiş olarak tutmaktır. IBM MQ kesinleştirmenin birinci aşamasını tamamladığında ve RRS ' den (kesinleştirme eşgüdümcüsü) bir karar beklenirken bir hata oluşursa, kurtarma birimi belirsiz duruma girer.

RRS ile IBM MQarasında iletişim yeniden kurulduğunda, RRS, kesinleştirmenin başlangıcını gösteren bir günlük kaydı olup olmadığına bağlı olarak her bir kurtarma birimini otomatik olarak kesinleştirir ya da geri çevirir. IBM MQ , RRS bağlantısı yeniden kuruluncaya kadar bu belirsiz kurtarma birimlerini çözemez (yani, IBM MQ kaynaklarında yapılan değişiklikleri kesinleştiremez ya da geri çekemez).

Bazı durumlarda, RRS belirsiz kurtarma birimlerini çözemez. Bu durumda IBM MQ , z/OS konsoluna aşağıdaki iletilerden birini gönderir:

- CSQ3011I
- CSQ3013I
- CSQ3014I
- CSQ3016I

Bu iletilerin ne anlama olduğu hakkında ayrıntılı bilgi 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 karşılık gelen kilitleri serbest bırakır. Çözümlenmemiş kurtarma birimleri yeniden başlatmadan sonra kalabilir. Bunları IBM MQ for z/OS Yönetimiiçinde açıklanan yöntemle çözün.

GROUP kurtarma birimi içeren belirsiz kurtarma birimleri nasıl çözülür?

GROUP kurtarma birimi olan belirsiz hareketler, GROUPUR kuyruk yöneticisi özniteliğinin etkinleştirildiği kuyruk paylaşım grubundaki (QSG) herhangi bir kuyruk yöneticisi tarafından hareket eşgüdümcüsü tarafından çözülebilir. Bir hareket koordinatörü yeniden bağlandığında, genellikle bekleyen belirsiz hareketlerin bir listesini ister ve bunları günlüklerindeki bilgileri kullanarak çözer.

Bir grup kurtarma yok etme birimiyle bağlantısı olan bir hareket eşgüdümcüsü belirsiz hareketlerin listesini istediğinde, döndürülen liste, kuyruk paylaşım grubunda var olan bir GROUP kurtarma birimi olan tüm belirsiz hareketlerden oluşur. Bu liste, belirsiz hareketlerin başlatıldığı kuyruk yöneticisine bağımlı değildir. Böyle bir isteği işleyen bir kuyruk yöneticisi, SYSTEM.QSG.UR.RESOLUTION.QUEUE. Kuyruk yöneticisi daha sonra, etkin olsalardı raporlayacakları ek belirsiz hareketleri saptamak için, etkin olmayan kuyruk yöneticilerinin günlüklerini son denetim noktasından okur.

Bir hareket eşgüdümcüsü belirsiz bir hareketin çözülmesini istediğinde, bağlı olduğu kuyruk yöneticisi, hareketin kendisinin kaynaklanıp kaynaklanmadığını ve çözülebilirse, QMGR kurtarma birimindeki hareketlerle aynı şekilde çözülür. Hareket QSG ' deki başka bir etkin kuyruk yöneticisinden kaynaklandıysa, çözümü tamamlama isteği SYSTEM.QSG.UR.RESOLUTION.QUEUE. Hareketin QSG 'deki etkin olmayan bir kuyruk yöneticisinde kaynaklandığı durumlarda, paylaşılan kuyruk işleri hemen çözülür ve SYSTEM.QSG.UR.RESOLUTION.QUEUE. Etkin olmayan kuyruk yöneticisi, yeni işi kabul etmeden önce başlatma işleminden sonra bu isteği işler. Bu senaryoda, özgün kuyruk yöneticisinin günlükleri, kurtarma birimi yeniden başlatılıp istek işleninceye kadar emin olmadığını gösterir.

z/OS Paylaşılan kuyruk kurtarma

Kuyruk paylaşım grubu ortamındaki çeşitli bileşenlerin IBM MQ kurtarmasını ve dayanıklılığını anlamak için bu konuyu kullanın.

- "İşlemsel kurtarma" sayfa 252
- "Eşdüzey kurtarma" sayfa 252
- "Paylaşılan kuyruk tanımlamaları" sayfa 253
- "Günlük Kaydı" sayfa 253
- "Bağlaşım olanağı ve yapı arızaları" sayfa 253
- "Yapı hatası senaryoları" sayfa 254
- "Bağlaşım olanağı bağlanırlığı arızalarına dayanıklılık" sayfa 255
- "Bağlaşım olanağı bağlanırlığı hatalarına Dayanıklılığın Yönetilmesi" sayfa 256
- "Operasyonel davranış" sayfa 258

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 oluştursa 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, sonlandırılmadan yönetim yapısına bağlanırlık kaybını tolere edebilir.

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 Sistem Komutları Başvurusuiçinde 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/OS içindeki güvenlik kavramları

IBM MQiçin güvenliğin önemini ve sisteminizde yeterli güvenlik ayarlarına sahip olmamanın etkilerini anlamak için bu konuyu kullanın.

IBM MQ kaynaklarını neden korumalısınız?

IBM MQ , değerli olabilecek bilgilerin aktarımını işler. Güvenliğin uygulanması, IBM MQ ' in sahip olduğu ve yönettiği kaynakların yetkisiz erişimden korunmasını sağlar. Bu tür bir erişim, bilgilerin kaybolmasına ya da açıklanmasına neden olabilir.

Aşağıdaki kaynaklara yetkisiz bir kullanıcı ya da işlem tarafından erişilmediğinden ya da değiştirilmediğinden emin olmanız gerekir:

- IBM MQ bağlantısı
- Kuyruklar, süreçler ve ad listesi gibi IBM MQ nesneleri
- IBM MQ iletim bağlantıları, yani IBM MQ kanalları

- IBM MQ sistem denetimi komutları
- IBM MQ ileti
- İletilerle ilişkili bağlam bilgileri

IBM MQ , gerekli güvenliği sağlamak üzere yetkilendirme isteklerini bir Dış Güvenlik Yöneticisi 'ne (ESM) yönlendirmek için z/OS sistem yetkilendirme olanağını (SAF) kullanır; örneğin, Güvenlik Sunucusu (önceki adı RACF). IBM MQ kendi güvenlik doğrulamasını yapmaz. Dağıtılmış kuyruğa alma ya da istemciler kullanıldığında, IBM MQ 'in kanal kimlik doğrulama kayıtlarını, kanal çıkışlarını, MCAUSER kanal özniteliğini ve TLS' yi sağladığı ek güvenlik önlemlerine gereksinim duyabilirsiniz.

Bir nesneye erişilmesine izin verme kararı ESM tarafından verilmiştir ve IBM MQ bu kararı izler. ESM bir karar veremezse, IBM MQ nesneye erişimi önler.

IBM MQ kaynaklarını korumazsanız ne olur?

Güvenlikle ilgili bir şey yapmazsanız, en olası etki *tüm* kullanıcılarının *her* kaynağa erişip bunları değiştirebilmeleri olabilir. Bu, yalnızca yerel kullanıcıları değil, dağıtımlı kuyruğa alma ya da istemcileri kullanan uzak sistemlerde olanları da içerir; burada oturum açma güvenliği denetimleri, z/OSiçin olağan durumda olduğundan daha az sıkı olabilir.

Güvenlik denetimini etkinleştirmek için aşağıdakileri yapmalısınız:

- Bir ESM takın ve etkinleştirin (örneğin, Güvenlik Sunucusu).
- Güvenlik Sunucusu dışında bir ESM kullanıyorsanız MQADMIN sınıfını tanımlayın.
- MQADMIN sınıfını etkinleştirin.

Büyük ve küçük harf karışık kaynak adları kullanmanın kuruluşunuz için yararlı olup olmayacağını göz önünde bulundurmanız gerekir. ESM profillerinizde büyük ve küçük harf karışık kaynak adları kullanıyorsanız, MXADMIN sınıfını tanımlamanız ve etkinleştirmeniz gerekir.

z/OS Veri Kümesi Şifrelemesi

DSE (Data Set Encryption; Veri Kümesi Şifrelemesi), z/OS veri kümelerini şifreleme yeteneği sağlar; böylece, içerdiği veriler yalnızca belirli bir izin verilen kullanıcı kimlikleri tarafından görüntülenebilir ya da değiştirilebilir. Bu, dosya sisteminde atıl durumdaki verilerin şifrelenmesini sağlar ve geçerli bir iş gereksinimi ve veri kümelerini yönetmek için izinlere sahip olan kullanıcılara hassas bilgilerin istemeden ifşa edilmesini önler.

IBM MQ for z/OS 9.1.4öncesinde IBM MQ for z/OS , IBM MQ iletileri için birincil kalıcılık mekanizmaları sağlayan etkin günlükler, sayfa kümeleri ve paylaşılan ileti veri kümeleriyle (SMDS) DSE kullanımını desteklemez.

Bunun yerine Advanced Message Security , IBM MQ ileti sistemi için uçtan uca bir şifreleme çözümü sağlar; bu çözüm, tüm IBM MQ ağını, hareket halindeki verilerin şifrelenmesini, atıl durumunu ve hatta çalıştırma zamanı IBM MQ süreçlerini kapsar.

IBM MQ altsisteminde kullanılan diğer VSAM ve sıralı veri kümeleri DSE kullanılarak şifrelenebilir. Örneğin:

- Bootstrap veri kümesi (BSDS)
- CSQINPx DDNAMEs kullanılarak başlatma sırasında okunan MQSC komutlarını bulunduran sıralı dosyalar
- IBM MQ arşiv günlükleri, genellikle denetleme amacıyla IBM MQ günlük verilerinin uzun vadeli arşivlenmesi için kullanılır.

Bir veri kümesi anahtar etiketiyle tanımlanan bir veri sınıfını ayırarak DSE kullanarak şifreleyebilirsiniz. Daha fazla bilgi için Günlük arşivi depolamanın planlanmasıbaşlıklı konuya bakın.

IBM MQ for z/OS 9.1.4'den IBM MQ for z/OS , önceki yayınlarda sağlanan desteğe ek olarak etkin günlükler ve sayfa kümeleriyle birlikte DSE' nin kullanılmasını destekler.

IBM MQ for z/OS , paylaşılan ileti veri kümeleri (SMDS) için DSE kullanımını desteklemez.

veri kümesi şifrelemesiyle IBM MQ for z/OS üzerinde atıl durumdaki veriler için gizlilik. başlıklı bölüme bakın. ek bilgi için.

İlgili kavramlar

Güvenlik kavramları

Kanal kimlik doğrulama kayıtları

z/OS üzerindeki IBM MQ nesneleriyle çalışma yetkisi

Şifreleme güvenliği iletişim kuralları: TLS

İlgili görevler

z/OS üzerinde güvenliğin ayarlanması

Bağlantı düzeyinde güvenlik ve uygulama düzeyinde güvenlik karşılaştırması

İlgili başvurular

IBM MQ for z/OS ile ilgili iletiler

z/OS IBM MQ for z/OS içindeki güvenlik denetimleri ve seçenekleri

Tüm IBM MQ altsistemi için güvenliğin açık olup olmadığını ve kuyruk yöneticisinde ya da kuyruk paylaşım grubu düzeyinde güvenlik denetimleri gerçekleştirmek isteyip istemediğinizi belirtebilirsiniz. API kaynak güvenliği için denetlenen kullanıcı kimliklerinin sayısını da denetleyebilirsiniz.

Altsistem güvenliği

Altsistem güvenliği, tüm kuyruk yöneticisi için güvenlik denetiminin yapılıp yapılmadığını belirten bir denetimdir. Güvenlik denetimine gerek duymuyorsanız (örneğin, bir test sisteminde) ya da IBM MQ ' e (istemciler ve kanallar dahil) bağlanabilen tüm kaynaklarda güvenlik düzeyinden memnunsanız, daha fazla güvenlik denetimi yapılmaması için kuyruk yöneticisi ya da kuyruk paylaşım grubu için güvenlik denetimini kapatabilirsiniz.

Bu, güvenliği tamamen kapatabilen ve diğer güvenlik denetimlerinin gerçekleştirilip gerçekleştirilmediğini belirleyebilen tek denetimdir. Diğer bir deyişle, kuyruk yöneticisi ya da kuyruk paylaşım grubu için denetimi kapatırsanız, başka bir IBM MQ denetimi gerçekleştirilmez; açık bırakırsanız, IBM MQ diğer kaynaklar için güvenlik gereksinimlerinizi denetler IBM MQ .

Komutlar gibi belirli kaynak kümeleri için güvenliği açabilir ya da kapatabilirsiniz.

Kuyruk yöneticisi ya da kuyruk paylaşımı grup düzeyi denetimi

Kuyruk yöneticisi düzeyinde ya da kuyruk paylaşım grubu düzeyinde güvenlik uygulayabilirsiniz. Kuyruk paylaşım grubu düzeyinde güvenlik uygularsanız, gruptaki tüm kuyruk yöneticileri aynı profilleri paylaşır. Bu, tanımlanacak ve korunacak daha az profil olduğu anlamına gelir ve güvenlik yönetimini kolaylaştırır. Var olan güvenlik profillerini devraldığı için kuyruk paylaşım grubuna yeni bir kuyruk yöneticisi eklenmesini de kolaylaştırır.

Kuruluşunuz gerektiriyorsa (örneğin, geçiş sırasında ya da kuyruk paylaşım grubunda, gruptaki diğer kuyruk yöneticilerine farklı güvenlik düzeyleri gerektiren bir kuyruk yöneticiniz varsa), her ikisinin birleşimini de uygulayabilirsiniz.

Kuyruk paylaşımı grup düzeyi güvenliği

Kuyruk paylaşım grubu düzeyinde güvenlik denetimi, tüm kuyruk paylaşım grubu için gerçekleştirilir. Daha az güvenlik profili tanımlamanızı gerektirdiğinden güvenlik yönetimini basitleştirmenizi sağlar. Bir kullanıcı kimliğinin belirli bir kaynağı kullanma yetkisi, kuyruk paylaşım grubu düzeyinde işlenir ve kullanıcı kimliğinin kaynağa erişmek için kullandığı kuyruk yöneticisinden bağımsızdır.

Örneğin, bir sunucu uygulamasının SERVER kullanıcı kimliği altında çalıştığını ve SERVER.REQUESTve sistem birleşimi (sysplex) içindeki her z/OS görüntüsünde bir SERVER eşgörünümü çalıştırmak

istiyorsunuz. SERVER sunucusunun SERVER.REQUEST (kuyruk yöneticisi düzeyinde güvenlik), 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 düzeyinde güvenlik profillerini kullanabilirsiniz.

Her iki düzeyin birleşimi
Hem kuyruk yöneticisi hem de kuyruk paylaşım grubu düzeyi güvenliğinin bir birleşimini kullanabilirsiniz.

Bu grubun üyesi olan belirli bir kuyruk yöneticisi için kuyruk paylaşımı grubu düzeyinde güvenlik ayarlarını geçersiz kılabilirsiniz. Bu, tek bir kuyruk yöneticisinde, gruptaki diğer kuyruk yöneticilerine farklı bir güvenlik denetimi düzeyi gerçekleştirebileceğiniz anlamına gelir.

Daha fazla bilgi için Kuyruk paylaşım grubunu ya da kuyruk yöneticisi düzeyinde güvenliği denetleyen profillerbaşlıklı konuya bakın.

Denetlenen kullanıcı kimliği sayısı denetleniyor

RESLEVEL, IBM MQ kaynak güvenliği için denetlenen kullanıcı kimliği sayısını denetleyen bir Güvenlik Sunucusu profilidir. Olağan durumda, bir kullanıcı bir IBM MQ kaynağına erişmeye çalıştığında, Güvenlik Sunucusu ilgili kullanıcı kimliğini ya da kimliklerini o kaynağa erişime izin verilip verilmediğini görmek için denetler. Bir RESLEVEL tanıtımı tanımlayarak, sıfır, bir ya da varsa, iki kullanıcı kimliğinin denetlenip denetlenmeyeceğini denetleyebilirsiniz.

Bu denetimler, bağlantı temelinde yapılır ve bağlantının ömrü boyunca devam eder.

Her kuyruk yöneticisi için tek bir RESLEVEL tanıtımı vardır. Denetim, bir kullanıcı kimliğinin bu tanıtım için sahip olduğu erişim tarafından gerçekleştirilir.

Büyük ve küçük harf karışık IBM MQ RACF sınıfları

Büyük ve küçük harf karışık kaynak adlarını kullanmanıza ve bunları korumak için IBM MQ RACF profillerini tanımlamanıza olanak sağlayan karışık büyük ve küçük harf RACF profil desteğini kullanabilirsiniz.

Aşağıdakilerden birini seçebilirsiniz:

- Önceki yayınlarda olduğu gibi yalnızca büyük harf IBM MQ RACF sınıflarını kullanmaya devam edin ya da
- Yeni büyük ve küçük harf karma IBM MQ RACF sınıflarını kullanın.

Büyük ve küçük harf karışık RACF profilleri kullanılmadan, IBM MQ for z/OS içinde büyük ve küçük harf karışık kaynak adlarını kullanmaya devam edebilirsiniz; Ancak, bu kaynak adları yalnızca büyük harfli IBM MQ sınıflarındaki soysal RACF tanıtımlarıyla korunabilir. Büyük ve küçük harf karışık IBM MQ RACF profil desteğini kullanırken, büyük ve küçük harf karışık IBM MQ sınıflarında IBM MQ RACF profilleri tanımlayarak daha ayrıntılı bir koruma düzeyi sağlayabilirsiniz.

z/OS IBM MQ for z/OS içinde koruyabileceğiniz kaynaklar

Bir kuyruk yöneticisi başlatıldığında ya da bir işletmen komutu tarafından istendiğinde, IBM MQ for z/OS hangi kaynakları korumak istediğinizi belirler.

Her bir kuyruk yöneticisi için hangi güvenlik denetimlerinin gerçekleştirileceğini denetleyebilirsiniz. Örneğin, bir üretim kuyruğu yöneticisine bir dizi güvenlik denetimi uygulayabilirsiniz, ancak bir test kuyruğu yöneticisinde bu denetim gerçekleştirilmez.

Bağlantı güvenliği

Uygulama programı bir kuyruk yöneticisine bağlanmayı denediğinde bağlantı güvenliği denetimi gerçekleştirilir. Bir MQCONN ya da MQCONNX isteği ya da kanal başlatıcısı ya da CICS ya da IMS bağdaştırıcısı bir bağlantı isteği yayınladığında gerçekleştirilir.

Kuyruk yöneticisi düzeyinde güvenlik kullanıyorsanız, belirli bir kuyruk yöneticisi için bağlantı güvenliği denetimini kapatabilirsiniz. Ancak, bunu yaparsanız, herhangi bir kullanıcı o kuyruk yöneticisine bağlanabilir.

CICS bağdaştırıcısı için, tek tek CICS uçbirim kullanıcı kimliği değil, bağlantı güvenliği denetimi için yalnızca CICS adres alanı kullanıcı kimliği kullanılır. IMS bağdaştırıcısı için, IMS denetim ya da bağımlı bölgeleri 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, IBM MQ for z/OSüzerinde MQSC ve PCF komutlarını yayınlayabileceğiniz kaynaklardan herhangi birinden bir MQSC komutu verdiğinde komut güvenliği denetimi gerçekleştirilir. “Komut kaynağı güvenliği” sayfa 262başlıklı konuda açıklandığı gibi, komutun belirttiği kaynakta ayrı bir denetim yapabilirsiniz.

Komut denetimini kapatırsanız, komut verenler komutu verme yetkisine sahip olup olmadıklarını görmek için denetlenmez.

MQSC komutları bir konsoldan girilirse, konsolun z/OS SYS konsolu yetki özniteliği olmalıdır. CSQINP1 ya da CSQINP2 veri kümelerinden ya da kuyruk yöneticisi tarafından içeride yayınlanan komutlar, tüm güvenlik denetimlerinden muaf tutulurken, CSQINPX kullanıcıları kanal başlatıcı adres alanının kullanıcı kimliğini kullanır. Bu veri kümelerini normal veri kümesi korumasıyla kimin güncellemesine izin verilmesini denetlemeniz gerekir.

Komut güvenliği denetimini, kuyruk yöneticisi ya da kuyruk paylaşım grubu düzeyinde açabilir ya da kapatabilirsiniz.

Komut kaynağı güvenliği

Yerel kuyruk tanımlama gibi bazı MQSC komutları, IBM MQ kaynaklarının işlenmesini içerir. Komut kaynak güvenliği etkin olduğunda, bir kaynağı içeren bir komut her yayınlandığında IBM MQ , kullanıcının o kaynağın tanımlamasını değiştirmesine izin verilip verilmediğini denetler.

Adlandırma standartlarının uygulanmasına yardımcı olmak için komut kaynağı güvenliğini kullanabilirsiniz. Örneğin, bir bordro yöneticisinin yalnızca adları "BORDRO" ile başlayan kuyrukları silmesine ve tanımlamasına izin verilebilir. Komut kaynak güvenliği etkin değilse, komut tarafından işlenmekte olan kaynakta güvenlik denetimi yapılmaz. Komut kaynağı güvenliğini komut güvenliğiyle karıştırmayın; bu ikisi bağımsızdır.

Komut kaynak güvenliği denetiminin kapatılması, özellikle komutlar içermeyen diğer işleme tipleri için yapılan kaynak denetimini etkilemez.

Komut kaynağı güvenliği denetimini, kuyruk yöneticisi ya da kuyruk paylaşım grubu düzeyinde açabilir ya da kapatabilirsiniz.

Kanal güvenliğiyle ilgili önemli noktalar

Kanal güvenliği

Kanalları kullanırken, kullanılabilir güvenlik özellikleri, kullanacağınız iletişim protokolüne bağlıdır. TCP kullanırsanız, iletişim protokolüyle birlikte sağlanan güvenlik özellikleri yoktur, ancak TLS ' yi

kullanabilirsiniz. APPC kullanıyorsanız, kullanıcı kimliği bilgilerini ağ üzerinden gönderen MCA 'dan doğrulama için hedef MCA' ya gönderebilirsiniz.

Her iki protokol için de, güvenlik amacıyla hangi kullanıcı kimliklerini denetlemek istediğinizi ve kaç tane kullanıcı kimliği istediğinizi belirtebilirsiniz. Yine, kullanabileceğiniz seçenekler, hangi protokolü kullandığınıza, kanalı tanımlarken belirteceğiniz ayarlara ve kanal başlatıcısına ilişkin RESLEVEL ayarlarına bağlıdır.

Kullanılabilir kanal güvenliği tipleri hakkında daha fazla bilgi için bkz. Kanal kimlik doğrulama kayıtları ve Güvenlik çıkışına genel bakış

İlgili başvurular

"IBM MQ for z/OS içinde API-kaynak güvenliği" sayfa 263
Bir uygulama MQOPEN ya da MQPUT1 çağrısıyla bir nesneyi açtığında kaynaklar denetlenir. Bir nesneyi açmak için gereken erişim, kuyruk açıldığında belirlenen açma seçeneklerine bağlıdır.

z/OS *IBM MQ for z/OS içinde API-kaynak güvenliği*

Bir uygulama MQOPEN ya da MQPUT1 çağrısıyla bir nesneyi açtığında kaynaklar denetlenir. Bir nesneyi açmak için gereken erişim, kuyruk açıldığında belirlenen açma seçeneklerine bağlıdır.

API-kaynak güvenliği aşağıdaki denetimlere bölünür:

- Kuyruk
- Süreç
- Namelist
- Diğer kullanıcı
- Bağlam

Kuyruk yöneticisi nesnesi açılırken ya da depolama sınıfı nesnelerine erişilirken güvenlik denetimi gerçekleştirilmez.

Kuyruk

Kuyruk güvenliği denetimi, kimin hangi kuyruğu açmasına izin verildiğini ve bu kuyruğu hangi seçeneklerle açmalarına izin verildiğini denetler. Örneğin, bir kullanıcının PAYROLL.INCREASE.SALARY (MQOO_BROWSE seçeneğini kullanarak) kuyruktaki iletilere göz atmak, ancak kuyruktan iletileri kaldırmak (MQOO_INPUT_ * seçeneklerinden birini kullanarak) için. Kuyruk denetimini kapatırsanız, herhangi bir kullanıcı MQOPEN ya da MQPUT1 çağrısındaki herhangi bir geçerli MQOO_ * seçeneğiyle herhangi bir kuyruğu açabilir.

Kuyruk yöneticisi ya da kuyruk paylaşım grubu düzeyinde kuyruk güvenliği denetimini açabilir ya da kapatabilirsiniz.

Süreç

Kullanıcı bir süreç tanımlaması nesnesini açtığında süreç güvenliği denetimi gerçekleştirilir. İşlem denetimini kapatırsanız, herhangi bir kullanıcı herhangi bir işlemi açabilir.

Kuyruk yöneticisi ya da kuyruk paylaşım grubu düzeyinde işlem güvenliği denetimini açabilir ya da kapatabilirsiniz.

Ad listesi

Namelist güvenlik denetimi, kullanıcı bir ad listesi açtığında gerçekleştirilir. Ad listesi denetimini kapatırsanız, 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 kimliğinin yetkisini kullanıp kullanamayacağını denetler.

Örneğin:

- PAYSERV kullanıcı kimliği altında çalışan bir sunucu programı, USER1kullanıcı kimliği tarafından kuyruğa konan bir kuyruktan bir istek iletisi alır.
- Sunucu programı istek iletisini aldığında, isteği işler ve yanıtı istek iletisiyle belirlenen yanıt kuyruğuna geri koyar.
- Sunucu, yanıt kuyruğunun açılmasını yetkilendirmek için kendi kullanıcı kimliğini (PAYSERV) kullanmak yerine, USER1adlı başka bir kullanıcı kimliği belirleyebilir. Bu örnekte, alternatif kullanıcı güvenliği, PAYSERV kullanıcı kimliğinin yanıt kuyruğunu açarken alternatif bir kullanıcı kimliği olarak USER1 kullanıcı kimliğini belirtmesine izin verilip verilmediğini denetler.

Alternatif kullanıcı kimliği, nesne tanımlayıcısının (MQOD) *AlternateUserId* alanında belirtilir.

Süreçler ya da ad listesi gibi herhangi bir IBM MQ nesnesinde alternatif kullanıcı kimliklerini kullanabilirsiniz. Bu, CICS güvenliği ya da z/OS veri kümesi güvenliği gibi diğer kaynak yöneticileri tarafından kullanılan kullanıcı kimliğini etkilemez.

Diğer kullanıcı güvenliği etkin değilse, diğer kullanıcı kimliği olarak başka bir kullanıcı kimliği kullanılabilir.

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 bir parçası olan ileti tanımlayıcısında (MQMD) bulunan bilgilerdir. Bağlam bilgileri iki kısımdan oluşur:

Kimlik bölümü
İletiyi ilk olarak kuyruğa koyan uygulamanın kullanıcısı. Aşağıdaki alanlardan oluşur:

- *UserIdentifier*
- *AccountingToken*
- *ApplIdentityData*

Kaynak bölümü
İletiyi şu anda saklandığı kuyruğa koyan uygulama. Aşağıdaki alanlardan oluşur:

- *PutApplType*
- *PutApplName*
- *PutDate*
- *PutTime*
- *ApplOriginData*

Uygulamalar, MQPUT ya da MQPUT1 çağrısı yapıldığında bağlam verilerini belirtebilir. Uygulama verileri oluşturabilir, veriler başka bir iletiden iletilebilir ya da kuyruk yöneticisi varsayılan olarak verileri oluşturabilir. Örneğin, sunucu programları, istekte bulunanın kimliğini denetlemek için bağlam verilerini kullanabilir; yani, bu ileti doğru uygulamadan mı geldi? Genellikle *UserIdentifier* alanı, alternatif bir kullanıcının kullanıcı kimliğini belirlemek için kullanılır.

Bağlam güvenliği, kullanıcının herhangi bir MQOPEN ya da MQPUT çağrısında bağlam seçeneklerinden herhangi birini belirtip belirtemeyeceğini denetlemek için kullanılır. Bağlam seçenekleriyle ilgili bilgi için İleti bağlamıyla ilgili MQOPEN seçeneklerinebakın. Bağlamla ilgili ileti tanımlayıcı alanlarının açıklamaları için bakınız: MQMD-Message descriptorMQMD -Message descriptor.

Bağlam güvenliği denetimini kapatırsanız, herhangi bir kullanıcı kuyruk güvenliğinin izin verdiği bağlam seçeneklerinden herhangi birini kullanabilir.

Kuyruk, kuyruk yöneticisi ya da kuyruk paylaşım grubu düzeyinde bağlam güvenliği denetimini açabilir ya da kapatabilirsiniz.

z/OS üzerinde kullanılabilirlik

IBM MQ for z/OS , yüksek kullanılabilirlik için birçok özelliğe sahiptir. Bu konuda, kullanılabilirlikle ilgili dikkat edilmesi gereken bazı noktalar açıklanmaktadır.

Kuyruk yöneticisi ya da kanal başlatıcısı başarısız olursa, IBM MQ ' in 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
- Kuyruk paylaşım grubunda kurtarma için z/OS GROUPUR özniteliğinin kullanılması
- Kullanılabilirlik hakkında daha fazla bilgi nereden edinilirsiniz

Sysplex ile ilgili önemli noktalar

Sysplex' te, bir dizi z/OS işletim sistemi görüntüsü tek bir sistem görüntüsünde işbirliği yapıyor ve bir bağlaşım olanağı kullanarak iletişim kuruyor. IBM MQ , gelişmiş kullanılabilirlik için sistem şebekesi ortamının olanaklarını kullanabilir.

Bir kuyruk yöneticisi ile belirli bir z/OS görüntüsü arasındaki yakınlıkların kaldırılması, bir görüntü arızası durumunda kuyruk yöneticisinin farklı bir z/OS görüntüsünde yeniden başlatılmasına olanak sağlar. Yeniden başlatma mekanizması el ile kullanılabilir, ARM ' yi kullanabilir ya da sistem otomasyonunu kullanabilir (aşağıdakilerden emin olun):

- Tüm sayfa kümeleri, günlükler, önyükleme veri kümeleri, kod kitaplıkları ve kuyruk yöneticisi yapılandırma veri kümeleri paylaşılan birimlerde tanımlanır.
- Altsistem tanımının sysplex kapsamı ve sysplex içinde benzersiz bir adı vardır.
- IPL zamanında her z/OS görüntüsünde kurulu *erken kod* düzeyi aynı düzeydedir.
- TCP sanal IP adresleri (VIPA), sistem şebekesinde bulunan her TCP yığınında bulunur ve IBM MQ TCP dinleyicilerini ve gelen bağlantılarını varsayılan anasistem adları yerine VIPA ' ları kullanacak şekilde yapılandırdınız.

Sysplex 'te TCP kullanımı hakkında daha fazla bilgi için bkz. *TCP/IP in a sysplex*, SG24-5235, bir IBM Redbooks yayını.

Ayrıca, bir sistem şebekesinde farklı işletim sistemi görüntülerinde çalışan birden çok kuyruk yöneticisini, daha yüksek kullanılabilirlik ve iş yükü dengelemesi için paylaşılan kuyruklardan ve paylaşılan kanallardan yararlanabilecek bir kuyruk paylaşım grubu olarak çalışacak şekilde yapılandırabilirsiniz.

Paylaşılan kuyruklar

Kuyruk paylaşım grubu ortamında, bir uygulama kuyruk paylaşım grubu içindeki herhangi bir kuyruk yöneticisine bağlanabilir. Kuyruk paylaşım grubundaki tüm kuyruk yöneticileri aynı paylaşılan kuyruk kümesine erişebildiğinden, uygulama belirli bir kuyruk yöneticisinin kullanılabilirliğine bağlı değildir; kuyruk paylaşım grubundaki herhangi bir kuyruk yöneticisi herhangi bir kuyruğa hizmet verebilir. Bu, kuyruk paylaşım grubundaki diğer tüm kuyruk yöneticileri kuyruğu işlemeye devam edebildiği için bir kuyruk yöneticisi durursa daha fazla kullanılabilirlik sağlar. Paylaşılan kuyrukların yüksek kullanılabilirliğine ilişkin bilgi için bkz. “Paylaşılan kuyrukları kullanmanın avantajları” sayfa 179.

Bir kuyruk paylaşım grubundaki iletilerin kullanılabilirliğini daha da artırmak için IBM MQ , gruptaki başka bir kuyruk yöneticisinin bağlaşım olanağıyla olağandışı bir şekilde bağlantı kurup kuramadığını saptar

ve bu kuyruk yöneticisine ilişkin iş birimlerini tamamlar (olanaklıysa). Bu, *eş kurtarma* olarak bilinir ve “Eşdüzey kurtarma” sayfa 252içinde açıklanır.

Eşdüzey kurtarma, hata sırasında şüphe içinde olan iş birimlerini kurtaramaz. Hataya dahil olan tüm sistemleri (CICS, Db2ve IBM MQ gibi) yeniden başlatmak ve bunların aynı yeni işlemcide yeniden başlatıldığından emin olmak için Otomatik Yeniden Başlatma Yöneticisi 'ni (ARM) kullanabilirsiniz. Bu, yeniden eşzamanlayabildikleri ve belirsiz iş birimlerinin hızlı bir şekilde kurtarılmasını sağladıkları anlamına gelir. Bu, “z/OS Automatic Restart Manager (ARM) olanağının kullanılması” sayfa 266içinde açıklanmıştır.

Paylaşılan kanallar

Kuyruk paylaşım grubu ortamında IBM MQ , ağa yüksek kullanılabilirlik sağlayan işlevler sağlar. Kanal başlatıcı, bir dizi uygun sunucu arasında ağ isteklerini dengeleyen ve sunucu hatalarını ağdan gizleyen ağ ürünleri kullanmanızı sağlar (örneğin, VTAM genel kaynakları). IBM MQ , gelen istekler için soysal bir kapı kullanır; böylece istekleri, kuyruk paylaşım grubundaki herhangi bir kullanılabilir kanal başlatıcısına yöneltebilir. Bu, “Paylaşılan kanallar” sayfa 199içinde açıklanmıştır.

Paylaşılan giden kanallar, paylaşılan bir iletim kuyruğundan gönderdikleri iletileri alır. Paylaşılan bir kanalın durumuna ilişkin bilgiler, tüm kuyruk paylaşım grubu düzeyi için tek bir yerde tutulur. Bu, kanal başlatıcısı, kuyruk yöneticisi ya da iletişim altsistemi başarısız olursa, bir kanalın kuyruk paylaşım grubundaki farklı bir kanal başlatıcısında otomatik olarak yeniden başlatılabileceği anlamına gelir. Buna *eş kanal kurtarma* denir ve Paylaşılan giden kanallar' da açıklanır.

IBM MQ ağ kullanılabilirliği

IBM MQ iletileri, kanallar kullanılarak bir IBM MQ ağında kuyruk yöneticisinden kuyruk yöneticisine taşınır. Bir kuyruk yöneticisinin ağ kullanılabilirliğini ve bir IBM MQ kanalının ağ sorununu saptama ve yeniden bağlanma yeteneğini geliştirmek için yapılandırmayı çeşitli düzeylerde değiştirebilirsiniz.

TCP *canlı tutma* TCP/IP kanalları için kullanılabilir. TCP ' nin ağ hatalarını saptamak için oturumlar arasında düzenli aralıklarla paket göndermesine neden olur. KAINT kanal özniteliği, bir kanal için bu paketlerin sıklığını belirler.

AdoptMCA , ağ kesintisi sonucu alma işleminde engellenen bir kanalın sonlandırılmasına ve yeni bir bağlantı isteğiyle değiştirilmesine olanak sağlar. AdoptMCA komutunu, MQSC yardımcı programıyla ADOPTMCA kuyruk yöneticisi özelliğini ya da Programmable Command Formats arabirimiyle AdoptNewMCAType özelliğini kullanarak denetleyebilirsiniz.

ReceiveTimeout , bir kanalın bir ağ alma çağrısında kalıcı olarak engellenmesini önler. RCVTIME ve RCVTMIN kanal başlatıcı parametreleri, kanallara ilişkin alma zamanaşımı özelliklerini sağlıklı işletim aralığının bir işlevi olarak belirler. Daha fazla ayrıntı için bkz. Kuyruk yöneticisi parametresi .

z/OS Automatic Restart Manager (ARM) olanağının kullanılması

IBM MQ for z/OS bağımsız değişkenini z/OS otomatik yeniden başlatma yöneticisi (ARM) ile birlikte kullanabilirsiniz. Bir kuyruk yöneticisi ya da kanal başlatıcısı başarısız olursa, ARM bunu aynı z/OS görüntüsünde yeniden başlatır. z/OS başarısız olursa, bir grup ilgili altsistem ve uygulama da başarısız olur. ARM, sistem şebekesi içindeki başka bir z/OS görüntüsünde, önceden tanımlanmış bir sırayla, tüm arızalı sistemleri otomatik olarak yeniden başlatabilir. Buna sistemler arası yeniden başlatma denir.

ARM, paylaşılan kuyruk ortamında belirsiz hareketlerin hızla kurtarılmasını sağlar. Ayrıca, kuyruk paylaşım gruplarını kullanmıyorsanız daha yüksek kullanılabilirlik sağlar.

z/OS hatası durumunda sysplex içindeki farklı bir z/OS görüntüsünde bir kuyruk yöneticisini yeniden başlatmak için ARM ' yi kullanabilirsiniz.

Otomatik yeniden başlatmayı etkinleştirmek için aşağıdakileri yapmalısınız:

1. Bir ARM bağlaşım veri kümesi ayarlayın.
2. z/OS ' in bir *ARM ilkesinde*gerçekleştirmesini istediğiniz otomatik yeniden başlatma işlemlerini tanımlayın.
3. ARM ilkesini başlatın.

Farklı z/OS görüntülerinde kuyruk yöneticilerini otomatik olarak yeniden başlatmak istiyorsanız, kuyruk yöneticisinin yeniden başlatılabileceği her z/OS görüntüsündeki her kuyruk yöneticisi, sistem birleşimi çapında benzersiz 4 karakterli altsistem adıyla tanımlanmalıdır.

ARM ' nin IBM MQ ile kullanılması IBM MQ ağında ARM kullanılmasıbaşlıklı konuda açıklanmaktadır.

z/OS Extended Recovery Facility (XRF) olanağının kullanılması

IBM MQ olanağını 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 üzerinde olmalıdır.

Kurtarma için XRF kullanıyorsanız, etkin işlemcide kuyruk yöneticisini durdurmanız ve diğer işlemcide başlatmanız gerekir. CICSiçin, bunu CICStarafından sağlanan komut listesi çizelgesini (CLT) kullanarak yapabilirsiniz ya da sistem işletmeni bunu el ile yapabilir. IMSiçin bu, el ile gerçekleştirilen bir işlemdir ve bunu, eşgüdüm IMS sistemi işlemci anahtarını tamamladıktan sonra yapmanız gerekir.

Kuyruk yöneticisinin diğer işlemciye geçebilmesi için IBM MQ yardımcı programları tamamlanmalı ya da sona erdirilmelidir. XRF kurtarma planlarınızı planlarken bu olası kesintiyi dikkatle göz önünde bulundurun.

Etkin işlemcideki kuyruk yöneticisi sona ermeden önce, kuyruk yöneticisinin diğer işlemcide başlatılmasını önleyin. Erken başlatma, verilerde, katalogda ve günlükte ciddi bütünlük sorunlarına neden olabilir. GRS 'nin (Global Resource serialization; Genel Kaynak Diziselleştirme) kullanılması, iki sistemde IBM MQ ' in eşzamanlı kullanımını önleyerek bütünlük sorunlarının önlenmesine yardımcı olur. BSDS ' yi korumalı bir kaynak olarak eklemeniz ve GRS halkasında etkin ve alternatif XRF işlemcilerini eklemeniz 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 üyesinde gerçekleştirilen, gerekli olabilecek herhangi bir belirsiz hareket kurtarmasına sahip olmasını sağlar.

Bir XA istemci uygulaması bir Sysplex aracılığıyla bir kuyruk paylaşım grubuna (QSG) bağlanırsa, hangi kuyruk yöneticisine bağlanacağını garanti edemez. Kuyruk paylaşım grubu içindeki kuyruk yöneticileri tarafından GROUPUR özniteliğinin kullanılması, QSG ' nin herhangi bir üyesinde oluşması gerekebilecek herhangi bir belirsiz hareket kurtarmasını etkinleştirebilir. Uygulamanın başlangıçta bağlandığı kuyruk yöneticisi kullanılamasa da, hareket kurtarma işlemi gerçekleştirilebilir.

Bu özellik, XA istemci uygulamasını QSG ' nin belirli üyelerine bağımlılıktan kurtararak kuyruk yöneticisinin kullanılabilirliğini artırır. Kuyruk paylaşım grubu, işlemsel uygulamaya tüm IBM MQ özelliklerini sağlayan ve tek bir kuyruk yöneticisi hata noktası olmadan tek bir varlık olarak görünür.

Bu işlev, işlemsel uygulama tarafından anlaşılmıyor.

Kullanılabilirlik hakkında daha fazla bilgi nereden bulunacak?

Bu konularla ilgili daha fazla bilgiyi aşağıdaki kaynaklardan bulabilirsiniz:

Çizelge 24. Kullanılabilirlik hakkında daha fazla bilgi nereden bulunacak?	
Konu	**Nereye bakıyorsunuz?**
Kuyruk paylaşım grupları	“Paylaşılan kuyruklar ve kuyruk paylaşım grupları” sayfa 160
Sistem değiştirbeleri	Sistem değiştirgelerinin yapılandırılması
Otomatik Yeniden Başlatma Yöneticisinin Kullanılması Yardımcı programlar	IBM MQ ağında ARM kullanılması
MQSC komutları	MQSC komutları

z/OS IBM MQ for z/OS üzerinde izleme ve istatistik

IBM MQ for z/OS ' in kuyruk yöneticisini izlemek ve istatistikleri toplamak için bir olanaklar kümesi vardır.

IBM MQ , sistemin izlenmesi ve istatistiklerin toplanması için olanaklar sağlar. Bu olanaklarla ilgili daha fazla bilgi için aşağıdaki bölümlere bakın:

- “Çevrimiçi izleme” sayfa 268
- “IBM MQ İz” sayfa 268
- “<” sayfa 269

Çevrimiçi izleme

IBM MQ , IBM MQ nesnelerinin durumunu izlemek için aşağıdaki komutları içerir:

- DISPLAY CHSTATUS, belirtilen bir kanalın durumunu görüntüler.
- DISPLAY QSTATUS, belirlenen 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 izlemesi, başarımı izlemenize ve sisteminizi ayarlamanıza yardımcı olmak için aşağıdaki bilgileri toplar:

- Farklı MQI isteklerinin sayısı (ileti yöneticisi istatistikleri)
- Farklı nesne isteklerinin sayısı (veri yöneticisi istatistikleri)
- Db2 kullanımıyla ilgili bilgiler (Db2 yönetici istatistikleri)
- Kupon Olanağı kullanımına ilişkin bilgiler (Coupling Facility yöneticisi istatistikleri)
- SMDS kullanımıyla ilgili bilgiler (paylaşılan ileti veri kümesi istatistikleri)
- Arabellek havuzu kullanımıyla ilgili bilgiler (arabellek yöneticisi istatistikleri)
- Günlüğe kaydetmeyle ilgili bilgiler (günlük yöneticisi istatistikleri)
- Depolama kullanımıyla ilgili bilgiler (depolama yöneticisi istatistikleri)
- Kilit istekleriyle ilgili bilgi (kilit yöneticisi istatistikleri)

Muhasebe verileri

- Muhasebe izlemesi, MQI çağrılarının işlenmesi için harcanan işlemci süresi ve belirli bir kullanıcı tarafından yapılan MQPUT ve MQGET isteklerinin sayısı hakkında bilgi toplar.
- IBM MQ , IBM MQkullanarak her bir görevle ilgili bilgi de toplayabilir. Bu veriler, iş parçacığı düzeyinde bir muhasebe kaydı olarak toplanır. Her iş parçacığı için IBM MQ , o iş parçacığı tarafından kullanılan her bir kuyruğa ilişkin bilgileri de toplar.

İzleme tarafından oluşturulan veriler, Sistem Yönetimi Olanağı 'na (SMF) ya da genelleştirilmiş izleme olanağına (GTF) gönderilir.

<

IBM MQ olayları, bir kuyruk yöneticisindeki hatalar, uyarılar ve diğer önemli olaylar hakkında bilgi sağlar. Bu olayları kendi sistem yönetimi uygulamanızla birleştirerek, birden çok IBM MQ uygulaması için birçok kuyruk yöneticisindeki etkinlikleri izleyebilirsiniz. Özellikle, sisteminizdeki tüm kuyruk yöneticilerini tek bir kuyruk yöneticisinden izleyebilirsiniz.

Olaylar, kullanıcı tarafından yazılan bir raporlama mekanizması aracılığıyla, olayların bir işletmene sunulmasını destekleyen bir yönetim uygulamasına bildirilir. Olaylar ayrıca, NetViewgibi diğer yönetim ağları için aracı olarak hareket eden uygulamaların raporları izlemesini ve uygun uyarıları oluşturmasını da sağlar.

İlgili görevler

IBM MQ izlemesini kullanma

IBM MQ olaylarını kullanma

z/OS z/OS üzerinde kurtarma birimi yok etme

Bazı işlemsel uygulamalar, kuyruk paylaşım grubundaki (QSG) bir kuyruk yöneticisine bağlandığında, kuyruk yöneticisi adı yerine QSG adını belirterek, QMGR yerine bir kurtarma atma birimi kullanabilir. Bu, QSG ' de aynı kuyruk yöneticisine yeniden bağlanma gereksinimini kaldırarak işlem kurtarmanın daha esnek ve güçlü olmasını sağlar.

Kuyruk paylaşım grubu adı kullanılarak bağlanan uygulamaların başlattığı hareketlerde de bir GROUP kurtarma birimi bulunur.

Bir hareket uygulaması bir GROUP kurtarma birimi ile bağlantı kurduğunda, mantıksal olarak kuyruk paylaşım grubuna bağlanır ve belirli bir kuyruk yöneticisine benzeşimi yoktur. QSG içindeki herhangi bir kuyruk yöneticisine bağlanıldığında, kesinleştirme işleminin phase-1 aşamasını tamamlamış olan, iki aşamalı kesinleştirme hareketleri sorulabilir ve çözülebilir. Bir kurtarma senaryosunda bu, hareket eşgüdümleyicinin aynı kuyruk yöneticisine yeniden bağlanmasına gerek olmadığı anlamına gelir; bu durum, o sırada kullanılamayabilir.

QMGR kurtarma birimi ile bağlantı kuran uygulamaların, bağlı oldukları kuyruk yöneticisine doğrudan benzerliği vardır. Bir kurtarma senaryosunda, hareket eşgüdümcüsü, kuyruk yöneticisinin bir kuyruk paylaşım grubuna ait olup olmadığına bakılmaksızın, belirsiz hareketleri çözmek için aynı kuyruk yöneticisine yeniden bağlanmalıdır.

Uygulamalar bir kuyruk paylaşım grubu adı belirttiğinde ve bu nedenle, grup kurtarma birimiyle QSG ' deki bir kuyruk yöneticisine bağlandığında, kuyruk paylaşım grubu mantıksal olarak ayrı bir kaynak yöneticisidir. Bu, belirsiz hareketlerin yalnızca aynı kurtarma birimiyle yeniden bağlantı kurması durumunda uygulama tarafından görülebileceği anlamına gelir. QMGR kurtarma birimi yok etme özelliğine sahip belirsiz hareketler, bir GROUP kurtarma birimi ile bağlı olan uygulamalar tarafından görülmez; tersi de geçerlidir.

İlgili kavramlar

“GROUP kurtarma birimlerinin etkinleştirilmesi” sayfa 270

Bir kuyruk paylaşım grubu, grup kurtarma birimlerini yapılandırabilir ve bu desteği etkinleştirebilir.

“Desteklenen Uygulamalar” sayfa 270
Hangi uygulamaların bir GROUP kurtarma birimiyle bağlanabileceğini saptamak için bu sayfayı kullanın.

z/OS GROUP kurtarma birimlerinin etkinleştirilmesi

Bir kuyruk paylaşım grubu, grup kurtarma birimlerini yapılandırabilir ve bu desteği etkinleştirebilir.

QSG içindeki bir kuyruk yöneticisinde GROUP kurtarma birimlerini kullanmak için GROUPUR kuyruk yöneticisi özniteliğini etkinleştirin. Bu kavramla ilgili daha fazla bilgi için bu konunun geri kalanını okumadan önce bkz. “z/OS üzerinde kurtarma birimi yok etme” sayfa 269 .

GROUPUR kuyruk yöneticisi özniteliği etkinleştirildiğinde, kuyruk yöneticisi GROUP kurtarma birimi içeren yeni bağlantıları kabul eder. Bu özniteliği geçersiz kılarsanız, bu yok etme ile yeni bağlantılar kabul edilmez; ancak, bağlantı kurulan uygulamalar bağlantı kesilinceye kadar bundan etkilenmez.

Bir uygulama bir grup kurtarma birimiyle bağlantı kurduğunda ve hangi hareketlerin şüpheli olduğunu sorduğunda ya da kuyruk paylaşım grubunda (QSG) başka bir yerde başlatılan bir hareketi çözme girişiminde bulunduğunda, bağlı olduğu kuyruk yöneticisinin isteği işleyebilmesi için kuyruk paylaşım grubunun diğer üyeleriyle iletişim kurabilmesi gerekir. Bunu yapmak için SYSTEM.QSG.UR.RESOLUTION.QUEUE. Bu kuyruk, CSQSYSAPPL adlı kurtarılabilir bir uygulama yapısında olmalıdır. Çözüm istekleri işlenirken kalıcı iletiler bu kuyrukta saklandığı için yapı kurtarılabilir olmalıdır.

GROUP kurtarma birimlerini etkinleştirmeden önce, bağlaşım olanağı yapısının ve paylaşılan kuyruğun tanımlandığından emin olmanız gerekir. CSQ4INSS örneğindeki tanımlamaları kullanabilirsiniz. Kuyruk tanımlandığında ya da başlatma sırasında saptandığında, kuyruk paylaşım grubundaki her kuyruk yöneticisi, gelen istekleri alabilmesi için kuyruğu açar. Kuyruk yanlış tanımlandığı için kuyruğu silmek ya da taşımak istiyorsanız, kuyruk yöneticilerinin MQGET isteklerini engellemek için kuyruk nesnesini güncelleyerek açık tanıtıcılarını kapatmalarını isteyebilirsiniz. Gerekli düzeltmeleri yaptıktan sonra, uygulamaların kuyruktan bir kez daha ileti almalarına izin vermeniz, her kuyruk yöneticisini yeniden açması için yönlendirir. Bir kuyrukta açık olan tanıtıcıları tanımlamak için DISPLAY QSTATUS komutunu kullanın.

Bu kuruluşu tamamladığınızda, hareket işleme uygulamalarının bir grup kurtarma birimiyle bağlanabilmesini istediğiniz her kuyruk yöneticisinde GROUP kurtarma birimlerini etkinleştirebilirsiniz. Bu, kuyruk paylaşım grubu içindeki tüm kuyruk yöneticilerinin olması gerekmez, ancak kuyruk paylaşım grubunun bir alt kümesinde bu işlevselliği etkinleştirmeyi seçerseniz, uygulamalarınızın yalnızca etkinleştirdiğiniz kuyruk yöneticilerine bağlanmayı denediğinden emin olmanız gerekir. Daha fazla bilgi için bkz “Desteklenen Uygulamalar” sayfa 270.

GROUPUR kuyruk yöneticisi özniteliğini etkinleştirme girişiminde bulunduğunuzda, bir dizi yapılandırma denetimi gerçekleştirilir. Kuyruk yöneticisi aşağıdakileri denetler:

- Bir kuyruk paylaşım grubuna ait.
- SYSTEM.QSG.UR.RESOLUTION.QUEUE , CSQ4INSStanımına göre tanımlandı.
- SYSTEM.QSG.UR.RESOLUTION.QUEUE , CSQSYSAPPL adlı kurtarılabilir bir CF yapısında.

Yukarıdaki denetimlerden herhangi biri başarısız olursa, GROUPUR özniteliği devre dışı kalır ve bir ileti kodu döndürülür.

Kuyruk yöneticisi özniteliği etkinleştirildiyse, kuyruk yöneticisi başlatıldığında da bu yapılanış denetimleri gerçekleştirilir. Grup kurtarma biriminin başlatılması sırasında herhangi bir denetim başarısız olursa ve kuyruk yöneticisi hangi denetimin başarısız olduğunu bildiren bir ileti gönderirse. Gerekli düzeltme işlemini gerçekleştirdiğinizde, kuyruk yöneticisi özniteliğini yeniden etkinleştirmeniz gerekir.

z/OS Desteklenen Uygulamalar

Hangi uygulamaların bir GROUP kurtarma birimiyle bağlanabileceğini saptamak için bu sayfayı kullanın.

GROUP kurtarma birimi yok etme desteği, IBM MQ for z/OS ' in kaynak yöneticisi olduğu, ancak hareket eşgüdümcüsü olmadığı belirli hareket uygulamaları tipleriyle sınırlıdır. Şu anda desteklenen hareket uygulamaları şunlardır:

- IBM MQ genişletilmiş işlemsel istemci uygulamaları
- WebSphere Application Servergibi bir uygulama sunucusunda çalışan IBM MQ classes for JMS uygulamaları.
- CICS MQCONN kaynak tanımlaması RESYNCMEMBER (GROUPRESYNC) ile yapılandırıldığında, CICS Transaction Server 4.2 ya da daha sonraki sürümlerde çalışan CICS uygulamaları.

İlgili kavramlar

“IBM MQ genişletilmiş işlemsel istemci uygulamaları” sayfa 271
IBM MQ genişletilmiş hareket işleme istemcisi uygulamalarının GROUP kurtarma birimini nasıl kullanabileceğini belirlemek için bu sayfayı kullanın.

“CICS uygulamalar” sayfa 271
CICS ' in GROUP kurtarma birimini nasıl kullanabileceğini belirlemek için bu sayfayı kullanın.

z/OS *IBM MQ genişletilmiş işlemsel istemci uygulamaları*

IBM MQ genişletilmiş hareket işleme istemcisi uygulamalarının GROUP kurtarma birimini nasıl kullanabileceğini belirlemek için bu sayfayı kullanın.

IBM MQ genişletilmiş hareket işleme istemcisi uygulaması örneği, JMS kullanan ve yerel bağ tanımları yerine TCP/IP üzerinden IBM MQ ' ye bağlanarak WebSphere Application Serveriçinde çalışan bir uygulamadır. Bu istemci uygulamaları, TCP/IP gibi ağ bağlantıları üzerinden IBM MQ for z/OS ' e bağlanır. Bu uygulamalar için, xa_open çağrısında iletilen xa_info dizgisinin QMNAME parametresi için belirlenen değer, bir QMGR ya da GROUP kurtarma yok etme biriminin kullanılıp kullanılmadığını belirtir. xa_open ile ilgili daha fazla bilgi için bkz. xa_open dizgisinin biçimi ve xa_open için ek hata işleme. JMS uygulamaları için bu, belirli bir kuyruk yöneticisinin adı yerine ConnectionFactory içindeki kuyruk paylaşım grubunun (QSG) adı belirtilerek yapılır.

XA istemci uygulamalarının GROUP kurtarma atma birimini kullanmanın avantajlarından yararlanabilmesi için, TCP/IP kurulumunuzu, istemci uygulamalarınızın belirli bir kuyruk yöneticisi yerine, GROUPUR özniteliğini etkinleştiren kuyruk paylaşım grubundaki kuyruk yöneticilerine yöneltilmesine izin verecek şekilde yapılandırmanız gerekir. Bunu yapmak için kullanabileceğiniz dinamik sanal IP adresi teknolojilerinden biri z/OS SysPlex Distribütörüdür. Daha fazla ayrıntı için bkz. z/OS Communications Server ve z/OS Temel Beceriler: Dinamik sanal adresleme . Grup kurtarma birimlerini kuyruk paylaşım grubunuzun kuyruk yöneticilerinin bir alt kümesinde etkinleştirmek istiyorsanız, istemci uygulamalarınızın etkinleştirilmedikleri uygulamalara yöneltilemeyeceğini doğrulayın.

İstemci uygulamalarınızın paylaşılan kanalları kullanarak kuyruk paylaşım grubuna bağlanması gerekmez.

z/OS *CICS uygulamalar*

CICS ' in GROUP kurtarma birimini nasıl kullanabileceğini belirlemek için bu sayfayı kullanın.

CICS 4.2 ve sonraki sürümleri, MQCONN kaynak tanımlamasında RESYNCMEMBER (GROUPRESYNC) grup yeniden eşzamanlama seçeneğini sağlar. Bu seçenekle yapılandırılan bir CICS , CICS bölgesiyle aynı LPAR üzerinde çalışan bir kuyruk paylaşım grubundaki herhangi bir uygun kuyruk yöneticisine bağlanabilir. CICS GROUPRESYNC seçeneğini desteklemek için bir kuyruk yöneticisinin MQ V7.1 ya da sonraki bir sürümde çalışması ve GROUPUR desteği için etkinleştirilmesi gerekir.

GROUPRESYNC kullanılarak MQ ' ya bağlı bir CICS bölgesinde çalışan hareketler, grup kurtarma atma birimiyle iş birimleri yaratır.

Bir kuyruk yöneticisi hatasından sonra daha hızlı kurtarmayı etkinleştirmek için RESYNCMEMBER (GROUPRESYNC) işlevini kullanabilirsiniz; çünkü CICS bölgesi, kuyruk yöneticisinin yeniden başlatılmasını beklemeden, aynı LPAR üzerinde çalışan alternatif bir uygun kuyruk yöneticisine hemen bağlanabilir ve belirsiz hareketleri çözebilir.

RESYNCMEMBER (GROUPRESYNC), CICSiçin daha esnek yeniden başlatma seçenekleri de sağlar. GROUPRESYNC ve MQ paylaşılan kuyruklarını kullanmak üzere yapılandırılmış MQ bağlantısına sahip bir CICS bölgesi, aynı kuyruk paylaşım grubunun bir üyesi olarak çalışan bir kuyruk yöneticisinin olduğu herhangi bir LPAR ' da yeniden başlatılabilir.

z/OS IBM MQ ve diğer z/OS ürünleri

IBM MQ ürününün diğer z/OS ürünleriyle nasıl çalışabileceğini anlamak için bu konuyu kullanın.

İlgili kavramlar

"IBM MQ ve CICS" sayfa 272
IBM MQ 9.0.0ve sonraki tarafından desteklenen tüm CICS sürümleri, bağdaştırıcının ve köprünün CICS tarafından sağlanan sürümünü kullanır.

"IBM MQ for z/OS ve WebSphere Application Server" sayfa 278
WebSphere Application Servertarafından IBM MQ for z/OS kullanımını anlamak için bu konuyu kullanın.

İlgili başvurular

"IBM MQ ve IMS" sayfa 273
IBM MQ öğesinin IMSile nasıl çalıştığını anlamak için bu konuyu kullanın. IMS bağdaştırıcısı, kuyruk yöneticinizi IMS'e bağlamanızı sağlar ve IMS uygulamalarının MQI' ı kullanmasını sağlar.

"IBM MQ ve z/OS Toplu İş, TSO ve RRS bağdaştırıcıları" sayfa 277
IBM MQ ' in z/OS Toplu İş, TSO ve RRS bağdaştırıcılarıyla nasıl çalıştığını anlamak için bu konuyu kullanın.

z/OS IBM MQ ve CICS

IBM MQ 9.0.0ve sonraki tarafından desteklenen tüm CICS sürümleri, bağdaştırıcının ve köprünün CICS tarafından sağlanan sürümünü kullanır.

IBM MQ CICS bağdaştırıcısını ve IBM MQ CICS bridge bileşenlerini yapılandırma hakkında daha fazla bilgi için CICS belgelerinin IBM MQ için bağlantı yapılandırılması bölümüne bakın.

İlgili görevler

IBM MQ ' yi CICS ile kullanma

z/OS CICS grup bağlantısı

CICS grup bağlantısı, bir CICS bölgesinin tek bir kuyruk yöneticisi belirtmek yerine, aynı LPAR ' daki bir IBM MQ kuyruk paylaşım grubunun herhangi bir etkin üyesine bağlanabilmesini sağlar. CICS , aynı anda tek bir kuyruk yöneticisine bağlanmaya devam etmektedir.

CICS grup eklemeyi desteklemek için LPAR üzerinde en az iki kuyruk yöneticisine gereksinim duyarsınız. Belirli bir kuyruk yöneticisinin etkin olması gerekmediği için, grup eklemenin kullanılması daha yüksek kullanılabilirlik sağlar. CICS , LPAR ' daki kuyruk paylaşım grubundaki herhangi bir kuyruk yöneticisine bağlanır.

Daha fazla bilgi için MQCONN kaynağıyla ilgili CICS belgelerine bakın.

CICS , bir kuyruk yöneticisiymiş gibi MQNAME ile bağlantı kurma girişiminde bulundu:

- Kuyruk yöneticisi varsa ve etkinse, bağlantı çalışır.
- Bağlantı başarısız olursa, CICS , aynı LPAR ' da etkin olan kuyruk yöneticilerinin durumunu sorgular.
- Birden çok kuyruk yöneticisi etkinse, CICS RESYNCMEMBER (YES) ve UOW durumunu denetleyerek CICS ' in belirli bir üyeye bağlanması mı, yoksa bağlanması mı gerektiğini ya da etkin değilse beklemesi mi gerektiğini belirler.
- Belirli bir üyeye bağlanmaya gerek yoksa, CICS bir kuyruk yöneticisi seçer (rasgele hale getirme algoritması kullanarak).
- CICS , seçilen kuyruk yöneticisine bağlanmayı dener.
- Girişim başarısız olursa, dönüş koduna bağlı olarak CICS sonraki üyeyi seçer ve seçim döngüsünden yeniden geçer.
- Etkin kuyruk yöneticisi yoksa, CICS kuyruk yöneticileri listesine birden çok bağlantı verir ve ilk kuyruk yöneticisi kullanılabilir oluncaya kadar ECBLIST üzerinde bekler.

İlgili kavramlar

"CICS için grup kurtarma birimi (GROUPUR)" sayfa 273

IBM MQ GROUPUR for CICS , bir kuyruk paylaşım grubundaki (QSG) belirsiz iş birimleri için eşdüzey kurtarma sağlar. Bir IBM MQ kuyruk yöneticisi, kuyruk paylaşım grubundaki başka bir kuyruk yöneticisi adına belirsiz iş birimlerini çözebilir. Bu, CICS 'in QSG' deki farklı bir kuyruk yöneticisine grup bağlantısı yoluyla yeniden bağlanması durumunda, önceki bir IBM MQ bağlantısından belirsiz hareketleri çözebildiği anlamına gelir.

İlgili bilgiler

IBM MQ kuyruk paylaşım grupları için destek

z/OS CICS için grup kurtarma birimi (GROUPUR)

IBM MQ GROUPUR for CICS , bir kuyruk paylaşım grubundaki (QSG) belirsiz iş birimleri için eşdüzey kurtarma sağlar. Bir IBM MQ kuyruk yöneticisi, kuyruk paylaşım grubundaki başka bir kuyruk yöneticisi adına belirsiz iş birimlerini çözebilir. Bu, CICS 'in QSG' deki farklı bir kuyruk yöneticisine grup bağlantısı yoluyla yeniden bağlanması durumunda, önceki bir IBM MQ bağlantısından belirsiz hareketleri çözebildiği anlamına gelir.

Bir CICS bölgesi bir kuyruk yöneticisiyle çalışıyorsa ve kuyruk yöneticisi olağandışı sona ererse, belirsiz hareketler kurtarılır. Bu, CICS bölgesinin, üzerinde çalıştığı kuyruk yöneticisinin yeniden başlatılmasını beklemesi ve daha sonra, şüpheli iş birimlerini çözmesi gereksinimini ortadan kaldırır. Bu, LPAR ' da en az iki kuyruk yöneticisine ihtiyacınız olduğu anlamına gelir; böylece CICS , ilk kuyruk yöneticisinin olağandışı bir şekilde sonlandırılması durumunda başka bir kuyruk yöneticisine bağlanabilir.

CICS MQCONN tanımlamasındaki yeni RESYNCMEMBER (GROUPRESYNC) ayarı:

- IBM MQ grup bağlama işlevini ve eşdüzey kurtarma işlevini kullanır.
- GROUPUR özniteliği etkinleştirilmiş bir kuyruk yöneticisi gerektirir.
- Yine de var olan CICS MQCONN RESYNCMEMBER ayarlarını destekler (YES ve NO):
  - Var olan CICS grup ekleme işlevini kullanır ve eşdüzey kurtarma yoktur.
  - RESYNCMEMBER ayarlarının değiştirilmesi, CICS sonraki IBM MQbağlantısına geçişte yürürlüğe girer.

İlgili kavramlar

“GROUP kurtarma birimlerinin etkinleştirilmesi” sayfa 270

Bir kuyruk paylaşım grubu, grup kurtarma birimlerini yapılandırabilir ve bu desteği etkinleştirebilir.

z/OS IBM MQ ve IMS

IBM MQ öğesinin IMSile nasıl çalıştığını anlamak için bu konuyu kullanın. IMS bağdaştırıcısı, kuyruk yöneticinizi IMS'e bağlamanızı sağlar ve IMS uygulamalarının MQI' ı kullanmasını sağlar.

İsteğe bağlı ek IBM MQ - IMS köprüsü, uygulamaların MQI kullanmayan bir IMS uygulamasını çalıştırmalarını sağlar. Bu, eski uygulamalarınızı IBM MQile yeniden yazmaya gerek kalmadan kullanabileceğiniz anlamına gelir.

Bu bileşenlerle ilgili daha fazla bilgi için aşağıdaki alt konulara bakın:

İlgili kavramlar

IBM MQ for z/OS üzerinde IMS ve IMS köprü uygulamaları

İlgili görevler

IMS bağdaştırıcısının ayarlanması

IMS köprüsünün ayarlanması

IMS bağdaştırıcısının çalıştırılması

İlgili başvurular

MQIIH- IMS bilgi üstbilgisi

z/OS IMS bağdaştırıcısı

IMS bağdaştırıcısı, IMS uygulama programları ile IBM MQ altsistemi arasındaki bir arabirimdir.

IBM MQ bağdaştırıcıları, farklı uygulama ortamlarının ileti kuyruklama ağı üzerinden ileti göndermesini ve almasını sağlar. IMS bağdaştırıcısı, IMS uygulama programları ile bir IBM MQ altsistemi arasındaki arabirimdir. IMS uygulama programlarının MQI kullanmasını sağlar.

IMS bağdaştırıcısı, IMStarafından sağlanan External Subsystem Attach Facility (ESAF) (Dış Altsistem Bağlantı Olanağı) olanağını kullanarak IBM MQ ' e erişim isteklerini alır ve yorumlar. Genellikle IMS , işletmen müdahalesi olmadan IBM MQ ' e otomatik olarak bağlanır.

IMS bağdaştırıcısı, aşağıdaki kiplerde ya da durumlarda çalışan programlar için IBM MQ kaynaklarına erişim sağlar:

- Görev (TCB) kipi
- Sorun durumu
- Bellek dışı kip
- Erişim dışı kayıt kipi

Bağdaştırıcı, bir uygulama görevi denetim bloğundan (TCB) IBM MQ' ye bir bağlantı iş parçacığı sağlar.

Bağdaştırıcı, syncpoint eşgüdümcüsü olarak IMS ile IBM MQ ' in sahip olduğu kaynaklarda yapılan değişiklikler için iki aşamalı bir kesinleştirme protokolünü destekler. IMS ' in syncpoint eşgüdümcüsü olmadığı etkileşimler (örneğin, APPC korumalı (SYNCLVL = SYNCPT) etkileşimleri, IMS bağdaştırıcısı tarafından desteklenmez.

Bağdaştırıcı ayrıca bir tetikleyici izleme hareketi (CSQQTRMN) de sağlar. Bu, “IMS tetikleyici izleme programı” sayfa 274içinde açıklanmıştır.

Bir IMS hatasından kurtarmaya yardımcı olmak için IBM MQ komutunu IMS Extended Recovery Facility (XRF) ile kullanabilirsiniz.

Not: IMS 15.2 Extended Recovery Facility (XRF) artık desteklenmemektedir. Daha fazla bilgi için IMS belgelerine bakın.

Bağdaştırıcının kullanılması

Uygulama programları ve IMS bağdaştırıcısı aynı adres alanında çalışır. Kuyruk yöneticisi, kendi adres alanında ayrıdır.

Uygun bir IMS dil arabirimi modülüne bir ya da daha fazla MQI çağrısı düzenleyen her bir programı, dinamik MQI çağrıları kullanmadığı sürece IBM MQtarafından sağlanan API kod parçası programı CSQQSTUB ' yi bağlamanız gerekir. Uygulama bir MQI çağrısı yayınladığında, sınırlı kod öbeği denetimi, ileti kuyruğu yöneticisi tarafından isteğin işlenmesini yöneten IMS dış altsistem arabirimi aracılığıyla bağdaştırıcıya aktarır.

IMS ile sistem yönetimi ve işletim

Yetkili bir IMS uçbirim işletmeni, IBM MQbağlantısını denetlemek ve izlemek için IMS komutları verebilir. Ancak, IMS uçbirim işleci IBM MQ adres alanı üzerinde denetime sahip değildir. Örneğin, işleç IBM MQ ' i bir IMS adres alanından kapatamadı.

Sınırlamalar

Aşağıdaki IBM MQ API çağrıları, IMS bağdaştırıcısını kullanan bir uygulamada desteklenmez:

- MQCB
- MQCB_FUNCTION
- MQCTL

IMS tetikleyici izleme programı

IMS tetikleyici izleme programı (**CSQQTRMN**) bir IBM MQ olayı ortaya çıktığında (örneğin, belirli bir kuyruğa bir ileti konduğunda) IMS işlemini başlatan IBM MQtarafından sağlanan IMS bir uygulamadır.

Nasıl çalışır

Bir ileti uygulama ileti kuyruğuna konduğunda, tetikleyici koşulları karşılandığında bir tetikleyici oluşturulur. Kuyruk yöneticisi daha sonra, *tetikleyici iletisi*olarak bilinen bir iletiyi (bazı kullanıcı tanımlı verileri içeren) o ileti kuyruğu için belirtilen başlatma kuyruğuna yazar. IMS ortamında, başlatma kuyruğunu izlemek ve tetikleyici iletilerini geldiklerinde almak için bir CSQQTRMN eşgörünümü başlatabilirsiniz. Genellikle, CSQQTRMN, IMS ileti kuyruğuna bir INSERT (ISRT) ile başka bir IMS hareketi zamanlar. Başlatılan IMS uygulaması, iletiyi uygulama ileti kuyruğundan okur ve işler. CSQQTRMN, ileti olmayan bir BMP olarak çalıştırılmalıdır.

Her CSQQTRMN kopyası tek bir başlatma kuyruğuna hizmet vermektedir. Başlatıldığında, tetikleyici izleme programı IBM MQ ya da IMS sona erinceye kadar çalışır.

CSQQTRMN için APPLCTN makrosu SCHDTYP=PARALLEL değerini belirtmelidir.

Tetikleyici izleme programı toplu iş odaklı bir BMP olduğundan, tetikleyici izleme programı tarafından başlatılan IMS hareketleri aşağıdakileri içerir:

- IOPCB ' nin LTERM alanındaki boşluklar
- IOPCB ' nin Kullanıcı Kimliği alanındaki tetikleyici izleme programı BMP 'sinin PSB adı

Hedef IMS hareketi Güvenlik Sunucusu tarafından korunuyorsa (daha önce RACF olarak biliniyordu), CSQQTRMN ' yi Güvenlik Sunucusu 'na kullanıcı kimliği olarak tanımlamanız gerekebilir.

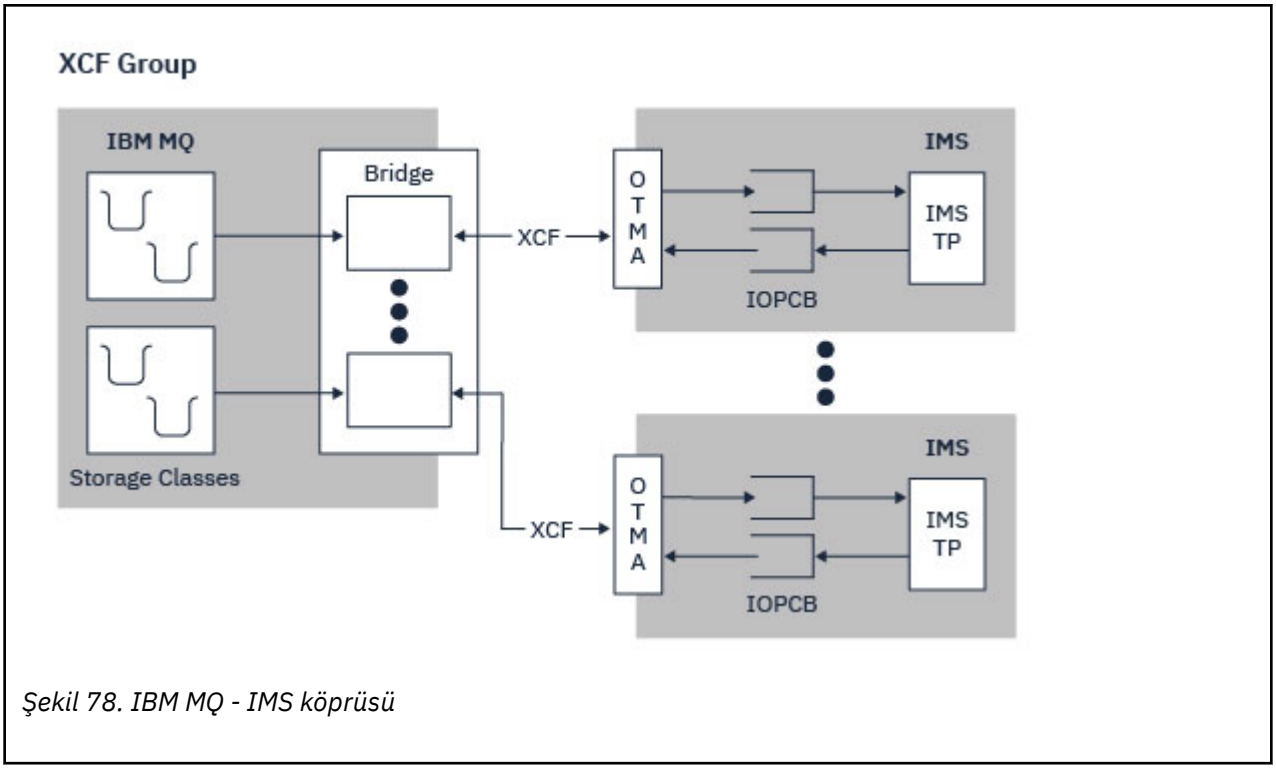
z/OS IBM MQ - IMS köprüsü

IBM MQ - IMS köprüsü, IBM MQ uygulamalarından IMS sisteminizdeki uygulamalara doğrudan erişim sağlayan IBM MQ for z/OS bileşenidir.

IBM MQ - IMS köprüsü, *örtük MQI desteğini*etkinleştirir. Başka bir deyişle, 3270 bağlantılı uçbirimler tarafından denetlenen eski uygulamaların yeniden yazılması, yeniden derlenmesi ya da yeniden bağlanması gerekmeden IBM MQ iletileri tarafından denetlenmesi için bunları yeniden tasarlayabilirsiniz. Köprü, bir IMS *Open Transaction Manager Access* (OTMA) istemcisidir.

Köprü uygulamalarında, IMS uygulaması içinde IBM MQ çağrısı yoktur. Uygulama girişini IOPCB 'ye bir GET UNIQUE (GU) kullanarak alır ve çıkışını ISRT kullanarak IOPCB' ye gönderir. IBM MQ uygulamaları, uygulamaların programlanamayan uçbirimler tarafından yönlendirildiğinde olduğu gibi yürütülebildiğinden emin olmak için ileti verilerinde IMS üstbilgisini (MQIIH yapısı) kullanır. Çok bölümlü iletileri işleyen bir IMS uygulaması kullanıyorsanız, tüm bölümlerin tek bir IBM MQ iletisinde bulunması gerektiğini unutmayın.

IMS köprüsü Şekil 78 sayfa 276içinde gösterilmektedir.



Şekil 78. IBM MQ - IMS köprüsü

Bir kuyruk yöneticisi bir ya da daha çok IMS sistemine bağlanabilir ve birden çok kuyruk yöneticisi birden çok IMS sistemine bağlanabilir. Tek kısıtlama, hepsinin aynı XCF grubuna ait olması ve hepsinin aynı sistem şebekesinde olması gerekmesi.

IMS köprüsünün ayarlanması ve aynı kuyruk yöneticisine ek IMS bağlantısı eklenmesiyle ilgili bilgi için IMS köprüsünün ayarlanması başlıklı konuya bakın.

OTMA nedir?

IMS OTMA olanağı, IMSüzerinde çalışan, işlem tabanlı bağlantısız bir istemci/sunucu iletişim kuralıdır. z/OS Cross Systems coupling facility (XCF)aracılığıyla IMS TM uygulamalarına erişen anasistem tabanlı iletişim sunucuları için bir arabirim işlevi görür.

OTMA, istemcilerin büyük bir ağ ya da çok sayıda oturum için IMS ile istemciler arasındaki etkileşimler için yüksek performans sağlamak üzere IMS ' e bağlanmalarını sağlar. OTMA, z/OS sysplex ortamında uygulanır. Bu nedenle, OTMA etki alanı XCF etki alanıdır.

OTMA Kaynak İzleme

IMS v10 ya da sonraki düzeylerde bulunan x '3C' OTMA iletişim kuralı iletileri için destek, IBM MQ for z/OSiçindeki IBM MQ - IMS köprüsünde bulunur. Bu iletiler, IMS tarafından OTMA istemcilerine sağlık durumunu raporlamak üzere gönderilir.

Bir IMS iş ortağı, gönderilmekte olan işlem istekleri hacmini işleyemezse, IBM MQ ' e bir sel uyarısı oluştuğunu bildirir. Yanıt olarak IBM MQ , isteklerin köprü üzerinden gönderilme hızını yavaşlatır.

IMS yine de işlem isteklerini işleyemezse ve tam taşma koşulu oluşursa, IMS iş ortağına yönelik tüm TPIPE ' ler askıya alınır. IMS iş ortağından sel ya da sel uyarısı koşulunun hafifletildiğine IBM MQ ilişkin bildirim üzerine, uygunsa, askıya alınan tüm TPIPE ' ler devam edecek ve işlem isteklerinin gönderilme hızını, hız üst sınırına ulaşılıncaya kadar kademeli olarak artıracaktır. Konsol iletileri, IMS iş ortaklarının durumundaki bir değişikliğe yanıt olarak IBM MQ tarafından yayınlanır.

IMS v10 iş ortakları kullanılıyorsa, UK45082 PTF ' nin uygulandığından emin olmanız gerekir.

IBM MQ ' dan IMS hareketlerinin gönderilmesi

Köprüyü kullanan bir IMS işlemini sunmak için, uygulamalar iletileri her zamanki gibi IBM MQ kuyruğuna koyar. İletiler IMS işlem verilerini içerir; bunlar bir IMS üstbilgisine (MQIIH yapısı) sahip olabilir ya da IBM MQ - IMS köprüsünün iletideki verilerle ilgili varsayımlar yapmasına izin verebilir.

IBM MQ daha sonra iletiyi bir IMS kuyruğuna yerleştirir (veri bütünlüğünü sağlamak için eşitleme noktalarının kullanımını etkinleştirmek üzere önce IBM MQ kuyruğunda kuyruğa alınır). IBM MQ kuyruğunun depolama sınıfı, kuyruğun bir *OTMA kuyruğu* (yani, iletileri IBM MQ - IMS köprüsüne iletmek için kullanılan bir kuyruk) ve ileti verilerinin gönderildiği IMS ortak olup olmadığını belirler.

Uzak kuyruk yöneticileri, IMS hareketlerini IBM MQ for z/OSüzerindeki bu OTMA kuyruklarına yazarak da başlatabilir.

IMS sisteminden döndürülen veriler, ileti tanımlayıcı yapısında (MQMD) belirtilen IBM MQ yanıt kuyruğuna doğrudan yazılır. (MQMD ' nin **`ReplyToQMgr`** alanında belirtilen kuyruk yöneticisine bir iletim kuyruğu olabilir.)

İlgili kavramlar

IBM MQ for z/OS üzerinde IMS ve IMS köprü uygulamaları

İlgili görevler

IMS köprüsünün özelleştirilmesi

İlgili başvurular

“IBM MQ ve IMS” sayfa 273

IBM MQ öğesinin IMSile nasıl çalıştığını anlamak için bu konuyu kullanın. IMS bağdaştırıcısı, kuyruk yöneticinizi IMS'e bağlamanızı sağlar ve IMS uygulamalarının MQI' ı kullanmasını sağlar.

z/OS IBM MQ ve z/OS Toplu İş, TSO ve RRS bağdaştırıcıları

IBM MQ ' in z/OS Toplu İş, 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 kullanmasını sağlar.

Bağdaştırıcılar, aşağıdaki kiplerde ya da durumlarda çalışan programlar için IBM MQ kaynaklarına erişim sağlar:

- Görev (TCB) kipi
- Sorun ya da gözetmen durumu
- Bellek dışı kip
- Erişim dışı kayıt kipi

Uygulama programları ile IBM MQ arasındaki bağlantılar görev düzeyindedir. Bağdaştırıcılar, bir uygulama görevi denetim bloğundan (TCB) IBM MQ' ye bir bağlantı iş parçacığı sağlar.

Batch/TSO bağdaştırıcısı, IBM MQ' in sahip olduğu kaynaklarda yapılan değişiklikler için tek fazlı bir kesinleştirme iletişim kuralını destekler. Çok aşamalı kesinleştirme protokollerini desteklemez. RRS bağdaştırıcısı, IBM MQ uygulamalarının, z/OS Resource Recovery Services (RRS) tarafından koordine edilen, RRS etkin diğer ürünlerle iki aşamalı kesinleştirme protokollerine katılmalarını 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 bloğu (IRB) çalıştırır. Bu IRB, IBM MQ sonlandırma ECB ' nin gönderilip gönderilmediğini denetler. Sonlandırma ECB 'si gönderildiyse, IRB, IBM MQ içinde bir olayı bekleyen uygulama ECB ' lerini gönderir (örneğin, bir sinyal ya da bekleme).

Batch/TSO bağdaştırıcısı

IBM MQ Batch/TSO bağdaştırıcısı, z/OS Batch ve TSO uygulamaları için IBM MQ desteği sağlar. z/OS Toplu iş ya da TSO altında çalışan tüm uygulama programlarında, API sınırlı kod öbeği programı CSQBSTUB bağlantısı düzenlenmelidir. Sınırlı kod öbeği, uygulamaya tüm MQI çağrılarına erişim sağlar. Uygulamalar için tek aşamalı kesinleştirme ve geriletme özelliğini, **MQCMIT** ve **MQBACK**MQI çağrılarını yayınlayarak kullanırsınız.

RRS bağdaştırıcısı

Resource Recovery Services (RRS), z/OS ürünleri arasında iki aşamalı kesinleştirmenin koordine edilmesi için sistem çapında hizmet sağlayan bir z/OS alt bileşenidir. IBM MQ Batch/TSO RRS bağdaştırıcısı (RRS bağdaştırıcısı), bu hizmetleri kullanmak isteyen z/OS Toplu İş ve TSO uygulamaları için IBM MQ desteği sağlar. RRS bağdaştırıcısı, IBM MQ ' in RRS koordinasyonunda tam katılımcı olmasını sağlar. Uygulamalar, RRS ' yi destekleyen diğer ürünlerle (örneğin, Db2) iki aşamalı kesinleştirme işlemine katılabilir.

RRS bağdaştırıcısı iki sınırlı kod öbeği sağlar; RRS ' yi kullanmak isteyen uygulama programlarını bu sınırlı kod öbeklerinden biriyle bağlamanız gerekir.

CSQBRSTB
Bu sınırlı kod öbeği, **MQCMIT** ve **MQBACK**MQI çağrıları yerine RRS çağrılabilir kaynak kurtarma hizmetlerini kullanarak uygulamalar için iki aşamalı kesinleştirme ve geriletme kullanmanızı sağlar.

SYS1.CSSLIB . MQI çağrılarını **MQCMIT** ve **MQBACK**kullanıyorsanız, `MQRC_ENVIRONMENT_ERROR`dönüş kodunu alırsınız.

CSQBRRSI
Bu sınırlı kod öbeği, **MQCMIT** ve **MQBACK** MQI çağrılarını kullanmanıza olanak sağlar; IBM MQ , bu çağrıları **SRRCMIT** ve **SRRBACK** RRS çağrıları olarak uygular.

RRS bağdaştırıcısını kullanan uygulama programları oluşturma hakkında bilgi için bkz. RRS toplu iş bağdaştırıcısı.

z/OS Toplu iş, TSO ve RRS bağdaştırıcılarına ilişkin daha fazla bilgi nerede bulunur?

Aşağıdaki kaynaklarda bu bölümdeki konularla ilgili daha fazla bilgi bulabilirsiniz:

Çizelge 25. z/OS Toplu İş 'i IBM MQ ile kullanma hakkında daha fazla bilgi nereden edinebilirsiniz?

Konu	Nereye bakıyorsunuz?
Toplu İş bağdaştırıcılarının ayarlanması	Görev 19: Toplu İş, TSO ve RRS bağdaştırıcılarını ayarlama
RRS çağrılabilir kaynak kurtarma hizmetleri	*MVS Programlaması: Üst Düzey Diller İçin Çağrılabilir Hizmetler*

z/OS IBM MQ for z/OS ve WebSphere Application Server

WebSphere Application Servertarafından IBM MQ for z/OS kullanımını anlamak için bu konuyu kullanın.

WebSphere Application Server altında çalışan Java içinde yazılan uygulamalar, ileti alışverişi gerçekleştirmek için Java Message Service (JMS) belirtimini kullanabilir. Bu ortamdaki noktadan noktaya ileti sistemi bir IBM MQ for z/OS kuyruk yöneticisi tarafından sağlanabilir.

İleti alışverişi sağlamak için IBM MQ for z/OS kuyruk yöneticisini kullanmanın bir yararı, bağlanan JMS uygulamalarının bir IBM MQ ağının işlevlerine tam olarak katılabilmektir. Örneğin, IMS köprüsünü kullanabilir ya da diğer platformlarda çalışan kuyruk yöneticileriyle ileti alışverişi yapabilirler.

WebSphere Application Server ile bir kuyruk yöneticisi arasındaki bağlantı

Daha fazla bilgi için bkz. IBM MQ ve WebSphere Application Server birlikte kullanma .

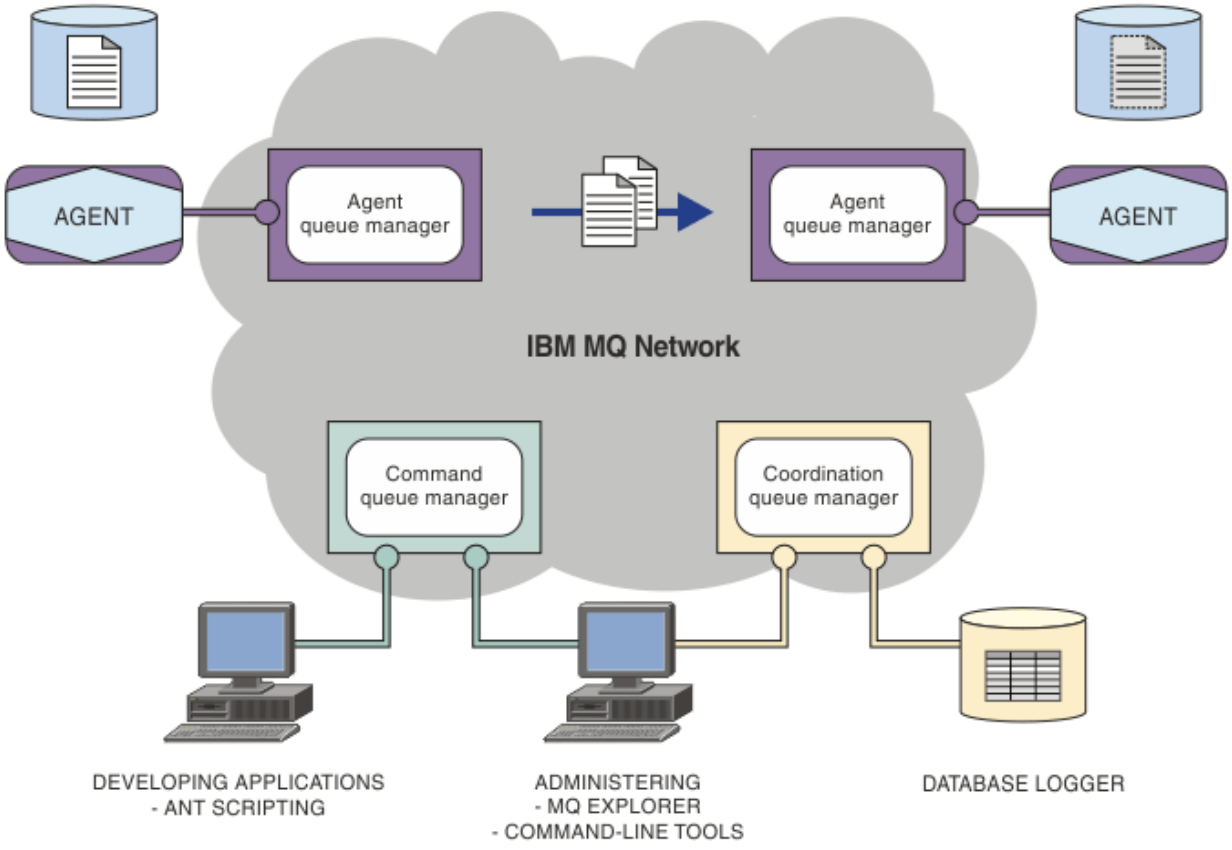
JMS uygulamalarındaki IBM MQ işlevlerinin kullanılması

Varsayılan olarak, IBM MQ kuyruklarında tutulan JMS iletileri, JMS ileti üstbilgisinden bazılarını tutmak için MQRFH2 üstbilgisini kullanır. Birçok kalıt IBM MQ uygulaması bu üstbilgilere sahip iletileri işleyemez ve kendi ayırıcı özellik üstbilgilerini (örneğin, MQCIH for CICS Bridge ya da MQWIH for IBM MQ Workflow uygulamaları) gerektirir. Bu özel konularla ilgili daha fazla ayrıntı için JMS iletilerini IBM MQ iletilerle eşlemebaşlıklı konuya bakın.

Managed File Transfer

Managed File Transfer , dosya boyutundan ya da kullanılan işletim sistemlerinden bağımsız olarak, yönetilen ve denetlenebilir bir şekilde sistemler arasında dosya aktarır.

Dosya aktarımlarını yönetmenizi, güvenmenizi ve güvenli hale gelmenizi sağlayan özelleştirilmiş, ölçeklenebilir ve otomatikleştirilmiş bir çözüm oluşturmak için Managed File Transfer ' yi kullanabilirsiniz. Managed File Transfer , yüksek maliyetli yedekleri ortadan kaldırır, bakım maliyetlerini düşürür ve var olan BT yatırımlarınızı en üst düzeye çıkarır.



Çizge, basit bir Managed File Transfer topolojisini gösterir. Her biri IBM MQ ağında kendi aracı kuyruk yöneticisine bağlanan iki aracı vardır. Bir dosya, çizgenin bir tarafındaki aracıdan IBM MQ ağı üzerinden çizgenin diğer tarafındaki aracıya aktarılır. Ayrıca, IBM MQ ağında koordinasyon kuyruk yöneticisi ve bir komut kuyruğu yöneticisi vardır. Uygulamalar ve araçlar, IBM MQ ağındaki Managed File Transfer etkinliğini yapılandırmak, yönetmek, çalıştırmak ve günlüğe kaydetmek için bu kuyruk yöneticilerine bağlanır.

Managed File Transfer , işletim sisteminize ve genel kuruluşa bağlı olarak dört farklı aksam olarak kurulabilir. Bu seçenekler şunlardır: Managed File Transfer Agent, Managed File Transfer Logger, Managed File Transfer Serviceya da Managed File Transfer Tools. Daha fazla bilgi için bkz. Yönetilen Dosya Aktarımı ürün seçenekleri.

Aşağıdaki görevleri gerçekleştirmek için Managed File Transfer kullanabilirsiniz:

- Yönetilen dosya aktarımları oluştur
  - Linux Windows Linux ya da Windows platformlarında IBM MQ Explorer içinden yeni dosya aktarımları oluşturun.
  - Desteklenen tüm platformlarda komut satırından yeni dosya aktarımları oluşturun.
  - Dosya aktarma işlevini Apache Ant aracıyla bütünleştirin.
  - Aracı komut kuyruklarına ileti koyarak Managed File Transfer ' u denetleyen uygulamaları yazın.
  - Dosya aktarımlarını daha sonra gerçekleşecek şekilde zamanlayın. Zamanlanmış dosya aktarımlarını, bir dizi dosya sistemi olayına (örneğin, yaratılmakta olan yeni bir dosya) dayalı olarak da tetikleyebilirsiniz.
  - Bir kaynağı (örneğin, bir dizin) sürekli olarak izleyin ve o kaynağın içeriği önceden tanımlanmış bazı koşulları karşıladığında bir görev başlatın. Bu görev bir dosya aktarımı, Ant komut dosyası ya da JCL işi olabilir.
  - Dosyaları IBM MQ kuyruklarına/kuyruklarından aktarın.
  - Dosyaları FTP, FTPS ya da SFTP sunucularına/sunucularından aktarın.
  - Dosyaları Connect:Direct düğümlerine/düğümlerinden aktarın.
  - Hem metin dosyalarını hem de ikili dosyaları aktarın. Metin dosyaları, kaynak ve hedef sistemlerin kod sayfaları ile satır sonu kuralları arasında otomatik olarak dönüştürülür.
  - Güvenli Yuva Katmanı (SSL) tabanlı bağlantılar için sektör standartları kullanılarak aktarımların güvenliği sağlanabilir.
- Devam eden aktarımları görüntüleyin ve ağınızdaki tüm aktarımlara ilişkin bilgileri günlüğe kaydedin
  - Linux Windows Linux ya da Windows platformlarında IBM MQ Explorer ' dan devam eden aktarımların durumunu görüntüleyin.
  - Linux Windows Linux ya da Windows platformlarında IBM MQ Explorer komutunu kullanarak tamamlanan aktarımların durumunu denetleyin.
  - Günlük iletilerini bir Db2 ya da Oracle veritabanına kaydetmek için Managed File Transfer veritabanı kaydedici özelliğini kullanın.

Managed File Transfer , uygulamalar arasında iletilerin güvenli, yalnızca bir kez teslim edilmesini sağlayan IBM MQtemel alınarak oluşturulmuştur. IBM MQ' in çeşitli özelliklerinden yararlanabilirsiniz. Örneğin, IBM MQ kanalları üzerinden aracılar arasında gönderdiğiniz verileri sıkıştırmak için kanal sıkıştırmayı kullanabilir ve aracılar arasında gönderdiğiniz verilerin güvenliğini sağlamak için SSL kanallarını kullanabilirsiniz. Dosyalar güvenilir bir şekilde aktarılır ve dosya aktarımının gerçekleştirildiği altyapının arızalanmasını tolere edebilir. Bir ağ kesintisi yaşarsanız, bağlantı geri yüklendiğinde dosya aktarımı kaldığı yerden yeniden başlar.

Dosya aktarımını var olan IBM MQ ağınızla birleştirerek, iki ayrı altyapıyı korumak için gereken kaynakları harcamaktan kaçınabilirsiniz. Önceden bir IBM MQ müşterisi değilseniz, Managed File Transfer desteklemek için bir IBM MQ ağı yaratarak gelecekteki bir SOA uygulaması için omurga oluşturursınız. Zaten bir IBM MQ müşterisiyseniz, Managed File Transfer IBM MQ Internet Pass-Thru ve IBM Integration Busdahil olmak üzere var olan IBM MQ altyapınızdan yararlanabilir.

Managed File Transfer yapılandırmanızın esnekliğini artırmak için IBM MQ yüksek düzeyde kullanılabilirlik çözümlerinden yararlanabilirsiniz. Aracılarınız eşlenmiş veri kuyruğu yöneticilerini (RDIMM ' ler) kullanıyorsa, bunları kayan IP adresi özelliğini kullanacak şekilde yapılandırmanız gerekir. Bu, aracıların şu anda çalışmakta olan üç RDQM eşgörünümünden hangisiyle iletişim kurmak için aynı IP adresini kullandığı ve hata durumunda yedek sisteme geçiş sırasında otomatik olarak yeniden bağlandığı anlamına gelir (bkz.

RDQM yüksek kullanılabilirlik ve Kayan IP adresi oluşturma ve silme). Çok eşgörünümlü kuyruk yöneticisi çözümünü kullanırsanız, uygulamalar her bir eşgörünümle iletişim kurmak için farklı bir IP adresi kullanır; bu adres, hata durumunda yedek sisteme geçiş sırasında istemci yeniden bağlanması ile işlenir (bkz. Çok eşgörünümlü 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 parçası olarak Managed File Transfer tarafından aktarılan dosyaları işleyin. Daha fazla bilgi için, bkz. IBM Integration Bus içinden MFT ile çalışma.

IBM Sterling Connect:Direct

Managed File Transfer Connect:Direct köprüsünü kullanarak var olan bir Connect:Direct ağına/ağından dosya aktarın. Daha fazla bilgi için bkz. Connect:Direct köprüsü.

IBM Tivoli Composite Application Manager

IBM Tivoli Composite Application Manager , koordinasyon kuyruk yöneticisine yayınlanan bilgileri izlemek için kullanabileceğiniz bir aracı sağlar.

İlgili kavramlar

Yönetilen Dosya Aktarma ürün seçenekleri

“MFT topolojiye genel bakış” sayfa 282
Managed File Transfer aracılarının bir IBM MQ ağındaki koordinasyon kuyruk yöneticisine nasıl bağlandığına ilişkin genel bakış.

“MFT IBM MQile nasıl çalışır?” sayfa 281
Managed File Transfer , IBM MQile çeşitli şekillerde etkileşim kurmaktadır.

MFT IBM MQile nasıl çalışır?

Managed File Transfer , IBM MQile çeşitli şekillerde etkileşim kurmaktadır.

- 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.
- Aracı, IBM MQ günlükleriniz üzerindeki etkiyi en aza indirmek için kalıcı olmayan iletileri kullanarak dosya verilerini işler. Aracı, birbiriyle iletişim kurarak dosya verilerini içeren iletilerin akışını düzenler. Bu, dosya verilerinin IBM MQ iletim kuyruklarında oluşturulmasını önler ve kalıcı olmayan iletilerden herhangi birinin teslim edilmemesi durumunda dosya verilerinin yeniden gönderilmesini sağlar.
- Managed File Transfer aracıları bir dizi IBM MQ kuyruğu kullanır. Daha fazla bilgi için bkz. MFT sistem kuyrukları ve sistem konusu.
- Bu kuyruklardan 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 aracıya IBM MQ Explorer eklenti IBM MQ iletileri gönderir ve aracıya istenen eylemi gerçekleştirmesi için talimat verir. Aracıyla bu şekilde etkileşimde bulunan IBM MQ uygulamaları yazabilirsiniz. Daha fazla bilgi için, iletileri aracı komut kuyruğuna koyarak Denetleme MFT konusuna bakın.
- Managed File Transfer aracıları, koordinasyon kuyruk yöneticisi olarak belirlenmiş bir MQ kuyruk yöneticisine aktarımların ilerleyişi ve sonucu hakkında bilgi gönderir. Bu bilgiler, eşgüdüm kuyruk yöneticisi tarafından yayınlanır ve aktarımın ilerleyişini izlemek ya da gerçekleşen aktarımların kayıtlarını tutmak isteyen uygulamalar tarafından abone olunabilir. Hem komut satırı komutları, hem de IBM MQ Explorer eklentisi yayınlanan bilgileri kullanabilir. Bu bilgileri kullanan IBM MQ uygulamaları yazabilirsiniz. Bilgilerin yayınlandığı konuyla ilgili daha fazla bilgi için bkz. SYSTEM.FTE konu.
- Managed File Transfer ' in temel bileşenleri, IBM MQ kuyruk yöneticilerinin iletileri saklama ve iletme yeteneğinden yararlanır. Bu, bir kesinti yaşamanız durumunda altyapınızın etkilenmeyen bölümlerinin dosya aktarmaya devam edebileceği anlamına gelir. Bu, koordinasyon kuyruk yöneticisine kadar uzanır; saklama ve iletme ve sürekli aboneliklerin birleşimi, koordinasyon kuyruk yöneticisinin, gerçekleşen dosya aktarımlarına ilişkin anahtar bilgilerini kaybetmeden kullanılamaz duruma gelmesine izin verir.

MFT topolojiye genel bakış

Managed File Transfer aracılarının bir IBM MQ ağındaki koordinasyon kuyruk yöneticisine nasıl bağlandığına ilişkin genel bakış.

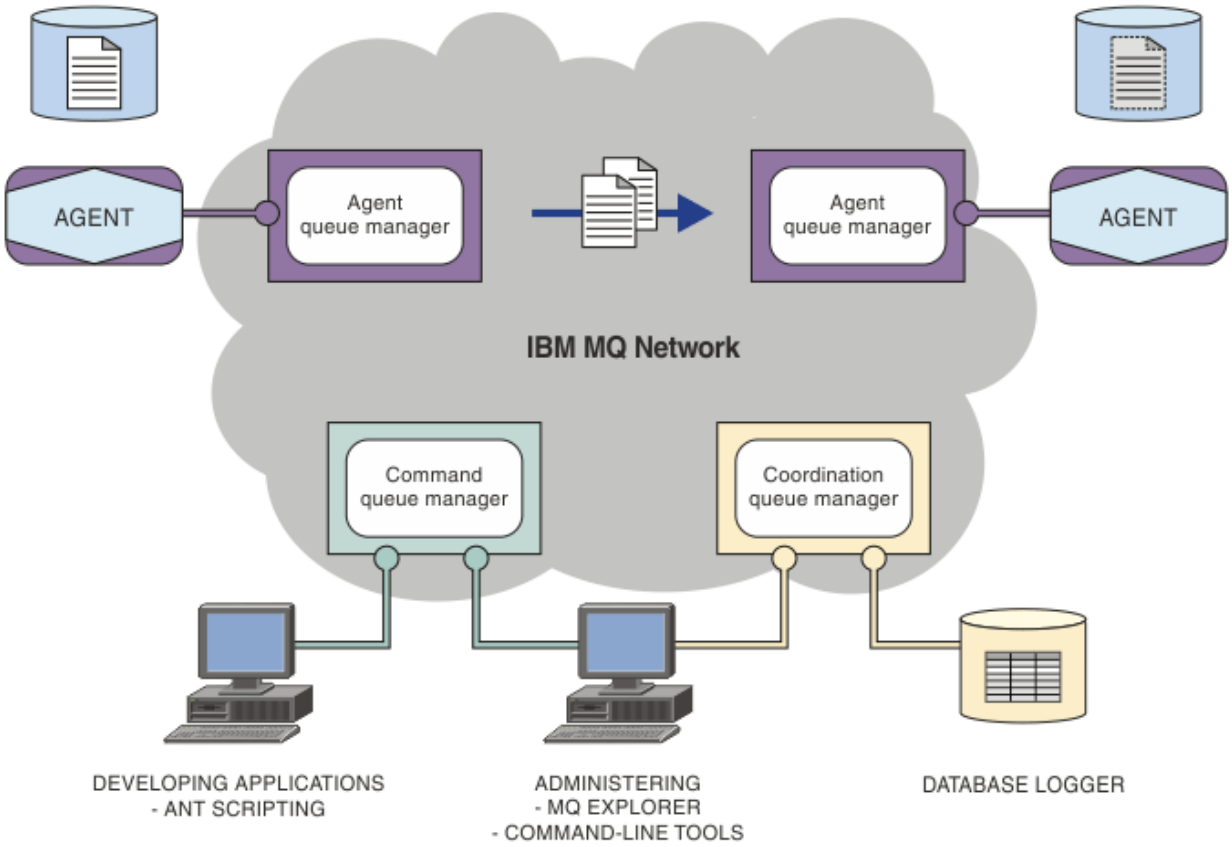
Managed File Transfer aracıları, aktarılan dosyaları gönderir ve alır. Her aracının ilişkili kuyruk yöneticisinde kendi kuyruk kümesi vardır ve aracı, bağ tanımlarında ya da istemci kipinde kuyruk yöneticisine bağlanır. Bir aracı, eşgüdüm kuyruk yöneticisini kuyruk yöneticisi olarak da kullanabilir.

Koordinasyon kuyruk yöneticisi, denetim ve dosya aktarımı bilgilerini yayınlar. Koordinasyon kuyruk yöneticisi, aracı, aktarım durumu ve aktarım denetim bilgilerinin toplanması için tek bir noktayı temsil eder. Aktarımların gerçekleşebilmesi için koordinasyon kuyruk yöneticisinin kullanılabilir olması gerekmez. Eşgüdüm kuyruk yöneticisi geçici olarak kullanılamaz duruma gelirse, aktarımlar olağan şekilde devam eder. Denetim ve durum iletileri, koordinasyon kuyruk yöneticisi kullanılabilir oluncaya kadar aracı kuyruk yöneticilerine saklanır ve daha sonra normal olarak işlenebilir.

Aracılar eşgüdüm kuyruk yöneticisine kaydolur ve ayrıntılarını o kuyruk yöneticisine yayınlar. Bu aracı bilgileri, IBM MQ Explorer' den gelen aktarımların başlatılmasını etkinleştirmek için Managed File Transfer eklentisi tarafından kullanılır. Koordinasyon kuyruk yöneticisinde toplanan aracı bilgileri, aracı bilgilerini ve aracı durumunu görüntülemek için komutlar tarafından da kullanılır.

Aktarma durumu ve aktarma denetimi bilgileri koordinasyon kuyruk yöneticisinde yayınlanır. Aktarma durumu ve aktarma denetimi bilgileri, Managed File Transfer eklentisi tarafından IBM MQ Explorer' den gelen aktarımların ilerleyişini izlemek için kullanılır. Koordinasyon kuyruk yöneticisinde saklanan aktarım denetimi bilgileri, denetlenebilirlik sağlamak için saklanabilir.

Komut kuyruğu yöneticisi, IBM MQ ağına bağlanmak için kullanılır ve Managed File Transfer komutlarını verdiğinizde bağlantı kurulan kuyruk yöneticisidir.



İlgili kavramlar

"Managed File Transfer" sayfa 279

Managed File Transfer , dosya boyutundan ya da kullanılan işletim sistemlerinden bağımsız olarak, yönetilen ve denetlenebilir bir şekilde sistemler arasında dosya aktarır.

“MFT IBM MQile nasıl çalışır?” sayfa 281
Managed File Transfer , IBM MQile çeşitli şekillerde etkileşim kurmaktadır.

Managed File Transfer Senaryo

MFTREST API'e genel bakış

REST API , aktarımların listelenmesi ve dosya aktarma aracılarıyla ilgili ayrıntılar da içinde olmak üzere belirli Managed File Transfer komutlarını destekler.

IBM MQ 9.1.0içinden 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), İnternet üzerinden uzak siteler arasında ileti sistemi çözümlerini uygulamak için kullanılabilecek isteğe bağlı bir IBM MQ bileşenidir.

IBM MQ 9.2.0'den MQIPT , IBM MQ' in isteğe bağlı bir bileşenidir. IBM MQ 9.3.xiçin MQIPT kuruluş dosyalarını edinmek üzere IBM MQ için IBM Fix Centraladresine gidin. IBM MQ 9.2.0' den önce MQIPT , bir destek paketi olarak mevcuttu.

IBM MQ 9.3içinde MQIPT ' yi kullanmak için IBM MQ 9.3 komutunu çalıştırmanız gerekmez. Desteklenen herhangi bir IBM MQsürümünü bağlamak için MQIPT kullanabilirsiniz ve MQIPTile aynı sürümde başka bir IBM MQ bileşeni kurmanız gerekmez.

IBM MQ yetkisi satın aldıysanız, gereken sayıda MQIPTkopyasını kurabilirsiniz. MQIPT kuruluşları, satın aldığınız IBM MQ yetki kapsamınıza dahil değildir. IBM MQ lisanslaması hakkında daha fazla bilgi için bkz. IBM MQ lisans bilgileri.

Not: Bu belge, IBM MQ 9.3içindeki 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: MQIPT 2.1 ya da önceki bir sürümünü kullanıyorsanız, MQIPT destek paketinin destek tarihinin sonu 30th olduğu için, IBM MQ 9.3için MQIPT sürümüne yükseltmeniz önerilir.

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 olmadığında bu bağlantıyı etkinleştirir.

Bir ya da daha çok MQIPT eşgörünümü, iki IBM MQ kuyruk yöneticisi arasındaki iletişim yoluna ya da bir IBM MQ istemcisi ile bir IBM MQ kuyruk yöneticisi arasına yerleştirilebilir. MQIPT yönetim ortamları, iki sistem arasında doğrudan TCP/IP bağlantısına gerek kalmadan iki IBM MQ sisteminin ileti değiş tokuşuna izin verir. Bu, güvenlik duvarı yapılandırması iki sistem arasında doğrudan TCP/IP bağlantısını engellerse kullanışlıdır.

MQIPT , gelen bağlantılar için bir ya da daha çok TCP/IP kapısını dinler; bu kapılar olağan IBM MQ iletileri, IBM MQ HTTPiçinde tünellenmiş iletileri ya da TLS (Transport Layer Security; İletim Katmanı Güvenliği) ya da SSL (Secure Sockets Layer; Güvenli Yuva Katmanı) kullanılarak şifrelenmiş iletileri taşıyabilir. MQIPT birden çok eşzamanlı bağlantıyı işleyebilir.

İlk TCP/IP bağlantı isteğini yapan IBM MQ kanalı, *çağıran*, *yanıtlayıcı*olarak bağlanmaya çalıştığı kanal ve sonuçta *hedef kuyruk yöneticisi*olarak iletişim kurmaya çalıştığı kuyruk yöneticisi olarak adlandırılır.

MQIPT , verileri kaynağından hedefine iletirken bellekte tutar. Diskte veri saklanmaz (işletim sistemi tarafından diske sayfalanan bellek dışında). MQIPT ' in diske belirtik olarak eriştiği tek zaman, diskin konfigürasyon dosyasını okumak ve bağlantı günlüğü ve izleme kayıtlarını yazmaktır.

IBM MQ kanal tiplerinin tam aralığı, bir ya da daha fazla MQIPTörneği aracılığıyla yapılabilir. Bir iletişim yolunda MQIPT öğesinin bulunması, bağlı IBM MQ bileşenlerinin işlevsel özelliklerini etkilemez, ancak ileti aktarımının başarımı üzerinde bir etkisi olabilir.

MQIPT , “Olası MQIPT yapılandırmaları” sayfa 287içinde açıklandığı gibi IBM MQ ve IBM Integration Busile birlikte kullanılabilir.

MQIPTürününü kurmak için bkz. MQIPTürününü kurma.

İlgili görevler

yapılandırmaIBM MQ Internet Pass-Thru

IBM MQ Internet Pass-Thru ' nin yönetilmesi ve yapılandırılması

İlgili başvurular

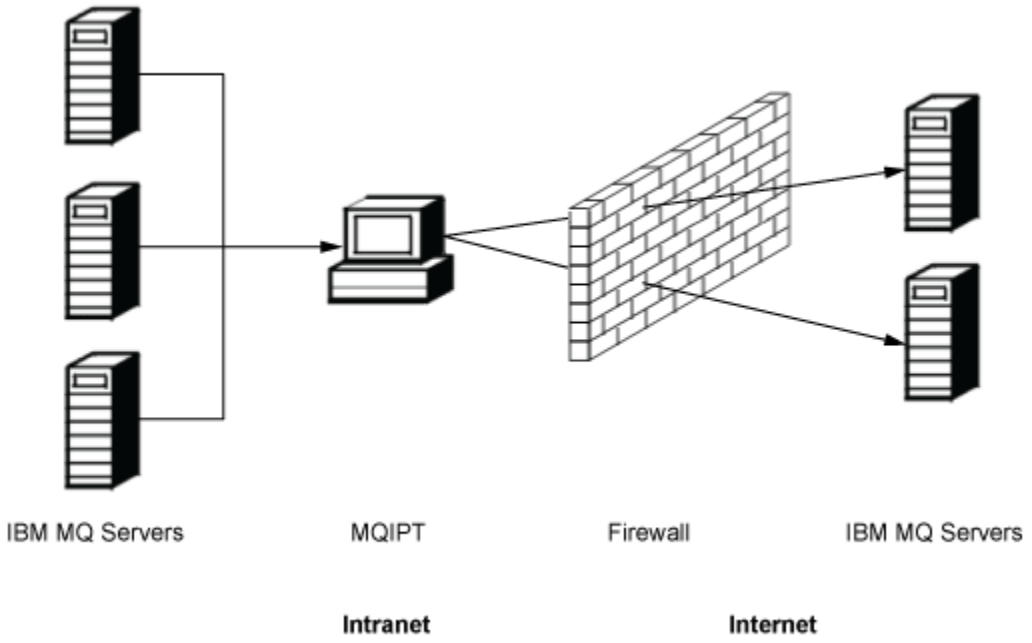
IBM MQ Internet Pass-Thru yapılandırma başvurusu

MQIPT kullanımı

IBM MQ Internet Pass-Thru (MQIPT) için bazı olası kullanımlar vardır.

MQIPT , kanal yoğunlaştırıcı olarak kullanılabilir

MQIPT komutunu bu şekilde kullanarak, birden çok ayrı ana makineden gelen ya da giden kanallar, bir güvenlik duvarına MQIPT ana makinesinden gelen ya da bu anasistemden gelen tüm kanallar gibi görünebilir. Bu, güvenlik duvarı süzme kurallarının tanımlanmasını ve yönetilmesini kolaylaştırır.

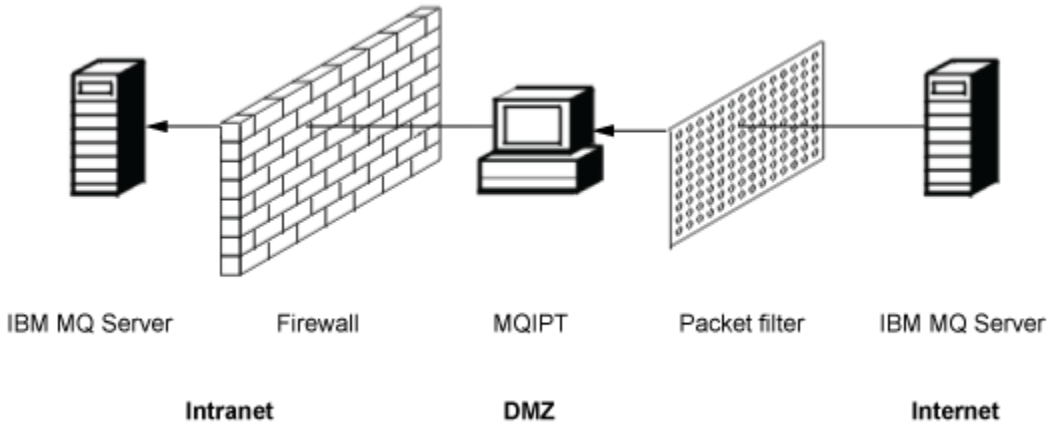


Şekil 79. Kanal yoğunlaştırıcı olarak MQIPT örneği

MQIPT , tek bir erişim noktası sağlamak için DMZ ' ye yerleştirilebilir

MQIPT DMZ güvenlik duvarına (yerel alan ağlarının güvenliğini sağlamak için bir güvenlik duvarı yapılandırması) yerleştirilirse, bilinen ve güvenilir İnternet iletişim kuralı (IP) adresine sahip bir bilgisayarda MQIPT , gelen IBM MQ kanal bağlantılarını dinlemek için kullanılabilir; iç güvenlik duvarı bu güvenilir bilgisayarın gelen bağlantıları yapmasına izin vermelidir. Bu yapılandırmada MQIPT , dış erişim isteklerinin güvenilir intranet içindeki bilgisayarların gerçek IP adreslerini almasını önler. Bu şekilde, MQIPT tek bir erişim noktası sağlar. Gerekirse, MQIPT TLS bağlantılarını kabul edecek ve verileri ayrı bir

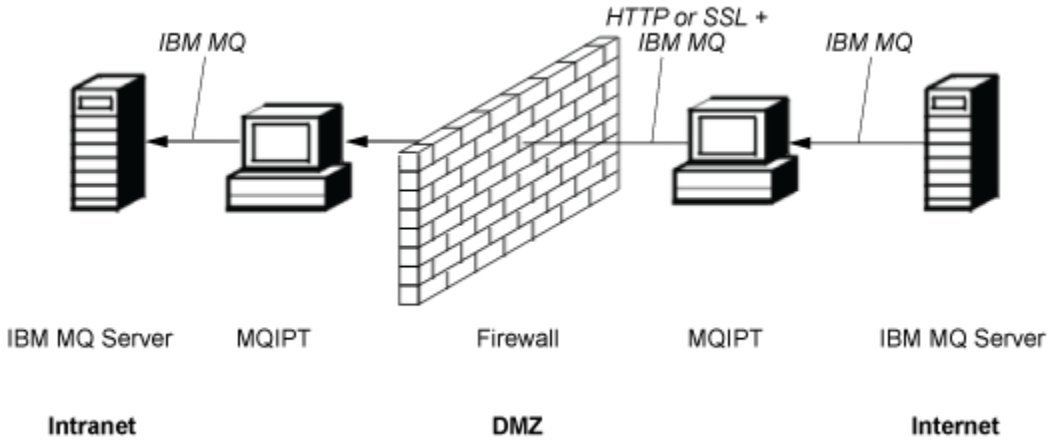
TLS bağlantısı kullanarak hedefe iletebilecek şekilde yapılandırılabilir, bu nedenle DMZ ' de TLS oturumu sonlandırılır.



Şekil 80. DMZ güvenlik duvarında MQIPT örneği

MQIPT , HTTP tüneli aracılığıyla iletişim kurabilir

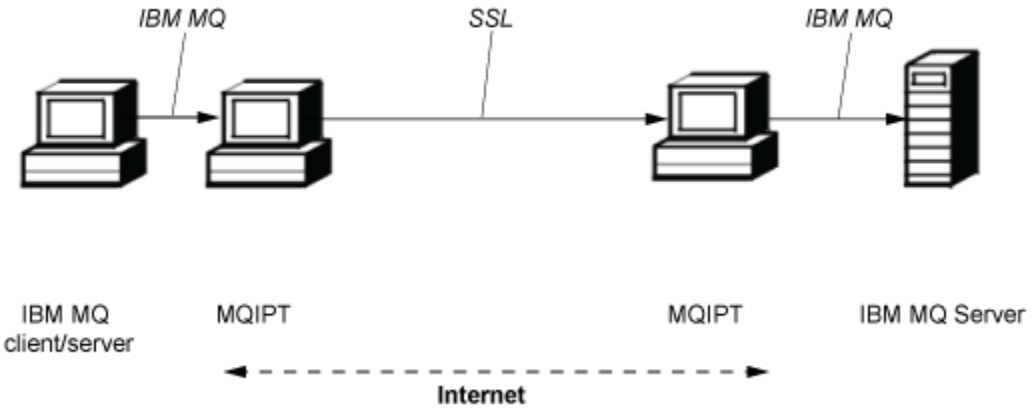
İki MQIPT eşgörünümü devreye alınırsa, HTTPkullanarak iletişim kurabilirler. HTTP tünel oluşturma özelliği, isteklerin var olan HTTP yetkili sunucularının kullanımıyla güvenlik duvarları üzerinden iletilmesini sağlar. İlk MQIPT , IBM MQ iletişim kuralını HTTP ' ye ekler ve ikinci, IBM MQ iletişim kuralını HTTP sarıcısından çıkarır ve hedef kuyruk yöneticisine iletir.



Şekil 81. MQIPT ve HTTP tünelleme örneği

MQIPT iletileri şifreleyebilir

MQIPT önceki örnekte olduğu gibi yapılandırıldıysa, istekler güvenlik duvarlarından iletilmeden önce şifrelenebilir. İlk MQIPT , verileri şifreler ve ikinci, hedef kuyruk yöneticisine göndermeden önce SSL/TLS kullanarak verilerin şifresini çözer.



Şekil 82. MQIPT ve SSL/TLS örneği

MQIPT nasıl çalışır

En basit yapılandırmasında MQIPT , bir IBM MQ iletişim kuralı iletici görevi görür. Bir TCP/IP kapısını dinler ve IBM MQ kanallarından gelen bağlantı isteklerini kabul eder.

Doğru biçimli bir istek alınırsa, MQIPT kendisi ile hedef IBM MQ kuyruk yöneticisi arasında başka bir TCP/IP bağlantısı kurar. Daha sonra, gelen bağlantıdan aldığı tüm protokol paketlerini hedef kuyruk yöneticisine geçirir ve hedef kuyruk yöneticisinden gelen protokol paketlerini özgün gelen bağlantıya geri döndürür.

IBM MQ iletişim kuralında (istemci/sunucu ya da kuyruk yöneticisinden kuyruk yöneticisine) herhangi bir değişiklik yoktur; her iki uç da aracı varlığından doğrudan haberdardır. IBM MQ istemcisi ya da sunucu kodunun yeni sürümleri gerekli değildir.

MQIPTkomutunu 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** özelliğiyle tanımlanır. MQIPT gelen verileri okur ve hedef kuyruk yöneticisine aktarır. İstemci/sunucu kanalındaki kullanıcı kimliği ve parola gibi diğer yapılandırma alanları da hedef kuyruk yöneticisine benzer şekilde geçirilir.

Birden çok kuyruk yöneticisi

MQIPT , birden çok hedef kuyruk yöneticisine erişime izin vermek için kullanılabilir. Bunun işe yaraması için, MQIPT ' e hangi kuyruk yöneticisine bağlanacağınızı MQIPT söyleyecek bir mekanizma olması gerekir; bu nedenle, hangi kuyruk yöneticisine bağlanacağınızı belirlemek için gelen TCP/IP kapı numarasını kullanır.

Bu nedenle, MQIPT ' yi birden çok TCP/IP kapısında dinleyecek şekilde yapılandırabilirsiniz. Her dinleme kapısı, MQIPT *rotası*aracılığıyla bir hedef kuyruk yöneticisiyle eşlenir. Dinleyen bir TCP/IP kapısını hedef kuyruk yöneticisinin anasistem adı ve kapısıyla ilişkilendirebilecek 100 'e kadar rota tanımlayabilirsiniz. Bu, hedef kuyruk yöneticisinin anasistem adının (IP adresi) kaynak kanal tarafından hiçbir zaman görünmeyeceği anlamına gelir. Her rota, her biri bağımsız olarak hareket eden, dinleme kapısı ve hedefi arasındaki birden çok bağlantıyı işleyebilir.

MQIPT yapılandırma dosyası

MQIPT , `mqipt.conf`adlı bir yapılandırma dosyası kullanır. Bu dosya, tüm rotaların tanımlarını ve ilişkili özelliklerini içerir. `mqipt.conf`hakkında daha fazla bilgi için bkz. Administering and configuring IBM MQ Internet Pass-Thru .

MQIPT başlatıldığında, yapılandırma dosyasında listelenen her rotayı başlatır. İletiler, her rotanın durumunu gösteren sistem konsoluna yazılır. Bir rota için MQCPI078 iletisi gösterildiğinde, o rota bağlantı isteklerini kabul etmeye hazırdır.

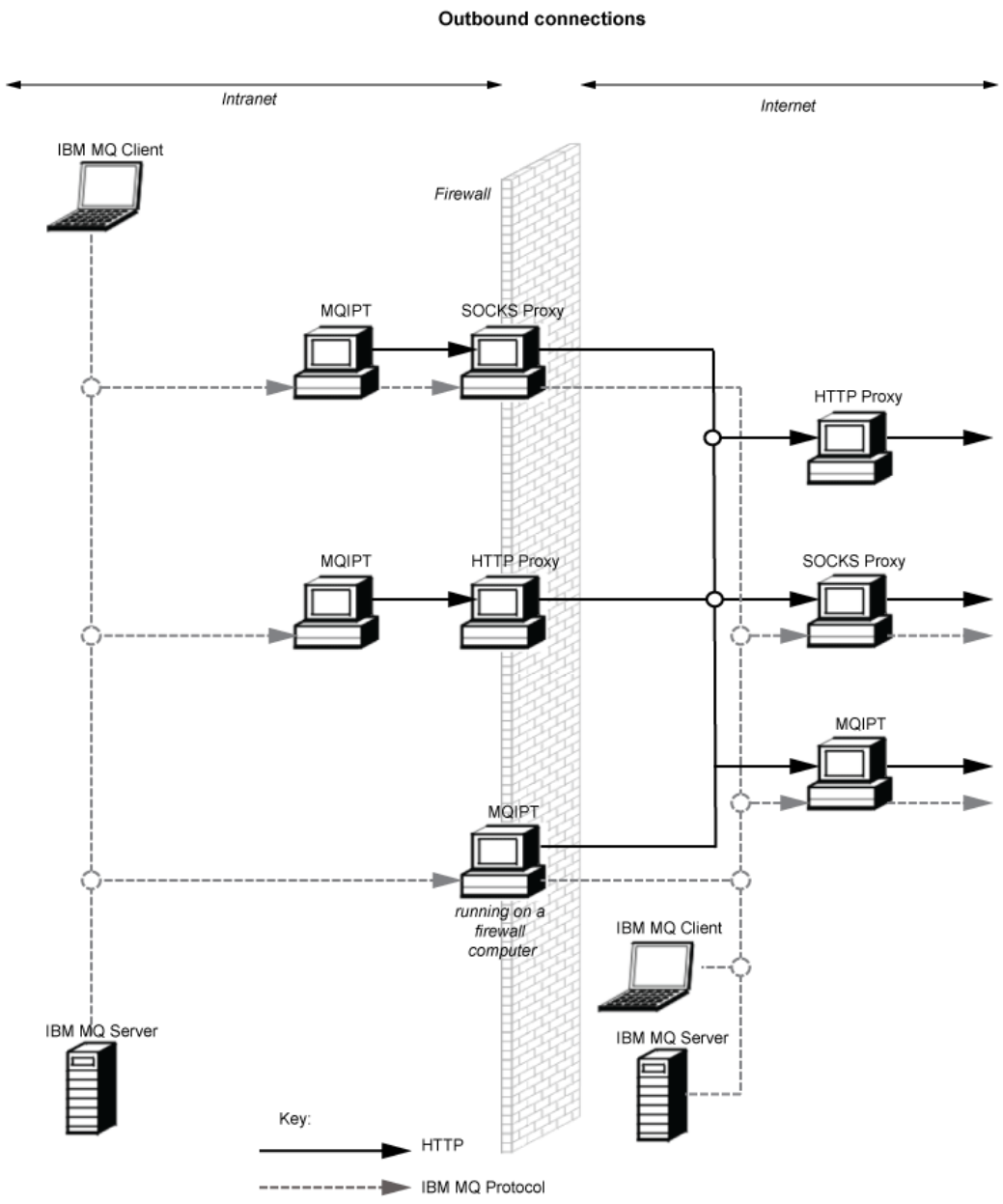
Olası MQIPT yapılandırmaları

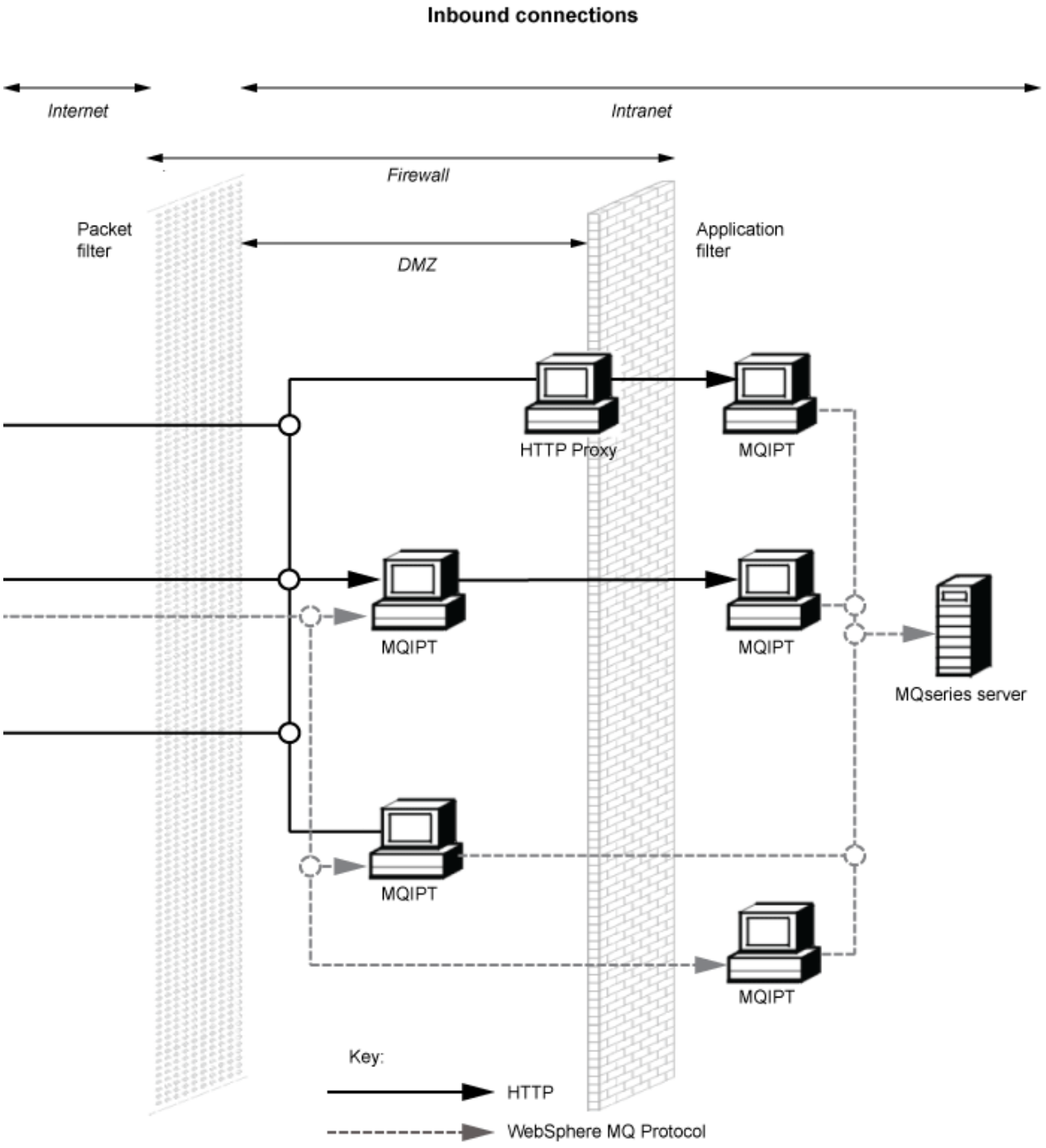
MQIPT , IBM MQ ve IBM Integration Busile birlikte kullanılabilir.

Aşağıdaki çok parçalı şekilde, IBM MQ topolojisinde MQIPT için olası yapılandırmaların çoğu gösterilmektedir. MQIPT ' in ileti gönderebildiği farklı yolları gösterir. İntranet üzerinde, güvenlik duvarının içinde ve İnternet 'te güvenlik duvarının dışında bulunan istemcileri ve sunucuları gösterir ve iletileri MQIPT, HTTP yetkili sunucusuna ya da SOCKS yetkili sunucusuna iletir.

İletiler, iletiyi gelen güvenlik duvarından sunucuya geçirmeden önce DMZ ' deki bir MQIPT yetkili sunucusu ya da HTTP yetkili sunucusu tarafından alınır.

Güvenlik duvarının intranet tarafındaki HTTP yetkili sunucusu, SOCKS yetkili sunucusu ve MQIPT bilgisayarlarının, İnternet 'te birbirine zincirlenmiş birden çok bilgisayar olasılığını temsil ettiğini unutmayın. Örneğin, bir MQIPT bilgisayarı, hedefine ulaşmadan önce bir ya da daha fazla SOCKS ya da HTTP yetkili bilgisayarı ya da daha fazla MQIPT bilgisayar 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 istemcisinin ya da kuyruk yöneticisinin MQIPTile iletişim kurduğu uyumlu bağlantı senaryoları. Hedef kuyruk yöneticisiyle iletişim kurmak için aynı ya da ikinci bir MQIPT rotası kullanılır.

Tek bir MQIPT rotasına sahip uyumlu yapılandırmalar

IBM MQile iletişim kurmak için tek bir MQIPT rotası kullanabilirsiniz.

Çizelge 26 sayfa 290 içindeki sütunlar aşağıdaki bilgileri içerir:

1. IBM MQ ile MQIPT rotası arasında kullanılan iletişim kuralı. Bağlantı bir IBM MQ istemcisi ya da kuyruk yöneticisi tarafından oluşturulabilir ve IBM MQ Formats and Protocols (FAP) ya da SSL/TLS iletişim kuralını kullanabilir.
2. MQIPT rotasının çalıştığı kip. Internet üzerinden MQIPT ve IBM MQarasındaki iletişimin biçimi, MQIPT rotasının yapılandırmasına göre belirlenir. Tablonun SSL 'den bahsettiği yerde TLS' yi de kullanabildiğinizi unutmayın.
3. MQIPT rotası ile hedef kuyruk yöneticisi arasında kullanılan protokol.

Çizelge 26. Tek bir MQIPT örneğine sahip geçerli yapılandırmalar

1. IBM MQ kaynak iletişim kuralı	**2. MQIPT rotasının kipi**	**3. IBM MQ hedef iletişim kuralı**
FAP	FAP-yetkili sunucu (varsayılan)	FAP
	FAP-sunucu ve SSL-istemci	SSL/TLS
SSL/TLS	SSL yetkili sunucusu	SSL/TLS
	SSL sunucusu ve FAP istemcisi	FAP
	SSL sunucusu ve SSL istemcisi	SSL/TLS

Birden çok MQIPT rotasına sahip uyumlu yapılandırmalar

IBM MQile iletişim kurmak için bir ya da daha çok MQIPTörneğinin üzerinde birden çok rota kullanmayı seçebilirsiniz.

Çizelge 27 sayfa 291 içindeki sütunlar aşağıdaki bilgileri içerir:

1. IBM MQ ile ilk MQIPT rotası arasında kullanılan iletişim kuralı. Bağlantı bir IBM MQ istemcisi ya da kuyruk yöneticisi tarafından oluşturulabilir ve IBM MQ Formats and Protocols (FAP) ya da SSL/TLS iletişim kuralını kullanabilir.
2. İlk MQIPT rotasının çalıştığı kip. Internet üzerinden MQIPT ve IBM MQarasındaki iletişimin biçimi, MQIPT rotasının yapılandırmasına göre belirlenir. Tablonun SSL 'den bahsettiği yerde TLS' yi de kullanabildiğinizi unutmayın.
3. İkinci MQIPT rotasının çalıştığı kip.
4. İkinci MQIPT rotası ile hedef kuyruk yöneticisi arasında kullanılan protokol.

Çizelge 27. Birden çok MQIPT eşgörünümü olan 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)	FAP
	FAP-sunucu ve SSL-istemci	SSL yetkili sunucusu	SSL/TLS
		SSL sunucusu ve FAP istemcisi	FAP
		SSL sunucusu ve SSL istemcisi	SSL/TLS
	HTTP-client	HTTP-server ve SSL-istemci	SSL/TLS
	HTTPS-istemci	HTTPS-sunucu ve SSL-istemci	SSL/TLS
	HTTP-client	HTTP-server	FAP
	HTTPS-istemci	HTTPS-sunucu	FAP
SSL/TLS	SSL yetkili sunucusu	SSL yetkili sunucusu	SSL/TLS
		SSL sunucusu ve FAP istemcisi	FAP
		SSL sunucusu ve SSL istemcisi	SSL/TLS
	HTTP-client	HTTP-server	FAP
	HTTPS-istemci	HTTPS-sunucu	SSL/TLS
	HTTP-client	HTTP-server ve SSL-istemci	FAP
	HTTPS-istemci	HTTPS-sunucu ve SSL-istemci	SSL/TLS

Desteklenen kanal yapılandırmaları

Tüm IBM MQ kanal tipleri desteklenir, ancak yapılandırma TCP/IP bağlantılarıyla sınırlıdır. Bir IBM MQ istemcisinde ya da kuyruk yöneticisinde, MQIPT hedef kuyruk yöneticisi gibi görünür. Kanal yapılandırması bir hedef anasistem ve kapı numarası gerektirdiğinde, MQIPT anasistem adı ve dinleyici kapı numarası belirtilir.

İstemci/sunucu kanalları

MQIPT , gelen istemci bağlantısı isteklerini dinler ve bunları HTTP tüneli oluşturma, SSL/TLS ya da standart IBM MQ iletişim kuralı paketleri olarak kullanarak iletir. MQIPT , HTTP tüneli ya da SSL/TLS kullanıyorsa, bunları bir bağlantıda ikinci bir MQIPT' e iletir. HTTP tüneli kullanmıyorsa, hedef kuyruk yöneticisi olarak gördüğü bir bağlantıya iletir (ancak bu işlem başka bir MQIPTolabilir). Hedef kuyruk yöneticisi istemci bağlantısını kabul ettiğinde, paketler istemci ile sunucu arasında aktarılır.

Küme gönderen/alıcı kanalları

MQIPT bir küme-gönderen kanalından gelen bir istek alırsa, kuyruk yöneticisinin SOCKS etkin olduğunu ve SOCKS el sallama işlemi sırasında gerçek hedef adresinin alınacağını varsayar. İsteği sonraki MQIPT ya da hedef kuyruk yöneticisine, istemci bağlantı kanalları ile aynı şekilde iletir. Bu, otomatik olarak tanımlanan küme gönderen kanallarını da içerir.

Gönderen/Alıcı
MQIPT bir gönderen kanalından gelen bir istek alırsa, bu isteği istemci bağlantı kanalları ile aynı şekilde bir sonraki MQIPT ya da hedef kuyruk yöneticisine iletir. Hedef kuyruk yöneticisi gelen isteği doğrular ve uygunsa alıcı kanalı başlatır. Gönderen ve alıcı kanal (güvenlik akışları dahil) arasındaki tüm iletişim aktarılır.

İstekte bulunan/sunucu
Bu birleşim, önceki yapılandırmalarla aynı şekilde işlenir. Bağlantı isteğini doğrulama, hedef kuyruk yöneticisindeki sunucu kanalı tarafından gerçekleştirilir.

İstekte bulunan/gönderen
İki kuyruk yöneticisinin birbiriyle doğrudan bağlantı kurmasına izin verilmezse, ancak her ikisinin de MQIPT ile bağlantı kurmasına ve bu yapılandırmadan gelen bağlantıları kabul etmesine izin verilirse, "geri çağırma" yapılandırması kullanılabilir.

Sunucu/istekte bulunan ve sunucu/alıcı
Bunlar MQIPT tarafından, `Sender/Receiver` yapılandırmasını işlediği şekilde işlenir.

Kanal sonlandırma ve arıza koşulları

MQIPT , bir IBM MQ kanalının kapanışını (olağan ya da olağandışı) algıladığında, kanal kapanışı yayılır. MQIPTkomutunu kullanarak bir rotayı kapatırsanız, o rotadan geçen tüm kanallar kapanır.

MQIPT , isteğe bağlı bir boşta durma zamanaşımı olanağı sağlar. MQIPT , bir kanalın zamanaşımını aşan bir süre boyunca boşta olduğunu algılarsa, söz konusu iki bağlantıda anında kapatma işlemi gerçekleştirir.

Kanalın her iki ucundaki IBM MQ sistemleri, bu olağandışı kapatma koşullarını ağ arızaları olarak ya da ortak tarafından kanalın sonlandırılması olarak gözlemler. Kanal, MQIPT kullanılmıyormuş gibi yeniden başlatılabilir ve kurtarılabilir (protokol belirsiz bir süre boyunca hata oluşursa).

İletilerin güvenliği

IBM MQ dağıtılmış kuyruk yönetimi, iletilerin düzgün bir şekilde teslim edilmesini sağlar. Bu durum, kanalın iki ucu arasında MQIPT olduğunda da devam etmektedir. MQIPT , doğru ileti teslimini sağlayan herhangi bir ileti verisi saklamaz ya da eşitleme noktası yordamına katılmaz.

Hızlı, kalıcı olmayan IBM MQ iletileri kullanılırken, MQIPT rotası başarısız olursa ya da bir IBM MQ iletisi hareket halindeyken 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 eşgörünümlü kuyruk yöneticileri ve yüksek kullanılabilirlik

MQIPT , yüksek kullanılabilirlikli ortamlarda çok eşgörünümlü kuyruk yöneticileriyle birlikte kullanılabilir.

MQIPT 'in kalıcı durumu yoktur ve bu nedenle MQIPT ' in başka bir sisteme geçilmesinde herhangi bir yarar yoktur. Bunun yerine, farklı sistemlerde çalışan aynı `mqipt.conf` yapılandırma dosyalarıyla birden çok MQIPT eşgörünümü vardır. MQIPT ' in her bir eşgörünümünün kullanılabilirliğini izleyin ve gerekirse yeniden başlatın (aynı sistemde). Bu, bağlantıları yönlendirmek için kullanılabilecek aynı MQIPT yönetim ortamlarını sağlar. Daha sonra IBM MQ 'in bağlantıları MQIPT ' e yöneltebildiğinden ve MQIPT ' in bu bağlantıları hedef kuyruk yöneticisine iletebildiğinden emin olmanız gerekir.

Giden IBM MQ kanalları, çeşitli şekillerde kullanılabilir bir MQIPT eşgörünümüne yönlendirilebilir; örneğin:

- WebSphere Edge Components ürününden IBM Network Dispatcher gibi bir yük dengeleyici ya da yüksek kullanılabilirlikli yönlendirici kullanın.
- Virgülle ayrılmış bir liste kullanarak IBM MQ kanal tanımında birden çok bağlantı adı belirtin. IBM MQ daha sonra, kullanılabilir bir MQIPT yönetim ortamı buluncaya kadar her bir MQIPT adresine sırayla bağlanmayı dener.

Ayrıca, bağlantıları MQIPT ' den hedef kuyruk yöneticisine yönlendirmeniz gerekir. Yüksek kullanılabilirlikli yapılandırma, IP adresinin hedef kuyruk yöneticisiyle birlikte geçişinin başarısız olmasını sağladıysa, özel

bir MQIPT yapılandırması gerekmez: **Destination** rota özelliğinde hedef IP adresini belirtin ve hata durumunda yedek sisteme geçiş işleminin kuyruk yöneticisiyle birlikte IP adresini taşımasına izin verin.

Ancak, hata durumunda yedek sisteme geçiş işleminden sonra kuyruk yöneticisinin IP adresi değişirse, MQIPT ' in bağlantıyı doğru hedefe iletmesini ayarlamanız gerekir. Bu birkaç şekilde yapılabilir:

- Hangi IP adresi ve kapı numarasının erişilebilir olduğunu denetleyen bir yöneltme çıkışı yazın ve her bağlantı için rota hedefini geçersiz kılın. Bazı örnek yöneltme çıkışları MQIPTile birlikte sağlanır; bunlar bu amaçla uyarlanabilir.
- Bağlantıyı yeniden yönlendirmek için yüksek kullanılabilirlikli bir yük dengeleyici kullanın.
- Kuyruk yöneticisinin çalışabileceği her bir IP adresi ve kapı için bir tane olacak şekilde birden çok MQIPT rotası tanımlayın. Daha sonra IBM MQ bağlantılarını çeşitli MQIPT rotalarına yönlendirin; örneğin, giden kanalın bağlantı adında tüm rota IP adreslerini ve kapı numaralarını virgülle ayrılmış bir listede listeleyerek.

Ağ yolundaki tüm uçtan uca bileşenlerin ayarlanması da önemlidir:

1. Yeniden bağlanma girişimlerinin ilk kullanılabilir hedefe geçebilmesi için, kullanılamayan sistemlere yönelik bağlantı girişimlerinin hemen başarısız olması gerekir.

   MQIPT SSL rotaları için, **SSLClientConnectTimeout** rota özelliğini, kullanılamayan hedefler için bilgi istemi bağlantısı arızası olduğundan emin olmak üzere ayarlayın. IBM MQ ayarlama parametrelerine ilişkin ayrıntılar için IBM MQ belgelerine bakın. İşletim sistemine ilişkin TCP/IP ayarlarına ilişkin ayrıntılar için işletim sisteminizin belgelerine bakın. Her durumda, başarısız bağlantı girişimlerinin hızlı bir şekilde bir ağ hatası döndürmesi (örneğin, TCP ilk duruma getirme paketi) ya da zamandışı gecikme olmadan zamandışı bırakması gerekir.
2. Yeni bağlantıların kurulabilmesi için, arızalı bir sisteme yapılan etkin bağlantıların hemen kesilmiş olması gerekir.

   Ayrıca, bağlantıların MQIPT' i etkin olarak kullandığı bir anda yedek sisteme geçiş etkisini de göz önünde bulundurmanız gerekir. Hata durumunda yedek sisteme geçiş sırasında ağ bağlantılarının kesilmesi olasıdır. İstemci uygulamaları için, IBM MQ otomatik istemci yeniden bağlanma özelliğini kullanarak kesilmiş bağlantıları yeniden kurabilirsiniz. İleti kanalları için, kanalın hemen yeniden bağlanması için kısa bir yeniden deneme aralığı belirtebilirsiniz. Otomatik istemci yeniden bağlantısı ve ileti kanalı yeniden deneme yapılandırmasıyla ilgili ek bilgi için IBM MQ belgelerine bakın.

V 9.3.5 IBM MQ Console ve REST API

HTTPkullanarak IBM MQ'yi yönetmek ve ileti alışverişi işlemleri gerçekleştirmek için IBM MQ Console ve REST API ' yi kullanabilirsiniz.

- Bir web tarayıcısından temel yönetim görevlerini gerçekleştirmek için IBM MQ Console olanağını kullanabilirsiniz. Daha fazla bilgi için bkz. Administration using IBM MQ Console.
- Kuyruk yöneticileri ve kuyrukları, Managed File Transfer aracıları ve aktarımları gibi IBM MQ nesnelerini yönetmek için administrative REST API olanağını kullanabilirsiniz. Daha fazla bilgi için bkz. Administration using REST API.
- Basit noktadan noktaya iletişim ve yayınlama ileti sistemi gerçekleştirmek için messaging REST API olanağını kullanabilirsiniz. Daha fazla bilgi için bkz. Messaging using the REST API.

Kuruluş seçenekleri

IBM MQ Console ve REST API , mqweb adı verilen bir WebSphere Liberty sunucusunda çalışır. IBM MQ 9.3.5' den, mqweb sunucusunu IBM MQ kuruluşunda isteğe bağlı bir bileşen olarak ya da bağımsız bir IBM MQ Web Server kuruluşu olarak kurabilirsiniz.

V 9.3.5 Linux Bağımsız IBM MQ Web Server kuruluşu

IBM MQ 9.3.5' den mqweb sunucusu, bağımsız bir IBM MQ Web Serverkuruluşunda çalışabilir. Bağımsız bir IBM MQ Web Server kuruluşu, mqweb sunucusunu IBM MQ kuruluşlarınızdan ayrı sistemlerde kurmanızı ve çalıştırmanızı sağlar. Bağımsız bir IBM MQ Web Server ' in kurulması, mqweb

sunucularınızı çalıştırmak için seçtiğiniz sistemlere ve sistem sayısına ilişkin daha fazla esneklik sağlar. Gerekiyorsa, gereksinim duyduğunuz ölçeklenebilirliği ve kullanılabilirliği sağlamak için mqweb sunucusunun birkaç eşgörünümü farklı makinelerde çalıştırılabilir.

IBM MQ yetkisi satın aldıysanız, bağımsız IBM MQ Web Serveriçin gerektiği kadar kopya kurabilirsiniz. IBM MQ Web Server kuruluşları, satın aldığınız IBM MQ yetki kapsamınıza dahil değildir. IBM MQ lisanslaması hakkında daha fazla bilgi için bkz. IBM MQ lisans bilgileri.

Bağımsız bir IBM MQ Web Server kuruluşunda aşağıdaki kısıtlamalar geçerlidir:

- IBM MQ Console yalnızca uzak kuyruk yöneticilerini denetlemek için kullanılabilir.
- messaging REST API yalnızca uzak kuyruk yöneticileriyle kullanılabilir.
- administrative REST API kullanılamıyor.

Bağımsız IBM MQ Web Server yalnızca Linux platformlarında desteklenir.

Bağımsız IBM MQ Web Serverkuruluşu hakkında daha fazla bilgi için bkz. Bağımsız kuruluşu IBM MQ Web Server.

IBM MQ kuruluşunun isteğe bağlı bileşeni

IBM MQ Console ve REST API bileşenini bir IBM MQ kuruluşunun parçası olarak kurmayı seçebilirsiniz.

Tüm IBM MQ Console ve REST API özellikleri, mqweb sunucusu bir IBM MQ kuruluşunda çalıştığında kullanılabilir.

- IBM MQ Console , yerel ve uzak kuyruk yöneticilerini denetlemek için kullanılabilir.
- messaging REST API , yerel ve uzak kuyruk yöneticileriyle birlikte kullanılabilir.
- administrative REST API , yerel ve uzak kuyruk yöneticilerini denetlemek için kullanılabilir.

IBM MQ Console ve REST API bileşenini kullanmak için IBM MQ kuruluşunuzun bir parçası olarak aşağıdaki bileşeni kurun:

- AIX AIX' da `mqm.web.rte` dosya kümesini kurun.
- IBM i IBM iüzerinde WEB bileşenini kurun.
- Linux Linux' da `MQSeriesWeb` bileşenini kurun.
- Windows Windowsüzerinde, `Web Administration` özelliğini kurun.
- z/OS z/OSüzerinde, `IBM MQ for z/OS UNIX System Services Web Components` özelliğini kurun.

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

İzleyen paragraf, bu tür kayıt ve koşulların, yasalarıyla bağdaşmadığı ülkeler ya da bölgeler için geçerli değildir: IBM BU YAYINI, "OLDUĞU GİBİ" , HİÇBİR KONUDA AÇIK YA DA ÖRTÜK GARANTİ VERMEKSİZİN SAĞLAMAKTADIR; TİCARİ KULLANIMA UYGUNLUK AÇISINDAN HER TÜRLÜ GARANTİ VE BELİRLİ BİR AMACA UYGUNLUK İDDİASI AÇIKÇA REDDEDİLİR. 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 Karayolu 52 N
Rochester, MN 55901
U.S.A.

Bu tür bilgiler, ilgili kayıt ve koşullar altında ve bazı durumlarda bedelli olarak edinilebilir.

© Copyright IBM Corp. 2007, 2024

Bu belgede açıklanan lisanslı program ve bu programla birlikte kullanılabilecek tüm lisanslı malzeme, IBM tarafından IBM Müşteri Sözleşmesi, IBM Uluslararası Program Lisans Sözleşmesi ya da taraflar arasında yapılan herhangi bir eşdeğer sözleşmenin koşulları kapsamında 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

Sağlandıysa, programlama arabirimi bilgileri, bu programla birlikte kullanılmak üzere uygulama yazılımı oluşturmanıza yardımcı olmak amacıyla hazırlanmıştır.

Bu kitapta, müşterinin WebSphere MQhizmetlerini elde etmek üzere program yazmasına olanak sağlayan amaçlanan programlama arabirimlerine ilişkin bilgiler yer alır.

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 bilgiler değişebilir.

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 listesine Web üzerinde "Copyright and trademark information"www.ibm.com/legal/copytrade.shtml(Telif hakkı ve ticari marka bilgileri) başlıklı konudan ulaşılabilir. Diğer ürün ve hizmet adları IBM'in veya diğer şirketlerin ticari markaları olabilir.

Microsoft ve Windows, Microsoft Corporation firmasının ABD'de ve/ya da diğer ülkelerdeki 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: