Upgrading IBM WebSphere MQ

Why Renew / Why Upgrade

Why Upgrade from 5.2 to 5.3

- No support beyond the end of 2003
- SSL - Secure Sockets Layer
- Improved Clustering Capability
- Improved performance
- Enhanced scalability and reliability
- Built in Java™ Messaging Service (JMS) capability

Why Upgrade from 2.1 to 5.3

- No support for 2.1 beyond December 2003.
- 2.1 does not have the capability to share queues.
- 2.1 does not have systems management capability, cannot display queue status.
- 2.1 does not have the ability to monitor queues and check on their status.
- 2.1 has no SSL security.
- 2.1 has no JMS functionality.

Why Upgrade from 1.2 to 5.3

- No support for 1.2 beyond the end of 2003
- 1.2 has no clustering capability
- 1.2 does not have the capability to share queues, therefore no workload balancing.
- 1.2 has no RRS support - Receipt and Returns System
- 1.2 has no Automatic Restart Capability.
- Faster performance

New Release Benefits

- IT Productivity / Reduced cost of Application ownership:
  - Insulates IT professionals from the "plumbing" of communications protocols and operating systems that have to be dealt with in homegrown application integration projects
  - developing, testing and maintenance of such environments can account for 60% of effort in homegrown projects
  - Breadth of platform coverage (39)
  - Rich portfolio of complementary products and services from Business Partners
- Business Flexibility
  - Applications links can be changed quickly and easily in response to changing business needs.
  - Applications run in an asynchronous manner
- Improved Information Management / Process Robustness
  - Assured delivery of information - anywhere in the network
  - Transactional messaging support for coordinated updating of multiple data sources

Security

1. Internet standard for secure communication
   - For protection of both client/server and qmgr/qmgr channels
   - Authentication, encryption, integrity
2. Can configure who is allowed to connect to queue manager
   - For many people, this will remove need for channel exits
3. Shared implementation with MQ for z/OS®
4. Key management
   - Uses OS capabilities on Microsoft Windows, z/OS, OS/400®
   - Uses same store technology as IBM HTTP Server on UNIX® operating systems
5. Configurable per-channel
   - Who can connect
   - Algorithms
6. Configurable per-queue manager
   - Keyring
7. New object-type for CRL access

The API Exit

1. All MQI verbs can be intercepted for management or security tools
   - Encryption
   - Enforcing local MQI rules (eg "no non-persistent messages allowed")
   - Monitoring sizes of messages
2. Initially supported on Solaris (5.2 CSD03)
3. Supported on all distributed platforms with V5.3
4. Multiple exits can be defined and chained
5. Run inside application program

Administration

1. DISPLAY QSTATUS
   - "Who's got the queue open"
   - Already available (V5.2) for z/OS
   - Similar information
     - eg PID, TID, username
     - Channel information
     - Are there uncommitted messages on queue
2. STOP CHANNEL
   - Target one instance of a channel name
     - STOP CHL(chlname) CONNAME(9.20.4.6)
     - STOP CHL(chlname) QMNAME(QM1)
   - Specify desired state of channel - stop retries without disabling completely
     - STOP CHL(chlname) MODE(INACTIVE)
3. RESET CLUSTER
   - Adds the QMID parameter
   - For situations where two qmgrs in a cluster have the same name
     - Source IP Address for channels

Performance, Scalability, Availability

1. Performance (numbers NOT from final drivers)
    - Continue the work started in V5.2
    - Even better throughput for persistent messages (150%-250% measured over V5.2)
    - JMS (MA88) and pubsub broker (MA0C) both updated
        - Number of Non-persistent messages delivered to subscribers improved by 3.3:1
        - Number of Persistent messages delivered to subscribers improved by 1.5:1
2. Scalability
    - Channel Process Pooling
        - Analogous to multi-threaded agents
        - More channels possible than via old runmqlsr; less memory used than inetd
    - Larger queues
        - Theoretical limit now ~2TB
        - Change max qdepth to from 640 000 to 999 999 999, to match z/OS
    - Minimum queue footprint reduced
        - From ~250K to ~64K
    - Rewrite of Shared Memory code
        - In particular for AIX®, to improve integration with Java, WebSphere Application Server etc.
        - Should allow more open queues
3. Availability
4. Microsoft Cluster (Support PAC MC74) integrated into base product

Miscellaneous Changes

1. New MQI option
    - 'shared hConn'
2. API Exits
    - Configurable through the Explorer for local qmgrs
3. Removal of
    - Internet Gateway (the cgi-bin interface)
    - Web Admin
4. Accessibility Compliance
    - For screen-readers etc
    - Consistency
    - FirstSteps to match LaunchPad (MSI install) style
5. Documentation
    - Supplied on separate CD-ROM
    - All books (not just family + Windows) available from Info Centre
    - Also PDF and HTML formats supplied

Licensing and Pricing

Questions to Ask: Licensing Entitlement and Compliance

- Has your company been involved with an acquisition or merger during the last year? Have you used MQ to interconnect the disparate systems between the companies?
- Is our company scaling up / adding CUs to maintain performance? Are you licensed for additional processor growth?
- Have you recently added an application server to connected new Internet servers to backend systems using MQ?
- How many licenses do you currently own? Do you keep current with new software versions? Is your software on maintenance?
- What platforms are your MQ servers installed on? How many processors are in those servers? Are you set up for test, QA and production environments and disaster recovery/failover? How many servers and CUs for each environment? Are they licensed properly for each?

This document is for Business Partner use only and is not a customer deliverable / handout.

IBM Corporation
Route 100
Somers, NY 10589
U.S.A.