#SHAREorg

SHARE

WebSphere MQ HA and DR

Mark Taylor
marke_taylor@uk.ibm.com
IBM Hursley

Session 13929

SHARE in Boston

Introduction

- Availability is a very large subject
- You can have the best technology in the world, but you have to manage it correctly
- Technology is not a substitute for good planning and testing!

What is DR – Wikipedia Version

- Disaster recovery is the process, policies and procedures related to preparing for recovery or continuation of technology infrastructure critical to an organization after a natural or human-induced disaster. Disaster recovery is a subset of business continuity. While business continuity involves planning for keeping all aspects of a business functioning in the midst of disruptive events, disaster recovery focuses on the IT or technology systems that support business functions.

What is DR

- Getting applications running after a major (often whole-site) failure or loss
- It is not about High Availability although often the two are related and share design and implementation choices
  - "HA is having 2, DR is having them a long way apart"
  - More seriously, HA is about keeping things running, while DR is about recovering when HA has failed.
- Requirements driven by business, and often by regulators
  - Data integrity, timescales, geography …
- One major decision point: cost
  - How much does DR cost you, even if it's never used?
  - How much are you prepared to lose

Disaster Recovery vs High Availability

- Designs for HA typically involve a single site for each component of the overall architecture
- Designs for DR typically involve separate sites

- Designs for HA (and CA) typically require no data loss
- Designs for DR typically can have limited data loss

- Designs for HA typically involve high-speed takeover
- Designs for DR typically can permit several hours down-time

HIGH AVAILABILITY

Single Points of Failure

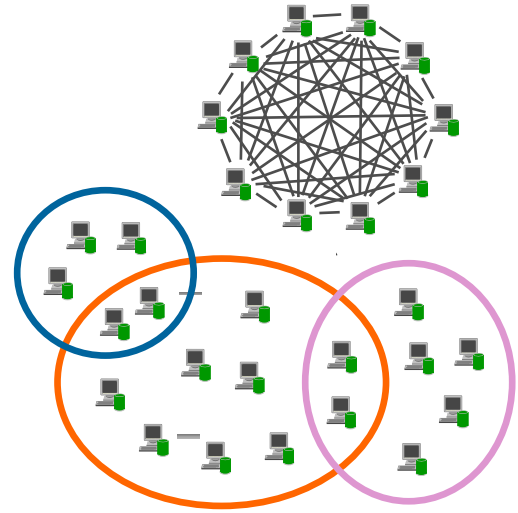
- With no redundancy or fault tolerance, a failure of any component can lead to a loss of availability
- Every component is critical. The system relies on the:
  - Power supply, system unit, CPU, memory
  - Disk controller, disks, network adapter, network cable
  - ...and so on
- Various techniques have been developed to tolerate failures:
  - UPS or dual supplies for power loss
  - RAID for disk failure
  - Fault-tolerant architectures for CPU/memory failure
  - ...etc
- Elimination of SPOFs is important to achieve HA

WebSphere MQ HA technologies

- Queue manager clusters
- Queue-sharing groups
- Support for networked storage
- Multi-instance queue managers
- HA clusters
- Client reconnection

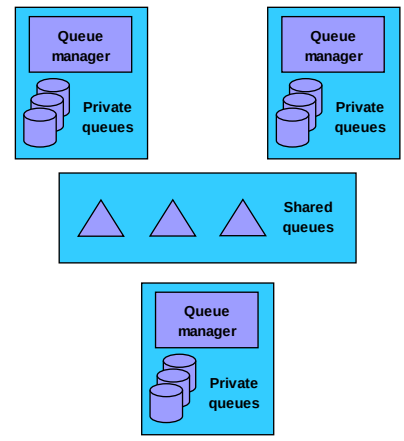
Queue Manager Clusters

- Sharing cluster queues on multiple queue managers prevents a queue from being a SPOF
- Cluster workload algorithm automatically routes traffic away from failed queue managers



Queue-Sharing Groups

- On z/OS, queue managers can be members of a queue-sharing group
- Shared queues are held in a coupling facility
  - All queue managers in the QSG can access the messages
- Benefits:
  - Messages remain available even if a queue manager fails
  - Pull workload balancing
  - Apps can connect to the group



Introduction to Failover and MQ

- Failover is the automatic switching of availability of a service
  - For MQ, the "service" is a queue manager
- Traditionally the preserve of an HA cluster, such as HACMP
- Requires:
  - Data accessible on all servers
  - Equivalent or at least compatible servers
    - Common software levels and environment
  - Sufficient capacity to handle workload after failure
    - Workload may be rebalanced after failover requiring spare capacity
  - Startup processing of queue manager following the failure
- MQ offers two ways of configuring for failover:
  - Multi-instance queue managers
  - HA clusters

Failover considerations

- Failover times are made up of three parts:
  - Time taken to notice the failure
    - Heartbeat missed
    - Bad result from status query
  - Time taken to establish the environment before activating the service
    - Switching IP addresses and disks, and so on
  - Time taken to activate the service
    - This is queue manager restart
- Failover involves a queue manager restart
  - Nonpersistent messages, nondurable subscriptions discarded
- For fastest times, ensure that queue manager restart is fast
  - No long running transactions, for example

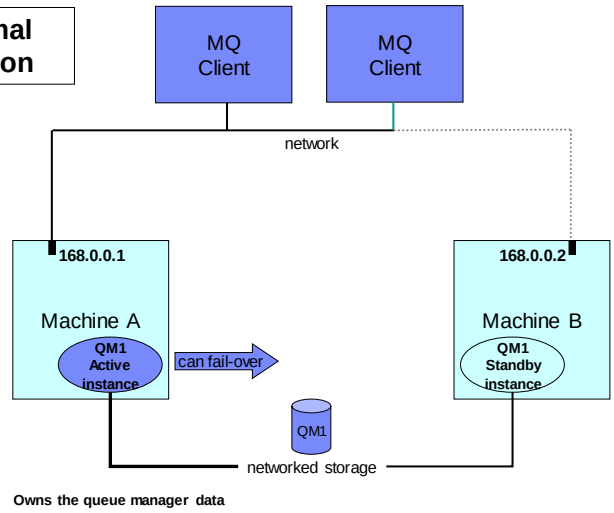
MULTI-INSTANCE QUEUE MANAGERS

Multi-instance Queue Managers

- Basic failover support without HA cluster
- Two instances of a queue manager on different machines
  - One is the “active” instance, other is the “standby” instance
  - Active instance “owns” the queue manager’s files
    - Accepts connections from applications
  - Standby instance monitors the active instance
    - Applications cannot connect to the standby instance
    - If active instance fails, standby restarts queue manager and becomes active
- Instances are the SAME queue manager – only one set of data files
  - Queue manager data is held in networked storage

Multi-instance Queue Managers

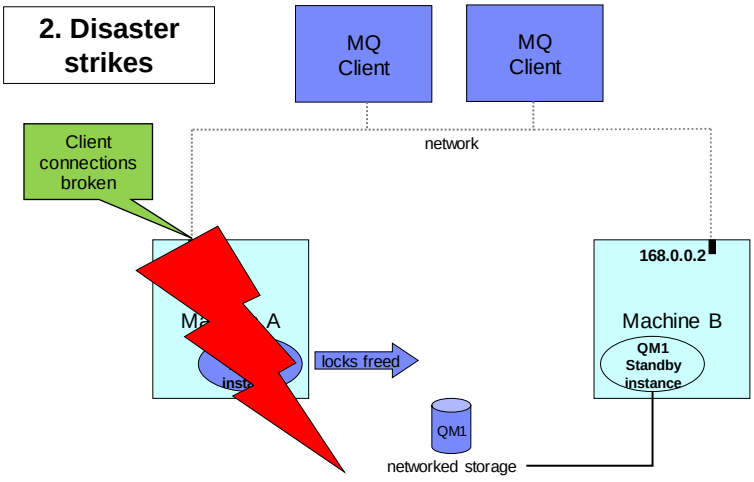
1. Normal execution



Owns the queue manager data

Multi-instance Queue Managers

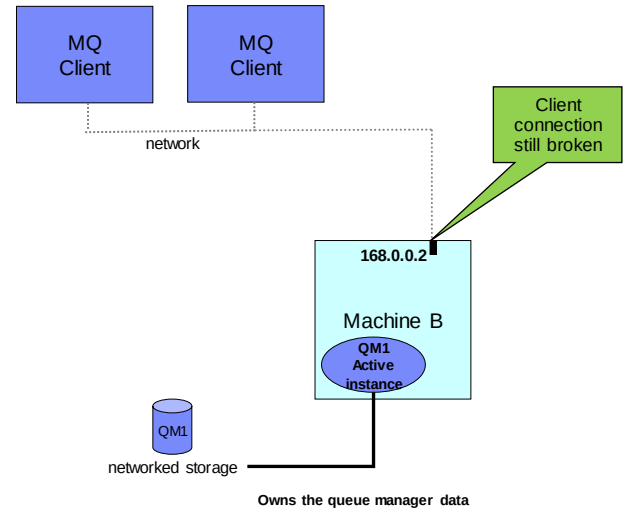
2. Disaster strikes



Multi-instance Queue Managers

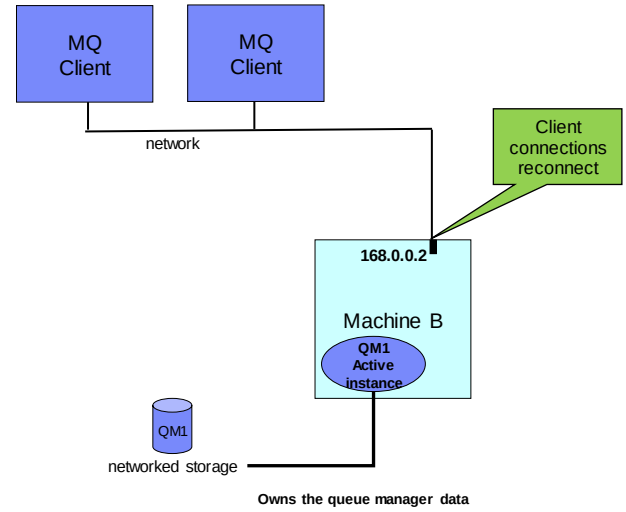
3. FAILOVER

Standby becomes active



Multi-instance Queue Managers

4. Recovery complete



Multi-instance Queue Managers

- MQ is NOT becoming an HA cluster coordinator
  - If other resources need to be coordinated, you need an HA cluster
  - WebSphere Message Broker integrates with multi-instance QM
  - Queue manager services can be automatically started, but with limited control
- System administrator is responsible for restarting another standby instance when failover has occurred

Dealing with multiple IP addresses

- The IP address of the queue manager changes when it moves
  - So MQ channel configuration needs way to select address
- Connection name syntax extended to a comma-separated list
  - CONNAME('168.0.0.1,168.0.0.2')
  - Needs 7.0.1 qmgr or client
- Unless you use external IPAT or an intelligent router or MR01
- WAS8 admin panels understand this syntax.
- For earlier levels of WAS
  - Connection Factories:
    - Set a custom property called XMSC_WMQ_CONNECTION_NAME_LIST to the list of host/port names that you wish to connect to
    - Make sure that the existing host and port values defined on the connection factory match the first entry in this property
  - Activation Specs:
    - Set a custom property called connectionNameList on the activation spec with the same format

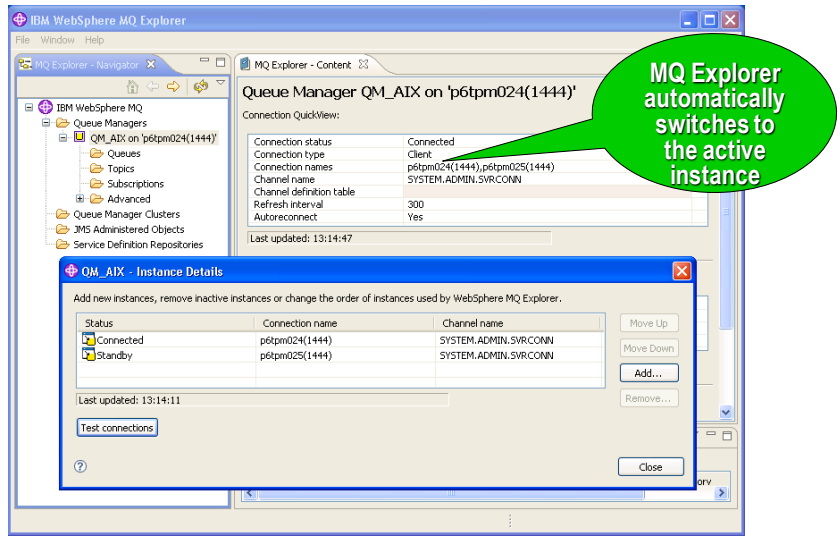
Administering Multi-instance QMgrs

- All queue manager administration must be performed on active instance
- dspmq enhanced to display instance information

```
$ hostname
staravia
$ dspmq -x
QMNAME(MIQM)        STATUS(Running as standby)
    INSTANCE(starly)      MODE(Active)
    INSTANCE(staravia)    MODE(Standby)
```

  - dspmq issued on “staravia”
  - On “staravia”, there’s a standby instance
  - The active instance is on “starly”

Multi-instance QMgr in MQ Explorer



HA CLUSTERS

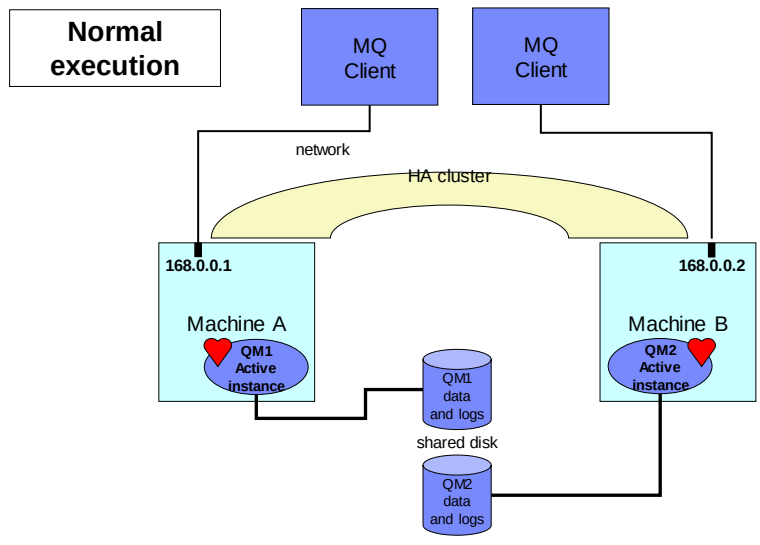
HA clusters

- MQ traditionally made highly available using an HA cluster
  - IBM PowerHA for AIX (formerly HACMP), Veritas Cluster Server, Microsoft Cluster Server, HP Serviceguard, …
- HA clusters can:
  - Coordinate multiple resources such as application server, database
  - Consist of more than two machines
  - Failover more than once without operator intervention
  - Takeover IP address as part of failover
  - Likely to be more resilient in cases of MQ and OS defects

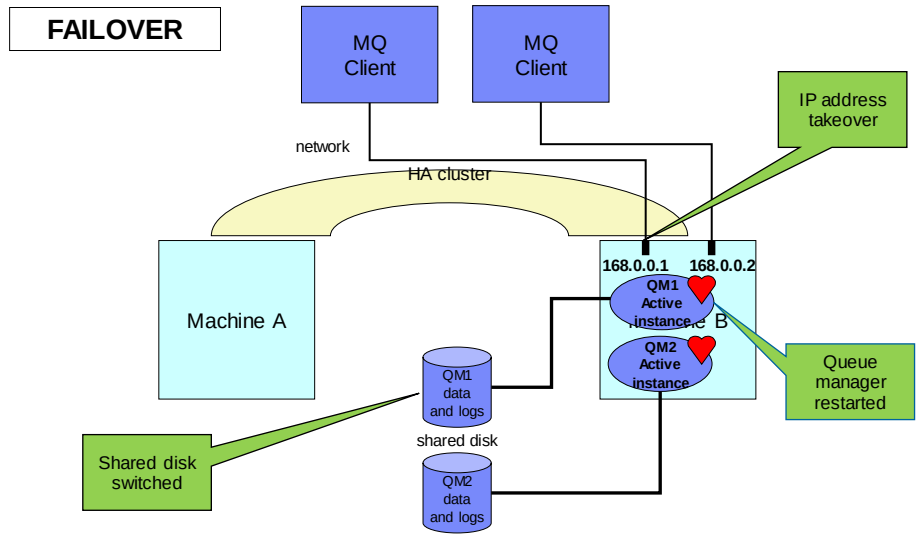
HA clusters

- In HA clusters, queue manager data and logs are placed on a shared disk
  - Disk is switched between machines during failover
- The queue manager has its own “service” IP address
  - IP address is switched between machines during failover
  - Queue manager’s IP address remains the same after failover
- The queue manager is defined to the HA cluster as a resource dependent on the shared disk and the IP address
  - During failover, the HA cluster will switch the disk, take over the IP address and then start the queue manager

MQ in an HA cluster – Active/active



MQ in an HA cluster – Active/active



Multi-instance QM or HA cluster?

- Multi-instance queue manager
  - ✓ Integrated into the WebSphere MQ product
  - ✓ Faster failover than HA cluster
    - ✓ Delay before queue manager restart is much shorter
  - × Runtime performance of networked storage
  - × Suitable storage can sometimes be a challenge
- HA cluster
  - ✓ Capable of handling a wider range of failures
  - ✓ Failover historically rather slow, but some HA clusters are improving
  - ✓ Capable of more flexible configurations (eg N+1)
  - × Required MC91 SupportPac or equivalent configuration
  - × Extra product purchase and skills required
- Storage distinction
  - Multi-instance queue manager typically uses NAS
  - HA clustered queue manager typically uses SAN

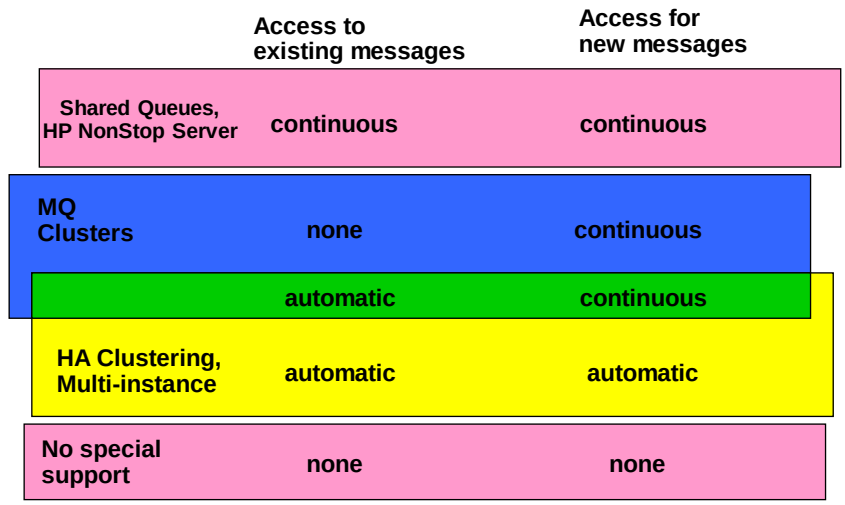
Creating QM for failover

- Create filesystems on the shared disk, for example
  - /MQHA/QM1/data for the queue manager data
  - /MQHA/QM1/log for the queue manager logs
- On one of the nodes:
  - Mount the filesystems
  - Create the queue manager
    - crtmqm –md /MQHA/QM1/data –ld /MQHA/QM1/log QM1
  - Print out the configuration information for use on the other nodes
    - dspmqinf –o command QM1
- On the other nodes:
  - Mount the filesystems
  - Add the queue manager's configuration information
    - addmqinf –s QueueManager –v Name=QM1 –v Prefix=/var/mqm –v DataPath=/MQHA/QM1/data/QM1 –v Directory=QM1

Virtual Systems

- Another mechanism being regularly used
- When MQ is in a virtual machine … simply shoot and restart the VM
- "Turning it off and back on again"
- Can be faster than any other kind of failover

Comparison of Technologies



APPLICATIONS AND AUTO-RECONNECTION

HA applications – MQ connectivity

- If an application loses connection to a queue manager, what does it do?
  - End abnormally
  - Handle the failure and retry the connection
  - Reconnect automatically thanks to application container
    - WebSphere Application Server contains logic to reconnect JMS clients
  - Use MQ automatic client reconnection

Automatic client reconnection

- MQ client automatically reconnects when connection broken
  - MQI C clients and standalone JMS clients
  - JMS in app servers (EJB, MDB) does not need auto-reconnect
- Reconnection includes reopening queues, remaking subscriptions
  - All MQI handles keep their original values
- Can reconnect to same queue manager or another, equivalent queue manager
- MQI or JMS calls block until connection is remade
  - By default, will wait for up to 30 minutes
  - Long enough for a queue manager failover (even a really *slow* one)

Automatic client reconnection

- Can register event handler to observe reconnection
- Not all MQI is seamless, but majority repaired transparently
  - Browse cursors revert to the top of the queue
  - Nonpersistent messages are discarded during restart
  - Nondurable subscriptions are remade and may miss some messages
  - In-flight transactions backed out
- Tries to keep dynamic queues with same name
  - If queue manager doesn't restart, reconnecting client's TDQs are kept for a while in case it reconnects
  - If queue manager does restart, TDQs are recreated when it reconnects

Automatic client reconnection

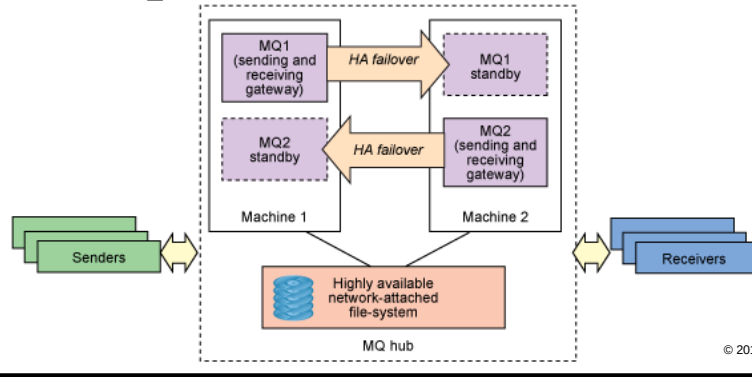
- Enabled in application code, ini file or CLNTCONN definition
  - MQI: MQCNO_RECONNECT, MQCNO_RECONNECT_Q_MGR
  - JMS: Connection factory properties
- Plenty of opportunity for configuration
  - Reconnection timeout
  - Frequency of reconnection attempts
- Requires:
  - Threaded client
  - 7.0.1 server – including z/OS
  - Full-duplex client communications (SHARECNV >= 1)

Client Configurations for Availability

- Use wildcarded queue manager names in CCDT
  - Gets weighted distribution of connections
  - Selects a “random” queue manager from an equivalent set
- Use multiple addresses in a CONNAME
  - Could potentially point at different queue managers
  - More likely pointing at the same queue manager in a multi-instance setup
- Use automatic reconnection
- Pre-connect Exit from V7.0.1.4
- Use IP routers to select address from a list
  - Based on workload or anything else known to the router

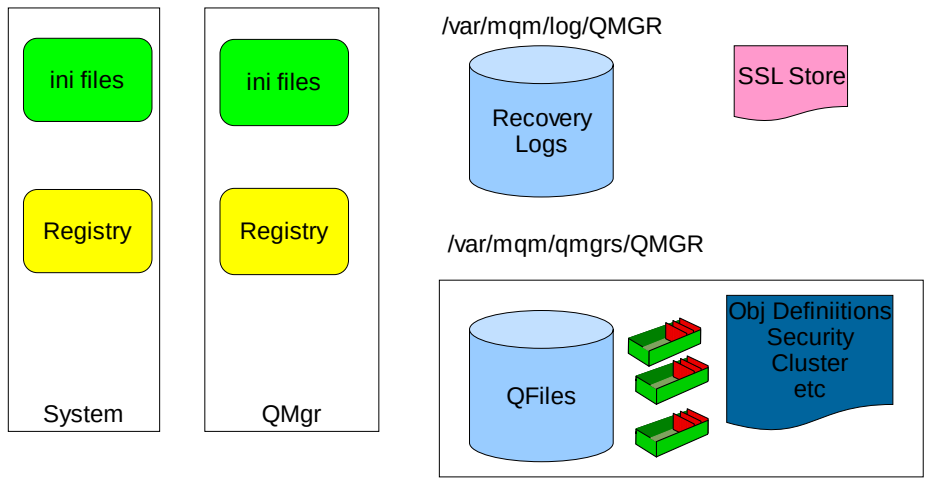
Application Patterns for availability

- Article describing examples of how to build a hub topology supporting:
  - Continuous availability to send MQ messages, with no single point of failure
  - Linear horizontal scale of throughput, for both MQ and the attaching applications
  - Exactly once delivery, with high availability of individual persistent messages
  - Three messaging styles: Request/response, fire-and-forget, and pub/sub
- http://www.ibm.com/developerworks/websphere/library/techarticles/1303_broadhurst/1303_broadhurst.html



Disaster Recovery

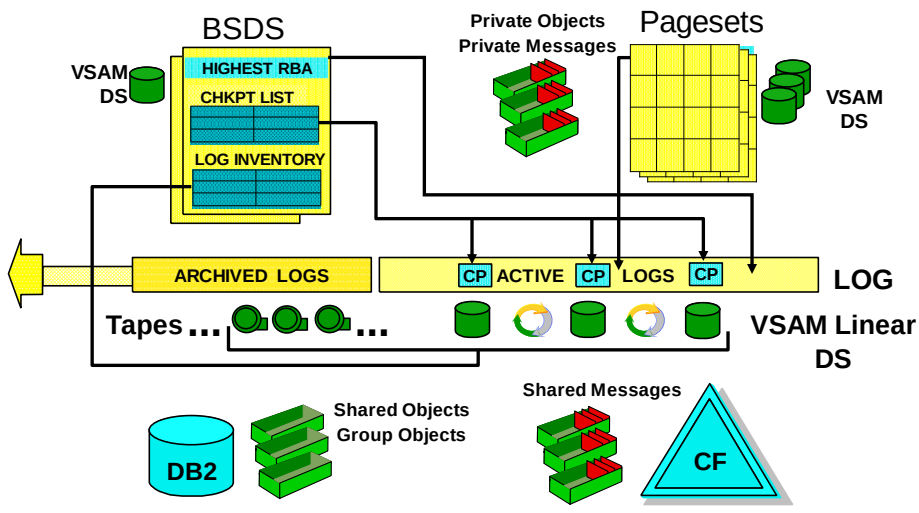
What makes a Queue Manager on Dist?



Backups

- At minimum, backup definitions at regular intervals
  - Include ini files and security settings
- One view is there is no point to backing up messages
  - They will be obsolete if they ever need to be restored
  - Distributed platforms – data backup only possible when qmgr stopped
- Use rcdmqimg on Distributed platforms to take images
  - Channel sync information is recovered even for circular logs
- Backup everything before upgrading code levels
  - On Distributed, you cannot go back
- Exclude queue manager data from normal system backups
  - Some backup products interfere with WMQ processing

What makes a Queue Manager on z/OS?



What makes up a Queue Manager?

- Queue manager started task procedure
  - Specifies MQ libraries to use, location of BSDS and pagesets and INP1, INP2 members start up processing
- System Parameter Module – zParm
  - Configuration settings for logging, trace and connection environments for MQ
- BSDS: Vital for Queue Manager start up
  - Contains info about log RBAs, checkpoint information and log dataset names
- Active and Archive Logs: Vital for Queue Manager start up
  - Contain records of all recoverable activity performed by the Queue Manager
- Pagesets
  - Updates made “lazily” and brought “up to date” from logs during restart
  - Start up with an old pageset (restored backup) is not really any different from start up after queue manager failure
  - Backup needs to copy page 0 of pageset first (don’t do volume backup!)
- DB2 Configuration information & Group Object Definitions
- Coupling Facility Structures
  - Hold QSG control information and MQ messages

Backing Up a z/OS Queue Manager

- Keep copies of ZPARM, MSTR procedure, product datasets and INP1/INP2 members
- Use dual BSDS, dual active and dual archive logs
- Take backups of your pagesets
  - This can be done while the queue manager is running (fuzzy backups)
  - Make sure you backup Page 0 first, REPRO or ADRDSSU logical copy
- DB2 data should be backed up as part of the DB2 backup procedures
- CF application structures should be backed up on a regular basis
  - These are made in the logs of the queue manager where the backup was issued

Remote Recovery

Topologies

- Sometimes a data centre is kept PURELY as the DR site
- Sometimes 2 data centres are in daily use; back each other up for disasters
  - Normal workload distributed to the 2 sites
  - These sites are probably geographically distant
- Another variation has 2 data centres “near” each other
  - Often synchronous replication
  - With a 3rd site providing a long-distance backup
- And of course further variations and combinations of these

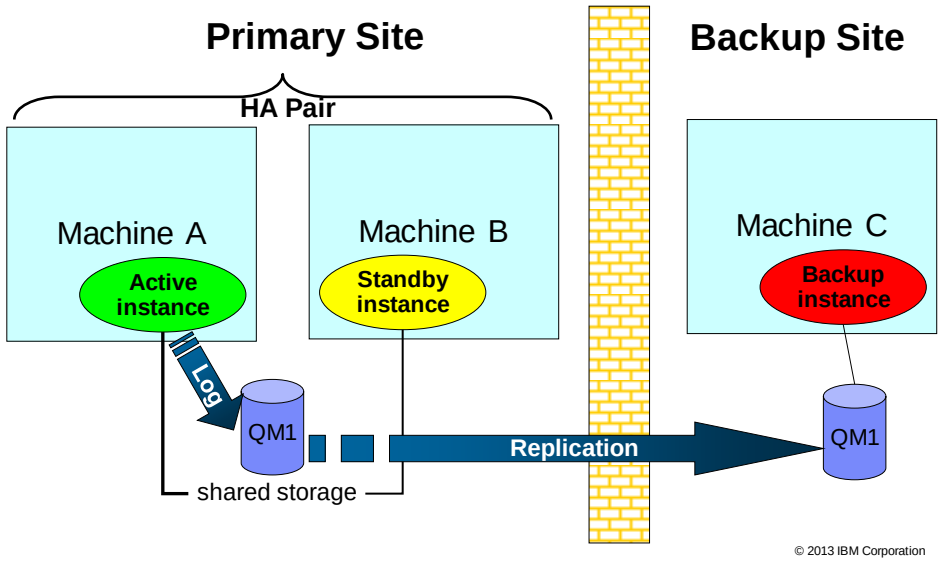
Queue Manager Connections

- DR topologies have little difference for individual queue managers
- But they do affect overall design
  - Where do applications connect to
  - How are messages routed
- Clients need ClntConn definitions that reach any machine
- Will be affected by how you manage network
  - Do DNS names move with the site?
  - Do IP addresses move with the site?
- Some sites always put IP addresses in CONNAME; others use hostname
  - No rule on which is better

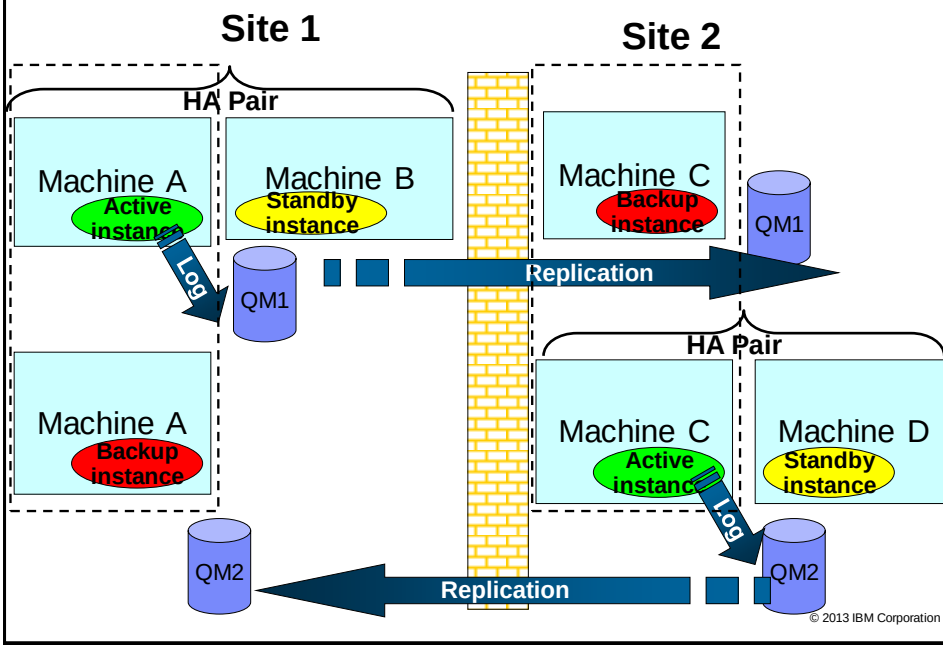
Disk replication

- Disk replication can be used for WMQ disaster recovery
- Either synchronous or asynchronous disk replication is OK
  - Synchronous:
    - No data loss if disaster occurs
    - Performance is impacted by replication delay
    - Limited by distance (eg 100km)
  - Asynchronous:
    - Some limited data loss if disaster occurs
    - It is critical that queue manager data and logs are replicated in the same consistency group if replicating both
- Disk replication cannot be used between the active and standby instances of a multi-instance queue manager
  - Could be used to replicate to a DR site in addition though

Combining HA and DR



Combining HA and DR – "Active/Active"



Backup Queue Manager - Objective

- Feature introduced in WMQ V6
- Prepares a queue manager for restart/recovery
  - Without needing to replay all logs at a critical time
  - For Windows, Unix and System i
- "Backup" queue manager takes the place of original
  - New QMID but contains original definitions and messages

Backup Queue Manager - Procedure

- Configure queue manager with linear logging
- Create a queue manager at the primary site
  - Create an identical one at the DR site – the backup queue manager
- Ship full, inactive log files from active QM to the DR site
  - Can use disk replication to do this
  - Or modify SupportPac or sample programs for log management to copy files at the same time as deleting/archiving local logs
- Replay log files on the backup QM to bring it up to date
  - Do this at regular intervals
  - strmqm -r
- If disaster occurs, activate the backup queue manager
  - strmqm -a
- For more control, can force filling of current log file
  - MQSC RESET QMGR TYPE(ADVANCELOG)

Integration with other products

- May want to have consistency with other data resources
  - For example, databases and app servers
- Only way for guaranteed consistency is disk replication where all logs are in same group
  - Otherwise transactional state might be out of sync
- DB2 can use WMQ as part of its own replication strategy
  - InfoSphere Replication Server

Planning and Testing

Planning for Recovery

- Write a DR plan
  - Document everything – to tedious levels of detail
  - Include actual commands, not just a description of the operation
    - Not “Stop MQ”, but “as mqm, run /usr/local/bin/stopmq.sh US.PROD.01”
- And test it frequently
  - Recommend twice a year
  - Record time taken for each task
- Remember that the person executing the plan in a real emergency might be under-skilled and over-pressured
  - Plan for no access to phones, email, online docs …
- Each test is likely to show something you’ve forgotten
  - Update the plan to match
  - You’re likely to have new applications, hardware, software …
- May have different plans for different disaster scenarios

Example Exercises from MQ Development

- Different groups have different activities that must continue
  - Realistic scenarios can help show what might not be available
- From the WMQ development lab …
- Most of the change team were told there was a virulent disease and they had to work from home
  - Could they continue to support customers
- If Hursley machine room was taken out by a plane missing its landing at Southampton airport
  - Could we carry on developing the WMQ product
  - Source code libraries, build machines, test machines …
  - Could fixes be produced
- (A common one) Someone hit emergency power-off button

Networking Considerations

- DNS - You will probably redirect hostnames to a new site
  - But will you also keep the same IP addresses?
  - Including NAT when routing to external partners?
  - Affects CONNAME
- Include external organisations in your testing
  - 3rd parties may have firewalls that do not recognize your DR servers
- LOCLADDR configuration
  - Not normally used by MQ, but firewalls, IPT and channel exits may inspect it
  - May need modification if a machine changes address
- Clustering needs special consideration
  - Easy to accidentally join the real cluster and start stealing messages
  - Ideally keep network separated, but can help by:
    - Not giving backup 'live' security certs
    - Not starting chinit address space (z/OS)
    - Not allowing channel initiators to start (distributed)
    - Use CHLAUTH rules
- Backup will be out of sync with the cluster
  - REFRESH CLUSTER() resolves updates

A Real MQ Network Story

- Customer did an IP move during a DR test
- Forgot to do the IP move back when they returned to prime systems
- Didn't have monitoring in place that picked this up until users complained about lack of response

Other Resources

- Applications may need to deal with replay or loss of data.
  - Decide whether to clear queues down to a known state, or enough information elsewhere to manage replays
- Order of recovery may change with different product releases
  - Every time you install a new version of a product revisit your DR plan
- What do you really need to recover
  - DR site might be lower-power than primary site
  - Some apps might not be critical to the business
  - But some might be unrecognised prereqs

If a Real Disaster Hits

- Hopefully you never need it. But if the worst happens:
- Follow your tested plan
  - Don't try shortcuts
- But also, if possible:
  - Get someone to take notes and keep track of the time tasks took
  - Prepare to attend post mortem meetings on steps you took to recover
  - Accept all offers of assistance
- And afterwards:
  - Update your plan for the next time

Summary

- Various ways of recovering queue managers
- Plan what you need to recover for WMQ
- Plan the relationship with other resources
- Test your plan

13929 – MQ HA and DR