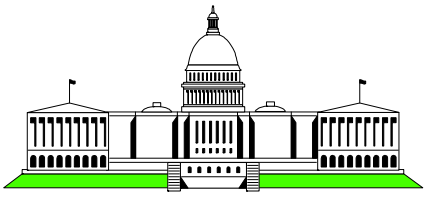
WebSphere MQ

a detailed view



Washington System Center

The Challenge

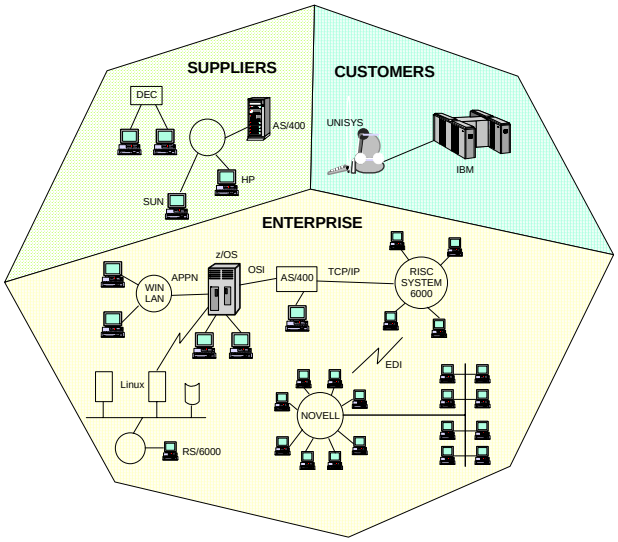
WSC
Advanced Technical Support

- Integration of Heterogeneous Applications on diverse platforms
  - Business divisions and applications are self contained, and do not easily share data
  - API's into applications are diverse
  - Format of data changes from application to application
  - Different code pages transportation protocols for inter-platform communication
  - Home grown methodology for data and process movement
- No standards for inter-business communication, many times new efforts must start from scratch
- MQSeries is the focal point of the IBM Business Integration strategy, which addresses integration of applications, data, and processes from both business and IT perspectives.

Current Situation

WSC Advanced Technical Support

The Challenge - Integration of Heterogenious platforms/applications/inter-business communication



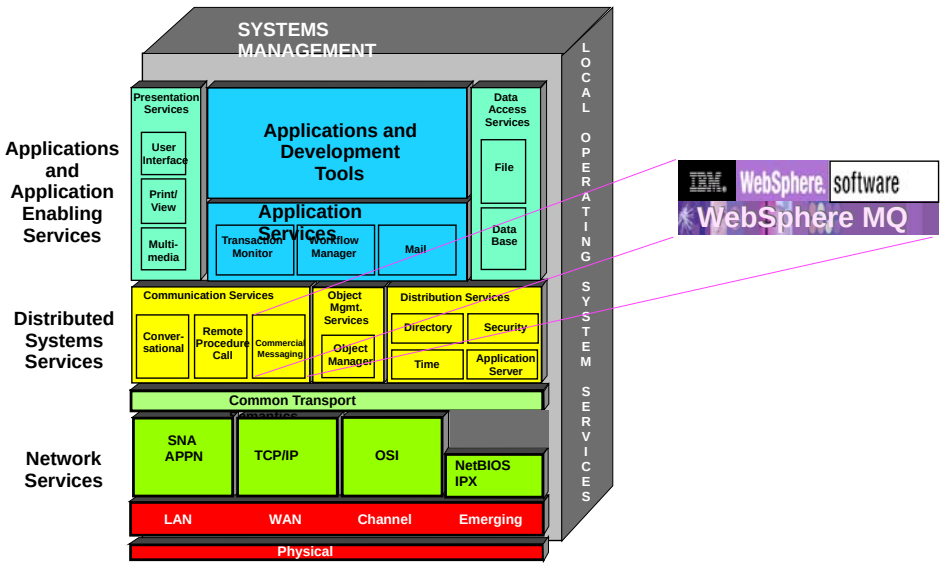
WebSphere MQ - the foundation of Business Integration (BI) technology

Provides -

- automatic inter-platform communication and conversion
- consistent framework and methodology for data movement
- Dynamic reconfiguration of WebSphere MQ network

Where does WebSphere MQ fit

WSC Advanced Technical Support

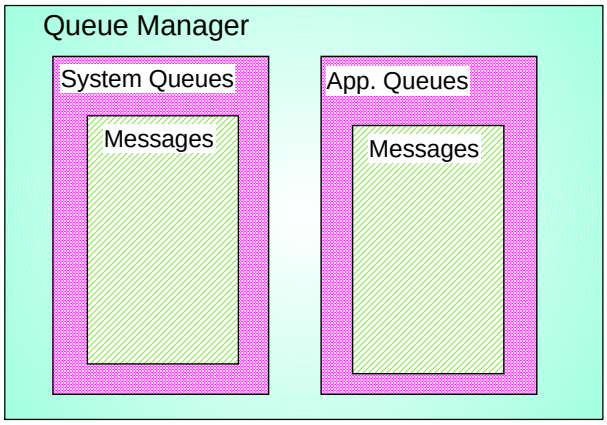


Foundational concepts of WebSphere MQ

- All communication and support structures within WebSphere MQ are based on messages and queues
- A Message -
  - a string of bytes that is meaningful to the applications that use it.
  - used to transfer information from one application program to another (or to different parts of the same application).
  - applications can be running on the same, or different platforms
- Queue
  - 2 types - system and application
  - A queue is a data structure used to store messages until they are retrieved by an application
  - system queues provide MQ support framework and functionality

Foundational Concepts of WebSphere MQ

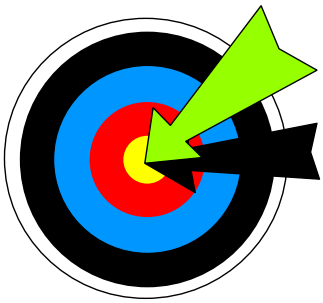
- Queue Manager
  - Messages and Queues are managed through a framework infrastructure called a queue manager
  - Queue managers can provide inter-machine communication
  - Not all platforms have a queue manager
    - Systems/platforms not able to support a queue manager have a client that can talk to a queue manager - details to follow



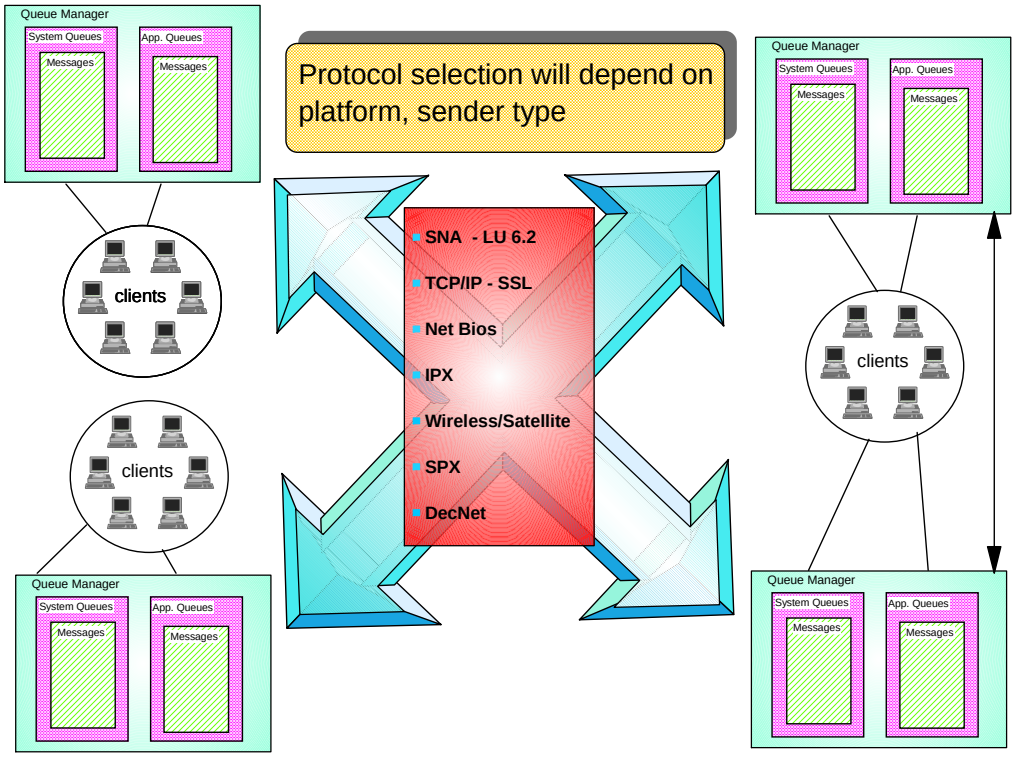
Why Messaging and Queuing

Provides unique capabilities and flexibility not found in other business infrastructure systems

- Provide build foundation for BI Event and Message Broker
- Provide build foundation for BI Workflow
- Elements included in WebSphere Application Server (JMS)
- Platform and operating system independence
- Virtualization of MQ networks
- User Application agnosticity
  - access to heterogeneous data types
- Asynchronous or 'Synchronous Like'
  - Link and Partner Process can be down
- Store and Forward - the hop concept
- Deliver once and only once
- Assured delivery



WebSphere MQ Network Intercommunication



WebSphere MQ Messaging Platforms

Servers:

- OS/390
- z/OS
- AIX
- Windows NT, 2000
- Solaris: Intel & SPARC
- HP-UX
- OS/400
- Compaq OpenVMS
- Compaq NSK
- Compaq Tru64 UNIX
- VSE/ESA
- Digital UNIX
- SunOS
- LInux (zSeries & Intel)

- SCO: Openserver, UnixWare
- IRIX
- Dynix/ptx
- NCR
- TPF
- DC/OSx
- Sinix
- PalmOS (MQe)
- EPOC (MQe)
- Java (MQe)
- Unisys 2200

Clients only:

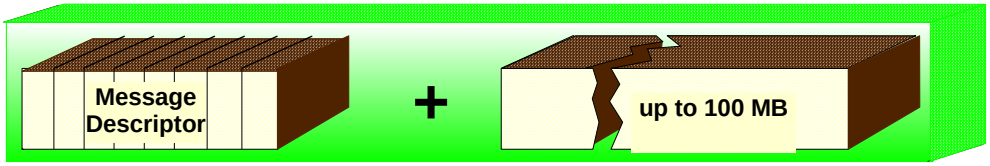
- DG/UX
- HP3000 MPE/iX
- Java
- Windows: 3.1,95,98
- DOS
- VM
- Apple MacOS
- Stratus VOS
- 4690
- Unisys A

39 Platforms

Message Composition

Messages are composed of 2 parts

- ✓ a control header used by the queue manager
- ✓ user data from the application

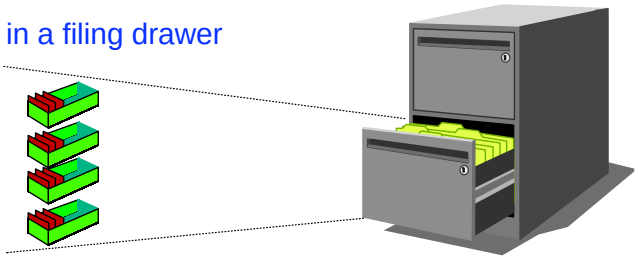


- Target Queue name
- Length of user data
- Message ID
- Correlation ID
- Message Priority
- Accounting token
- Time Stamp
- Reply to queue name
- Codepage
- Expiry time (option)
- Security (User Provided)

- Reporting options
  - Confirm on Arrival (COA)
  - Confirm on Delivery (COD)
- A set of message attributes, including:
  - Message Type
    1. Request - requires a reply
    2. Reply
    3. One-way message - does not require a reply (Datagram)
    4. Report - for errors
- Other

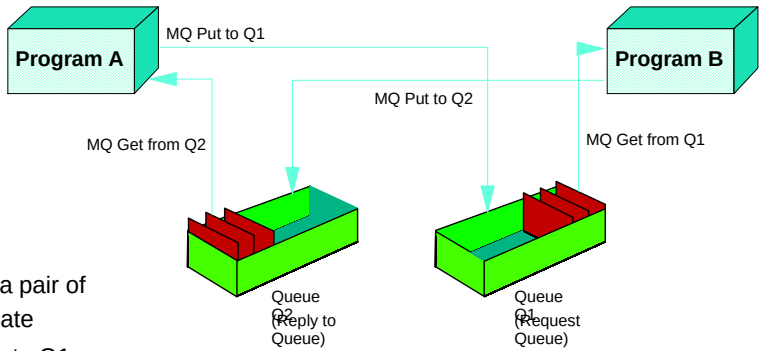
Component Interrelationships

- Queue manager and queues
  - all queues are created, managed and deleted by the queue manager
  - Accomplished through a control language
  - depending on the platform, a GUI may be used, or a command line prompt
- Messages and Queues
  - are given names and attributes - *more on this later*
  - Messages are added and removed from queues via the queue manager
    - added to a queue through MQ system call (put)
    - removed from a queue through MQ system call (get)
    - applications initiate puts/gets - queue manager services the call
- View a queue manager as a filing cabinet
  - queues as filing drawers
  - messages as sheets of paper in a filing drawer



How do Programs exchange data ?

- Programs communicate through the outlined infrastructure of queue manager and queues
- are one way only, an application does not put messages on a queue, and receive replies (gets) on the same queue



Application programs agree on a pair of queue names to intercommunicate

- Program A puts message to Q1
- Program B gets messages from Q1
- Program B puts replies to Q2
- Program A gets replies from Q2

Queues in detail

- A designated location on a platform
  - locations can be in main memory, or disk
  - underlying implementation is hidden from applications
    - implementation varies by platform - based on available hardware/software functionality
    - implementation used is based on platform strengths - *more later on this*
- Queues can have persistent/non-persistent attributes
- Persistent attribute
  - messages survive a system failure
  - activity is logged to assist in recovery
  - message hardened to disk by queue manager regardless of queue location
- Non-Persistent attribute
  - messages are not recovered across a system failure
  - activity is not logged
- Shared
  - queues are accessible by multiple queue managers

Note: some attributes can be overridden by the application, reference manuals should be checked for valid combinations

Implementation of Persistence by OS Platform

- WebSphere MQ for z/OS
  - MQ own storage methods
    - Linear page data sets
    - Commit at log time
    - Asynchronous write to Disk Storage
- WebSphere MQ for iSeries
    - Uses OS/400 data structures
- WebSphere MQ for XXX (Other than above)
    - Generally Flat Files

WebSphere MQ MQI (Message Queue Interface)

Applications can interface with MQ through the following interface calls, all major programming languages supported including PL/1, C, Java and Cobol, addition framework available to shield developers from the raw MQI calls

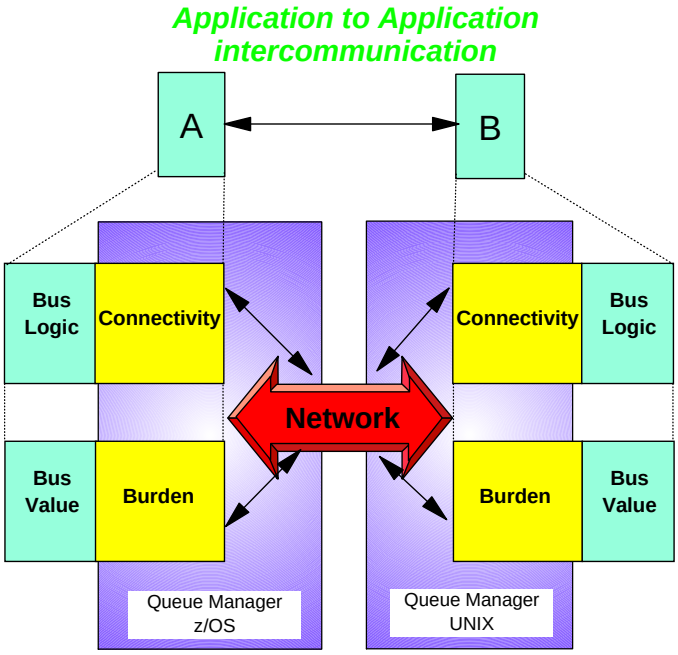
- **MQCONN** - Connect the application program to a Queue Manager
- **MQDISC** - Disconnected from a Queue Manager
- **MQOPEN** - Open a Queue
- **MQCLOSE** - Close a queue
- **MQPUT** - Put a message on a queue (one of a sequence of messages)
- **MQPUT1** - Put a single message on a queue
- **MQGET** - Get a message from a queue
- **MQINQ** - Inquire about queue attribute values
- **MQSET** - Changes queue attributes
- **MQCMIT** - Commit changes to MQM
- **MQBACK** - Backout changes to MQM

WebSphere MQ Networking

- MQ networks involve inter-queue manager communication
- All platforms that support a queue manager can participate in an MQ network
- The inter connect of queue managers provides a "virtualization" of the MQ network, making it independent of transport protocol
- Applications never have to worry about transport layer calls, burden of transport management and platform specific code page conversion are handled by WebSphere MQ.
- Virtualization process permits integration opportunities for diverse applications, through a common interface

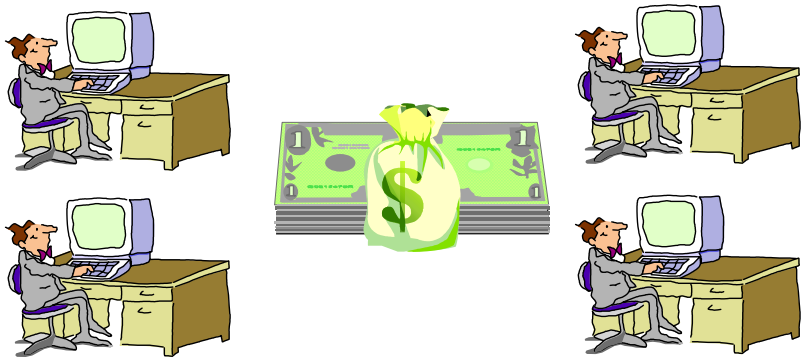
...Surfing the virtual MQ net

WebSphere MQ Networking



Connectivity Burden

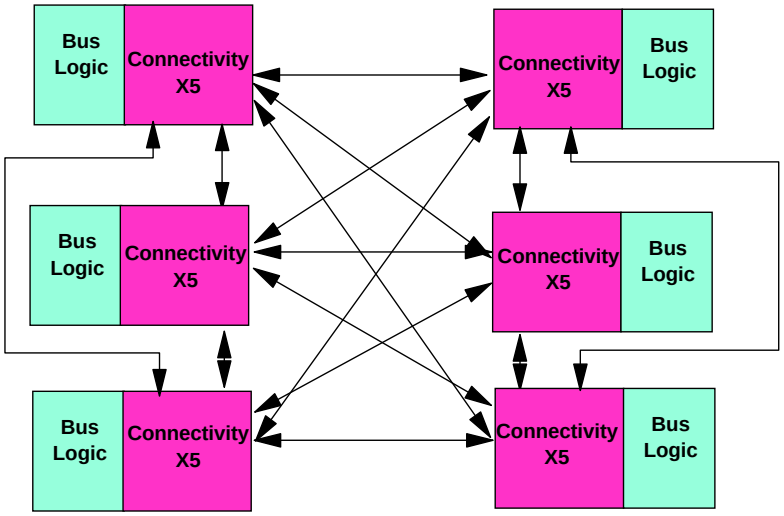
- Low level network programming
- High skill level required
- Time consuming
- Expensive
- Difficult to maintain
- Difficult to manage
- Complexity increases as additional connections are required



Application Connectivity Complexity

WSC Advanced Technical Support

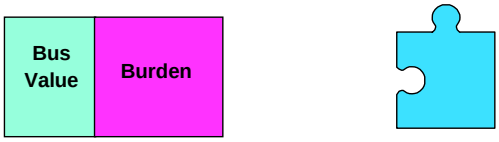
Application to Application



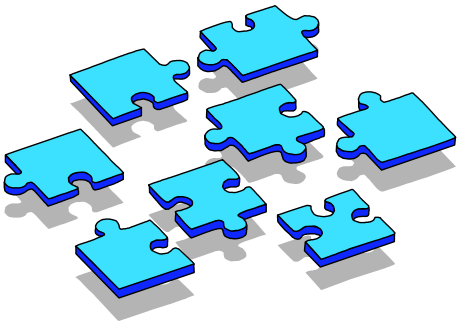
The Burden exists for each connection from each application

1 Piece of the Puzzle

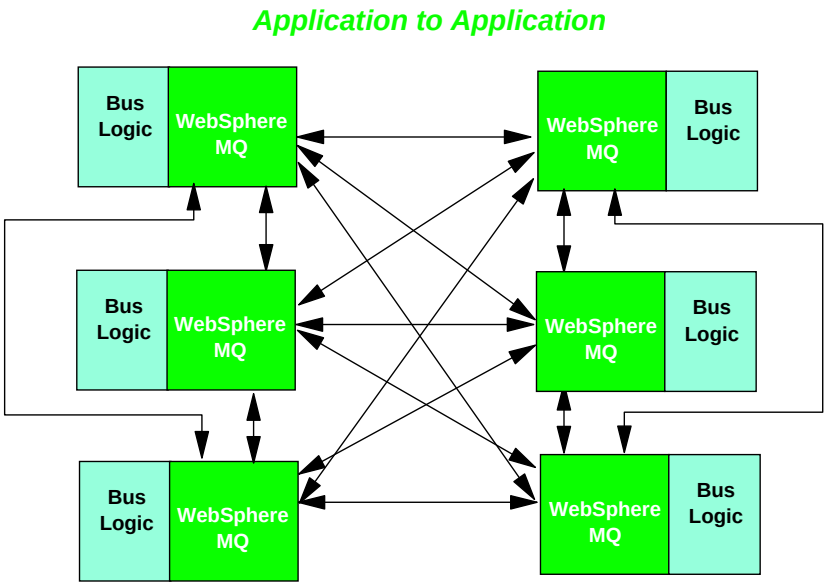
WSC Advanced Technical Support



- Burden Relief
  - Simplify the communications piece
  - Single common method on all platforms
  - Insulate the developer from the network
  - Provide assured message delivery
  - Allow time independent processing
  - Secure message transmission



WebSphere MQ



Platforms can (and probably are) heterogeneous

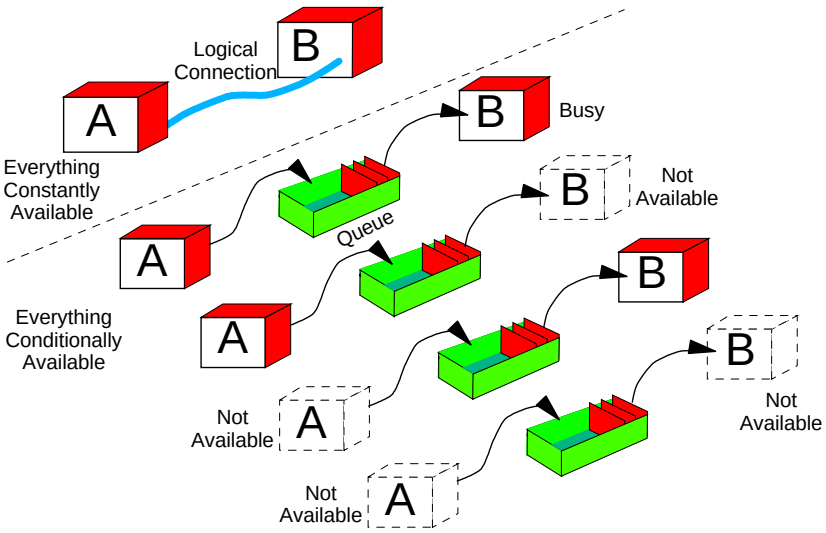
WebSphere MQ - Network Virtualization

- Alias
  - Through the use of the alias parameters on queue definitions there is the ability to dynamically redirect queue traffic
  - Application programs are not aware whether the queue name is real or an alias
  - changing which queue manager services a queue does not require the application program to be changed
- Clustering
  - Capabilities exist in WebSphere MQ to simplify the management of the queue manager interconnections
  - Clustering effectively provides fastpath inter-queue manager communication
- Sharing
  - The ability exists for one queue workload to be serviced by multiple queue managers.

WebSphere MQ - Network Virtualization

- Distributive Queue Manager (DQM)
- Vehicle used to enable queue managers to connect to a network of WebSphere MQ queue managers
- tools used to implement DQM
  - transmission queues
  - message channel agent (MCA)
    - manage code page conversions
    - exits for security (SSL), data compression
    - provides assured and delivery once of a message
  - queue manager alias definitions
  - reply to queue alias
  - additional headers contain routing information dynamically added to message to assist in delivery

Message Delivery & Availability Choices



WebSphere MQ and programming languages

- All major development languages are supported by MQ
- Java
  - WebSphere MQ base Java enables Java applets, applications, and servlets to issue calls and queries to WebSphere MQ. This gives access to mainframe and legacy applications, typically over the Internet, without necessarily having any other WebSphere MQ code on the client machine. With WebSphere MQ base Java, Internet users can become true participants in transactions, rather than just givers and receivers of information
  - Complete set of Java support classes comes with WebSphere MQ 5.3

MQ/Java Connectors - MQ Verb Support

- The MQ Java classes support the following basic MQ function verbs
  - MQBACK
  - MQBEGIN
  - ***MQCLOSE***
  - MQCMIT
  - ***MQCONN***
  - MQCONNX,
  - ***MQDISC***
  - ***MQGET***
  - MQINQ
  - ***MQOPEN***
  - ***MQPUT***
  - MQPUT1
  - MQSET

MQ/Java Connectors - basic classes

WSC Advanced Technical Support

- The MQ package contains the following Java classes
  - MQChannelDefinition
  - MQChannelExit
  - MQDistributionList
  - MQDistributionListItem
  - ***MQEnvironment*** - Most of the parameters are ignored in bind mode - required in network mode
  - MQException
  - MQGetMessageOptions
  - MQManagedObject
  - ***MQMessage*** - contains methods to read/write (build) a message in an MQBuffer
  - MQMessageTracker
  - MQPutMessageOptions
  - MQProcess
  - ***MQQueue*** - contains get() and put() methods (++) to send/receive/manipulate MQ queues
  - ***MQQueueManager*** - contains methods to set up a connection with a queue manager

is served

WebSphere MQ - a Foundation

WSC Advanced Technical Support

- Messaging and queuing with WebSphere MQ on 39 platforms
  - provide common and flexible mechanism for intercommunication of heterogeneous platforms
  - provides a set of common tools and architected framework to support additional Business Integration (BI) core products - WebSphere Business Integration Event and Message Broker and WebSphere Workflow
  - Various elements of WebSphere MQ found in other WebSphere based products
  - Support all major application development languages

WebSphere MQ install

- Comes in rpm format
- line command install
- current release is WebSphere MQ 5.3 + CSD levels
- fix packs also come in rpm format
- install directories are
  - for runtime code
    - /opt/mqm
  - for log and support files
    - /var/mqm
    - /var/mqm/qmgrs - directory location for all queue managers
- group and userids automatically generated - mqm/mqm

WebSphere MQ customization

- various line commands used to create and customize a queue manager
- crtmqm *qmanager* - creates a queue manager, and all the required system queues to
- System queues are named SYSTEM. *
- strmqm *qmanager* - used to start a queue manager
- runmqsc - interactive command line interface to issue commands to the queue manager , start channels, define queues, set attributes, etc
- runmqlsr - used to start up a listener on a tcp/ip port

WebSphere MQ - Intercommunication

WSC Advanced Technical Support

- Inter queue manager communication requires the following
- a local queue defined
  - queue name must be second queue manager
  - the usage of the queue must be of type *'transmission'*
  - command to define the queue is
    - define qlocal (*machine_name*) usage(xmitq)
    - command is run under the runmqsc command processor
    - queue is the means that WebSphere MQ finds additional network resources, and the receiving queue manager
- Channels
  - the actual vehicle that sends data out over the network
  - 2 types of channels need to be defined
    - a sender channel (sdr) used for outbound communication
    - a receiver channel (rcvr) used to received in bound communication

WebSphere MQ - Intercommunication

WSC Advanced Technical Support

- Channels - definition commands
  - for sender channel
    - define channel (*to_receiver_queue_manager*) chltype(sdr) conname(*machine_name*(*port*)) xmitq(*machine_name)* trptype(tcpip)
      - conname can be an ip address or real machine name
      - xmitq - a predefined queue used (in part) to establish communication with the receiving queue manager
      - channel name usually in format *to.queue.manager.name*, not required, however naming conventions should be used for clarity
  - for receiver channel
    - define channel (*to_sender_queue_manager*) chltype(rcvr) trptype(*tcp*)
    - *to_sender_queue_manager* is name of sender queue manager -
    - some naming convention should be established to avoid confusion
- listener must be running - *provides communication with the network*
    - runmqlsr -t tcp -p *port_number* -m *queue_manager &*
    - *-p* parameter can be omitted - defaults to port 1414
    - *-m* can be omitted - listener will be associated with the default queue manager
    - & - places the process in background and frees up the telnet/ssh session
    - endmqlsr -m *queue_manager* - used to terminate a listener

WebSphere MQ - Intercommunication

WSC Advanced Technical Support

- The same definitional procedure must be followed on the receiving queue manager
  - same channel names must be used - HOWEVER - sender channel name on sender queue manager now becomes receiver channel on receiving queue manager
  - transmission queue on receiving queue manager must be the same name as the sender queue manager
  - listener must be started, port number usage by both queue managers must be coordinated.
- Channels can also be started through inetd setup
  - if inetd is used, a manual start of a listener is not required - inetd does the listening for you
  - files to update are /etc/services and /etc/inetd.conf

WebSphere MQ - Intercommunication

WSC Advanced Technical Support

- This type of intercommunication structure is sufficient for a small number of queue managers
- More queue managers that require intercommunication require same type of set up
  - this becomes a burden to manage all the interconnections
- Solution - is to use Clustering
  - each queue manager contains a cluster sender channel, and a cluster receiver channel
- 2 queue managers act as repositories to manage and coordinate the cluster configuration

WebSphere MQ - Clustering Benefits

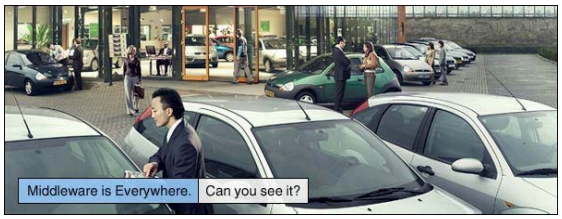
- Reduced administration overhead
- Improved availability and recovery capabilities
- Workload balancing
- no definition of a transmission queue is required
- remote queue definitions are not required

WebSphere MQ & Application Server

- WebSphere Application server and WebSphere MQ
  - 2 configurations available
  - A reduced function form of WebSphere MQ -
    - the queue manager providing the point-to-point messaging runs the WebSphere MQ function code provided with WebSphere Application Server. This environment is suited to simple messaging between JMS applications running in the WebSphere Application Server environment
  - Full function WebSphere MQ
    - the queue manager providing the point-to-point messaging runs WebSphere MQ full function code installed as a separate product from WebSphere Application Server. In this environment, JMS applications running in the WebSphere Application Server can participate fully in the functionality of a WebSphere MQ network. For example, they can make use of the IMS Bridge, or exchange messages with WebSphere MQ queue managers running on other platforms

Summary - WebSphere MQ

- Removes many barriers to the intercommunication of Enterprise applications
- Enables WebSphere based applications to participate in a MQ network (*WebSphere MQ required*)
- Provides WebSphere based applications to intercommunicate through the use of JMS
- WebSphere MQ is used as the foundation for WebSphere Business Integration Products



Resources

- http://www.redbooks.ibm.com
  - currently 83 redbooks referencing WebSphere MQ/MQSeries
- SC34-5349 - Queue Manager Clusters
- WebSphere MQ Information Center - online documentation
- http://www.software.ibm.com/ts/mqseries *
  - support pacs
  - add-on functionality
  - documentation, white papers, studies
- http://www-306.ibm.com/software/integration/support/supportpacs/category.html#cat1*