MQ Version 8 on NonStop:

High Availability and Options for DR

BITUG Little SIG
December 2019

This presentation is for entertainment purposes only. The opinions expressed by the presenter are not necessarily those by HPE or IBM.

No animals were harmed during the making of this presentation.

MQ 8 HA and DR
Agenda

1. NonStop high availability 2019 reviewed
2. MQ 8 high availability
3. MQ 8 disaster recovery options
4. Optional: runnscnf demystified

MQ 8 HA and DR

NonStop High Availabilty Reviewed

In prehistoric times Tandem (remember the name?) had concepts for fault tolerance/high availability:

- Hardware redundancy everywhere
  - dual paths to disks
  - disks with their own power supplies
  - …
- Processor lock-stepping
- DMR/TMR in the NSAA architecture
- Process pairs in user written applications
- These were great at that point in time, but …

MQ 8 HA and DR

NonStop High Availabilty Reviewed

Times have changed – and NonStop has changed.

Most of the old concepts are gone (or will be soon).

WHY?

Is high availability less important today?

MQ 8 HA and DR

NonStop High Availabilty Reviewed

High availability is as important today as ever – maybe even more important…

BUT:

MQ 8 HA and DR

NonStop High Availabilty Reviewed

- Hardware in general has become much more reliable
- When did you have your last hardware CPU or controller failure?
- 90 % of hardware failures are software failures…
- and this number is growing…
- … as more and more “hardware” features are implemented in software
- In a Virtualized NonStop many “hardware” components are software!
- With correct configuration availability of Virtualized NonStop very good

MQ 8 HA and DR

NonStop High Availabilty Reviewed

What about software?

- TMF in many cases has replaced the need for process pairs
- Many important OS subsystems are not process pairs:
  - Java
  - TCP/IP
- Fast (a few seconds), reliable, automatic restart as good as “invisible” takeover
- Many transactions are recoverable (retried on failure)
- External systems (like in manufacturing) have more local autonomy
- NonStop technology has become easier to develop for
- Complex concepts (checkpointing process pairs) rarely needed anymore …
- … to preserve NonStop fundamentals (like availability and data integrity)

MQ 8 HA and DR

MQ High Availabilty: The MQ 5 Approach

- Several checkpointing, highly complex process pairs:
  - Execution controller
  - Queue server
  - Channel server
- Processes can be distributed across several CPUs: product scales
- CPU down typically affects only part of the product
- Failover is fast, as backup processes always up to date
- Design very NonStop specific, large deviation from IBM “distributed” code base
- Bringing patches from “distributed” to NonStop difficult…
- … and so in many cases was not done

MQ 8 HA and DR

MQ High Availabilty: The MQ 8 Approach

- Main process (“EC”, amqzxma0) process pair, but NOT checkpointing
- Backup EC prestarts part of queue manager on its CPU
- CPU down typically affects all or nothing of the product
- If main MQ CPU goes down, backup EC continues bootstrap until queue manager fully up
- Failover takes longer than with MQ 5
- … but typically still done within a few seconds
  - your mileage may vary
  - a lot faster on NonStop x
  - depends on system load, disk type (SSD?), size of queues etc.

MQ 8 HA and DR

MQ High Availabilty: The MQ 8 Approach

Checkpointing process pairs:

- Cache manager (amqcache)
  - Multiple instances (on different CPUs) possible
  - Stores non persistent messages
  - Checkpointing optional (per process)
  - Can cause significant load
  - Can be run on any CPU

- Setsignal Manager (amqssmgr)
  - Sends signal IPC message to getting applications
  - Very lightweight process
  - Typically no consideration for scalability
  - Can be run on any CPU

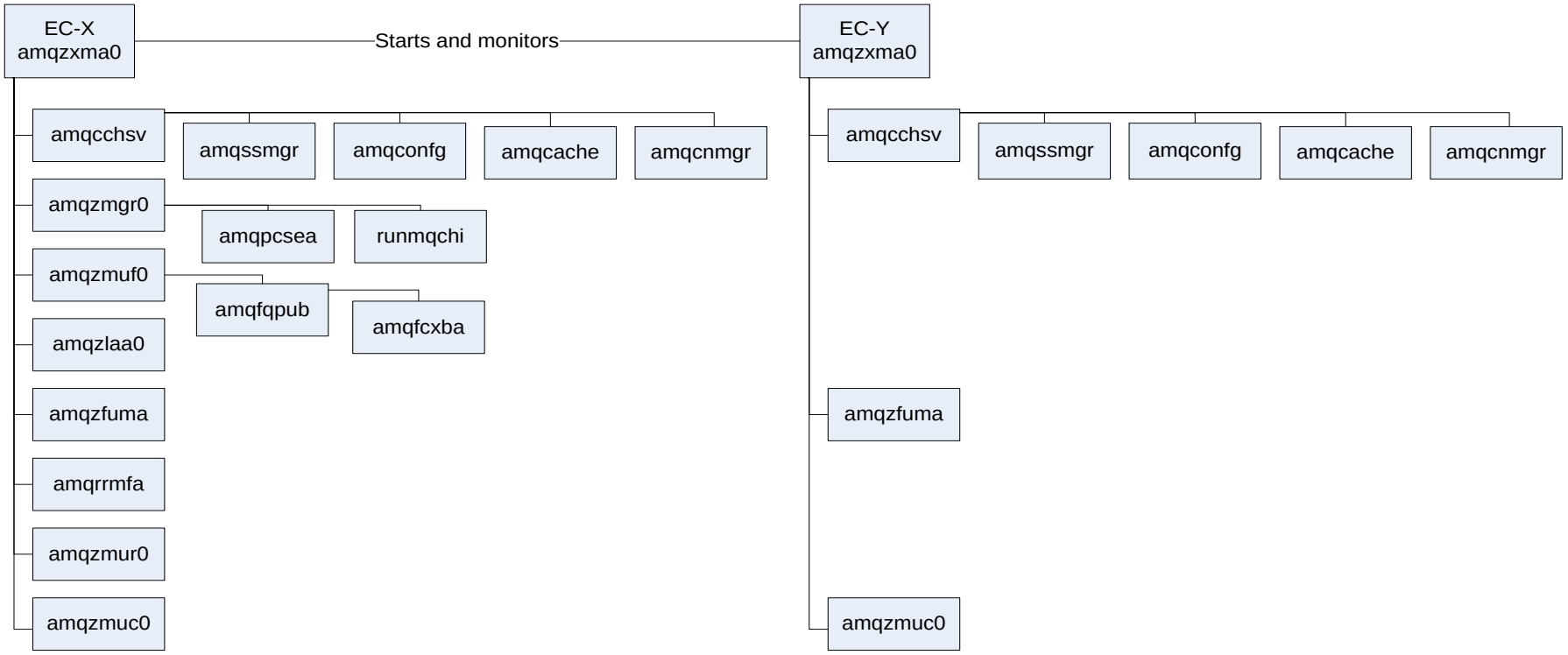
MQ 8 HA and DR

MQ High Availabilty: The MQ 8 Approach

- MQ 5: CPU consumption of backup queue server > 60 % of primary
- Checkpointing in MQ 5 is expensive …
- … for the really rare case of a CPU down event
- MQ 8: No checkpointing, so total CPU usage less than MQ 5
- But pathlength still similar!
- MQ 5 checkpointing not without issues:
  - memory leaks in backup
  - failover in many cases not reliable
  - almost impossible to test

MQ 8 HA and DR

MQ High Availabilty: The MQ 8 Approach



MQ 8 HA and DR

MQ High Availabilty: The MQ 8 Approach

- Customer statement:
  "In most CPU down cases we had to restart the MQ 5 queue manager anyway"
- Trade off:
  - Lower total CPU use most of the time vs.
  - longer failover time in the (rare) CPU down case
- Improved takeover reliability
- Can practically restart queue manager without stopping application
  - Can be VERY helpful in critical situations
- Application must
  - Handle "connection broken" (2009) error by small delay and reconnect
  - Handle "queue manager not available" (2059) error by small delay and reconnect
- MQCONN may take somewhat long
- Queues will be "loaded" when takeover occurs
  - MQOPEN may take longer
  - High disk usage expected
  - Only an empty queue is a good queue!

MQ 8 HA and DR

MQ NonStop Disaster Recovery: Disclaimer

- Disaster recovery on NonStop is not a supported MQ feature
- Everything you do, you do on your own risk
- Support will try to help – within limits
- Understanding the mechanisms sometimes helps…
- Professional services are available, if needed (for a fee…)

MQ 8 HA and DR

MQ NonStop Disaster Recovery: Basics

Goal

- Have a (dormant) queue manager on a backup system
- At any point in time being able to switch from primary to backup
- Queue manager on backup has up to date data and configuration
- Standard setup
  - Tool for replicating audited data (RDF, Shadowbase)
  - Tool for replicating OSS data (AutoSYNC)
  - Others setups (“nomadic disk”) not considered here

MQ 8 HA and DR

MQ NonStop Disaster Recovery: Components

- Audited files
  - Must be on same volume/subvolume on backup system
  - Qualified filenames (without node) are referenced in file content
- OSS files
  - opt tree and var tree
  - opt tree only changed by installation and upgrade
  - var tree has some dynamic data
- SSL certificates
  - Can be outside of var tree
- IP connections
  - Outgoing connections should not cause problems (if network ok)
  - Incoming connections: what is the IP of your backup system?

MQ 8 HA and DR

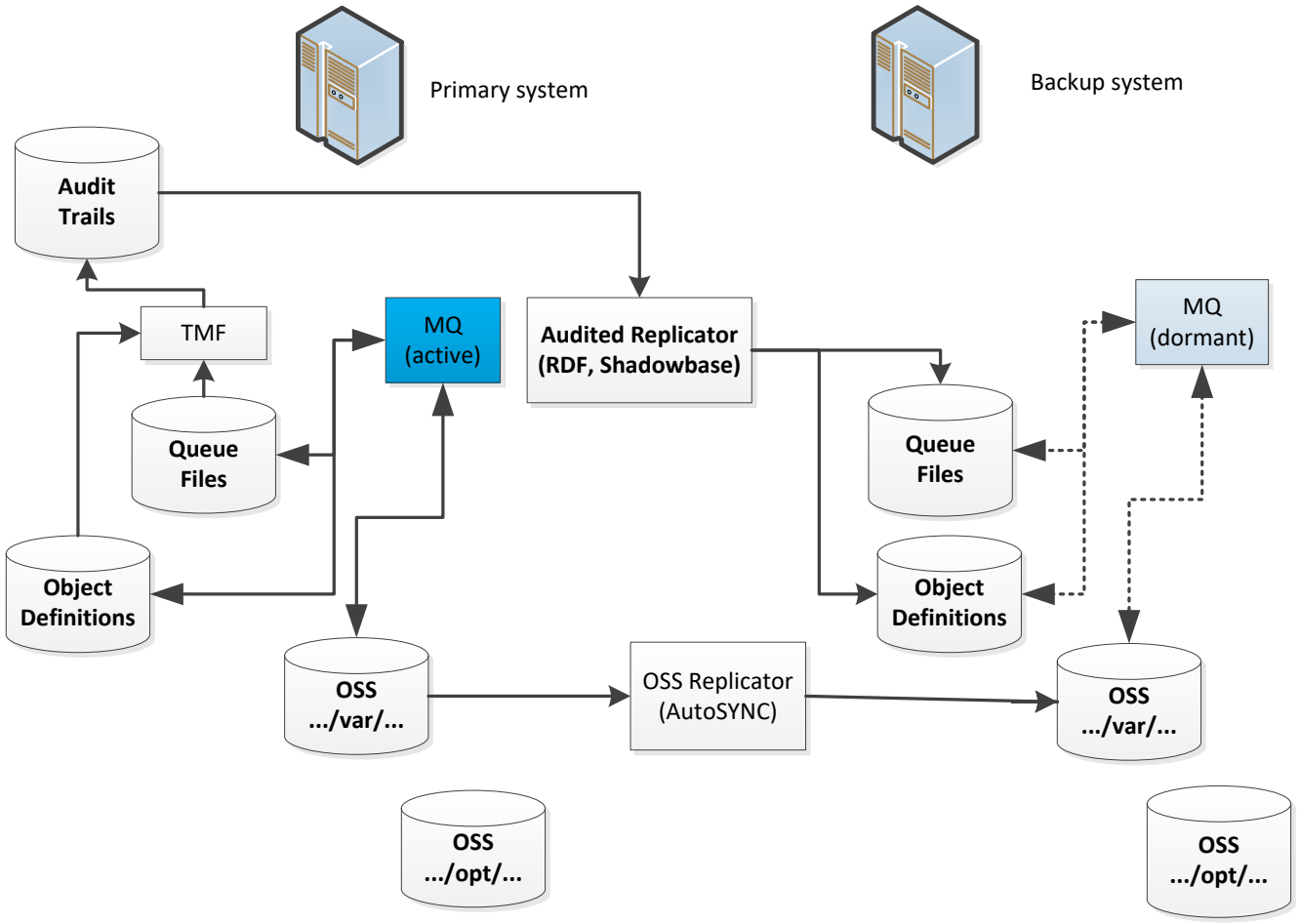
MQ NonStop Disaster Recovery: What is Where?

- Persistent queue messages: audited queue files
  - Replicator MUST be able to replicate file creations/deletions
- Non-persistent queue messages: cache manager process
  - Will always be lost with switch to backup system
- Configuration information from runmqsc: audited AMQOBJMD file
  - Object definitions
  - Global settings
  - One logical file missing from AMQOBJMD (see below)
- SSL certificates, keys, truststore
  - Do not change frequently
  - May be kept consistent using administrative process
- MQ settings in .ini files
  - blockaddr.ini recreated on queue manager start
  - mqs.ini (installation global) replicated via audited “shadow”
  - qm.ini replicated via audited “shadow”

MQ 8 HA and DR

MQ NonStop Disaster Recovery: What is Where?

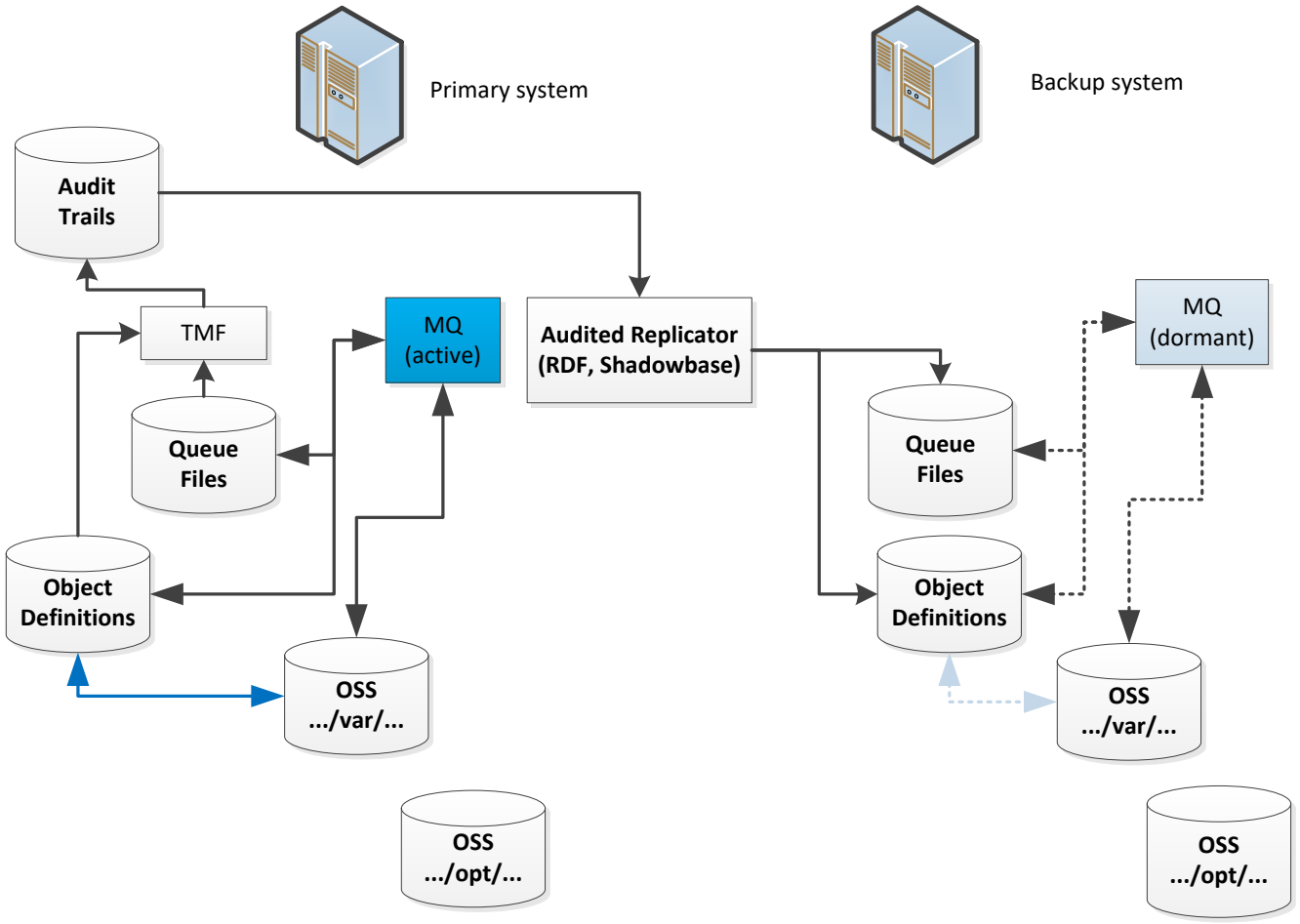
Before 8.1.0.2



MQ 8 HA and DR

MQ NonStop Disaster Recovery: What is Where?

After 8.1.0.2



MQ 8 HA and DR

MQ NonStop Disaster Recovery: The Audited Shadow

- Our goal: AutoSYNC not needed for MQ DR purposes
- Simplifies processes needed
- Eliminates potential issues when AutoSYNC synchronizes FDC files
- How it works for mqs.ini
  - When queue manager writes into mqs.ini, it also writes into audited AMQSINI
  - Queue manager monitors (in 1 minute interval) mqs.ini for manual changes
  - If mqs.ini timestamp is newer than entry in AMQSINI, AMQSINI is refreshed
  - If on startup the audited entry is newer than OSS entry, OSS refreshed from audited
- How it works for qm.ini and SSL files
  - Queue manager monitors (in 1 minute interval) files
  - If OSS timestamp is newer than entry in AMQOBJMD, AMQOBJMD is refreshed
  - If on startup the audited entry is newer than OSS entry, OSS refreshed from audited

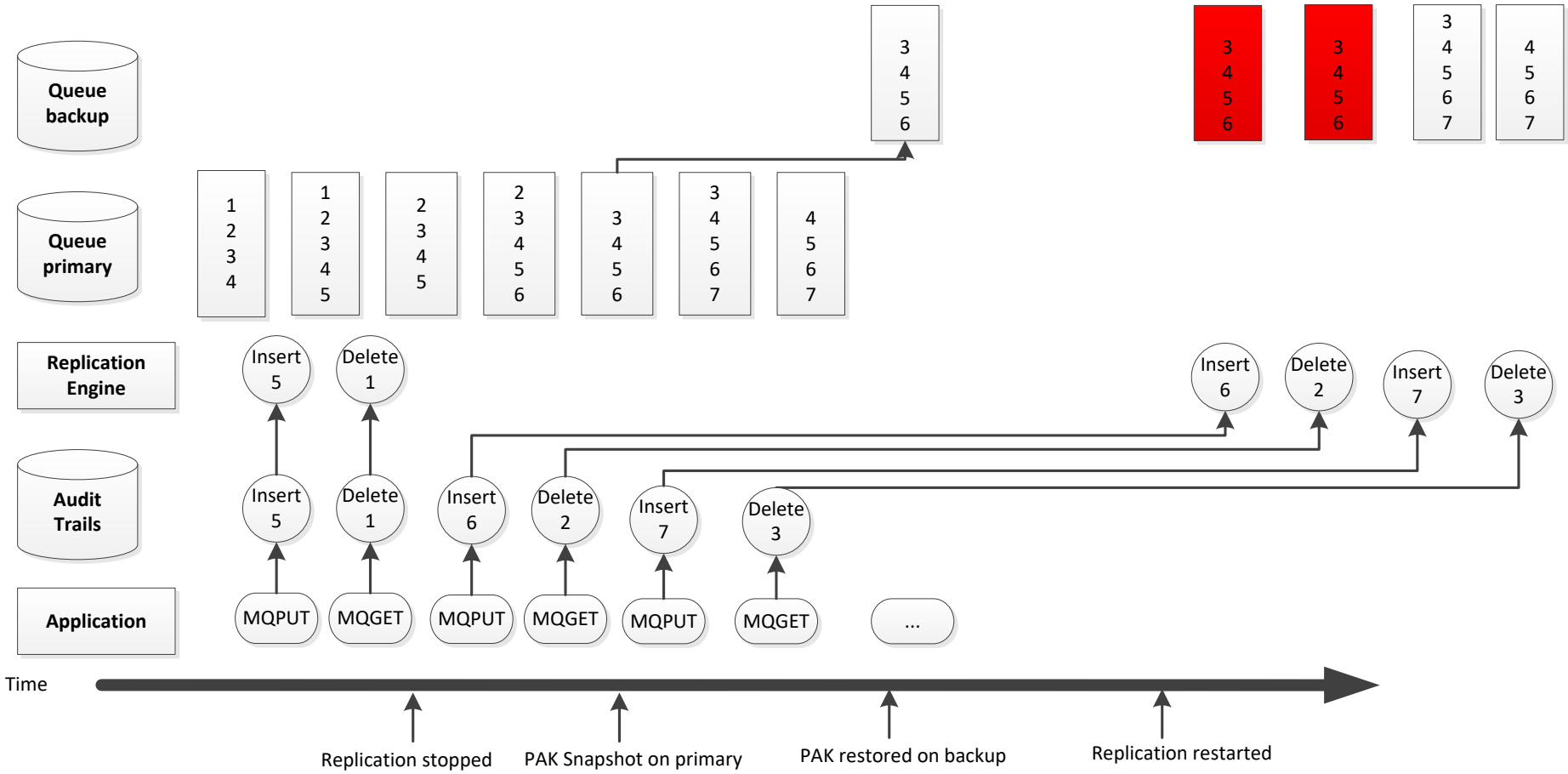
MQ 8 HA and DR

MQ NonStop Disaster Recovery: Steps Needed

1. Ensure that backup system has same disk names as primary
   - SMF disks may be used
2. Install MQ on backup system to same locations as on primary
   - OSS path and Guardian subvolumes must be identical
3. Delete complete var tree on backup system (if 8.1.0.2 and no AutSYNC)
4. Configure audited replicator to replicate all MQ subvolumes
5. Make sure, file creations and deletions are replicated
6. If MQ already running on primary, take PAK snapshot of subvols
   - Include installation subvol, queue manager subvol(s) and all queue subvols
   - Snapshot may be inconsistent – no problem
7. UNPAK snapshot on backup system
8. Start replication
9. Ignore certain errors from replication (see below)

MQ 8 HA and DR

MQ NonStop Disaster Recovery: Replication Startup Inconsistencies



MQ 8 HA and DR

MQ NonStop Disaster Recovery: The mqs.ini Dilemma

- File mqs.ini contains info about created queue managers
- … so is maintained by the system
- But it can also have user defined entries: default queue manager
- On strmqm MQ will overwrite mqs.ini from AMQSINI
- … to make queue managers visible on backup system
- So user defined entry will be overwritten!

MQ 8 HA and DR

MQ NonStop Disaster Recovery: The mqs.ini Dilemma Solution

- MQ remembers the NonStop Expand node name where queue manager was last started
- If the host name did not change, the OSS mqs.ini is always winning
- If the host name changed, the Guardian (replicated) AMQSINI wins
- Check is done on first strmqm on “backup” (now active) system
- … which also updates the node name
- With this scheme results are consistent!

MQ 8 HA and DR

MQ NonStop Disaster Recovery: Starting MQ on Backup

1. Ensure that replication direction is correct (off or backup is source)
2. If IP address of backup same as primary: Ensure primary listeners are down
3. Ensure that queue managers are not running on other system
4. Ensure that IP connectivity to all remote MQ installations work
5. If using SSL: Ensure that the correct SSL certificates are in place
6. If AutoSYNC is used: dspmq should display queue managers
7. Use strmqm to start your queue managers

   This should work, even if no AutoSYNC in use!
8. If backup has different IP address:
   1. Ensure that remote queue managers know about that
   2. If using MQ Explorer: Ensure that MQ Explorer knows about that
9. Start listeners and channels
10. If (previous) primary system up: Start replication in correct direction

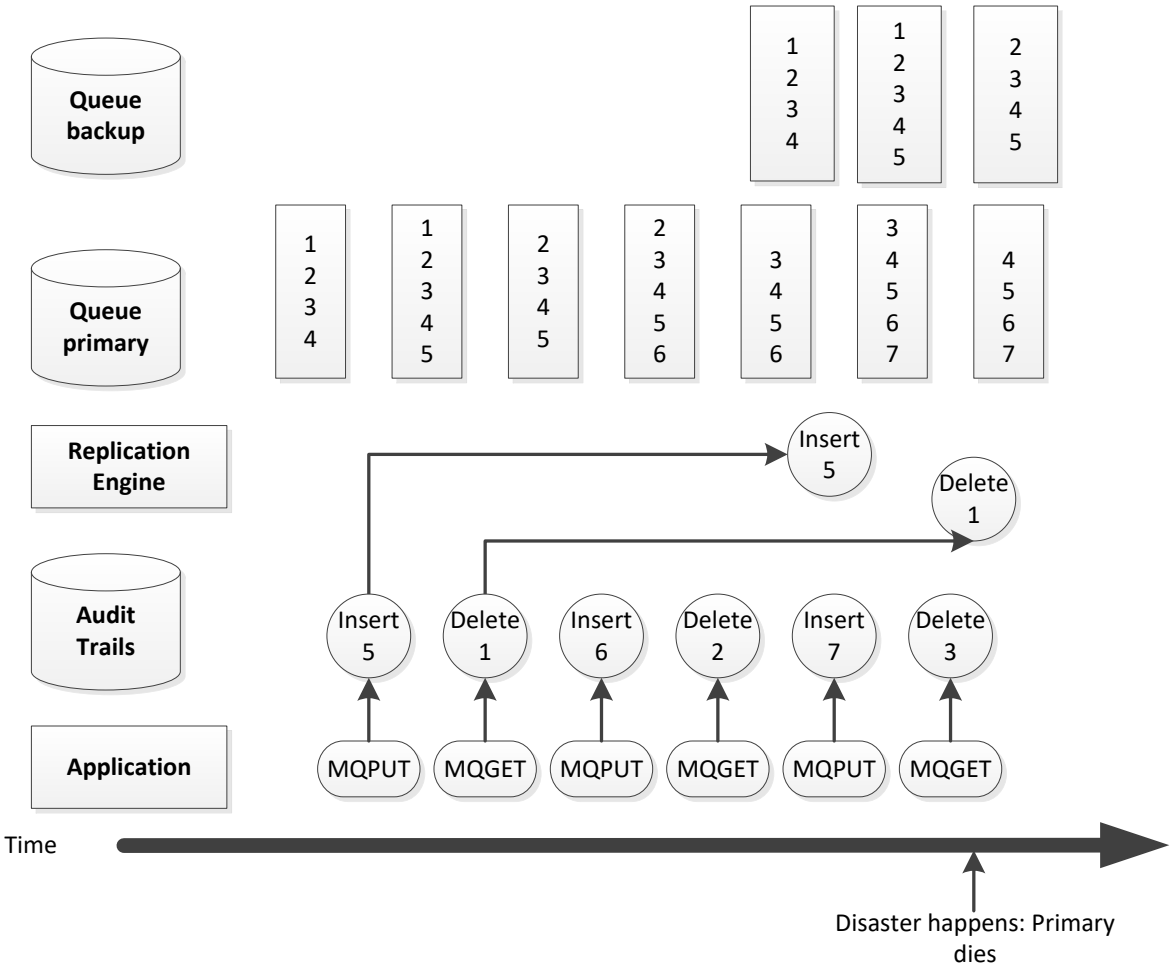
MQ 8 HA and DR

MQ NonStop Disaster Recovery: AMQRSYNA.DAT

- AMQRSYNA.DAT is an OSS file changed by the queue manager from time to time
- File is currently not transported via audited replication
- Consequences:
  - Error messages like “channel not found”
  - Error message mentions a repair tool, which is not delivered with MQ for NonStop
  - Repair tool is available from IBM on request…
  - … works and is tested on NonStop
- With version/fixpack 8.1.0.2 tool will be shipped with product…
- … and AMQRSYNA.DAT will be replicated
- So tool will not be needed anymore!

MQ 8 HA and DR

MQ NonStop Disaster Recovery: Inconsistencies



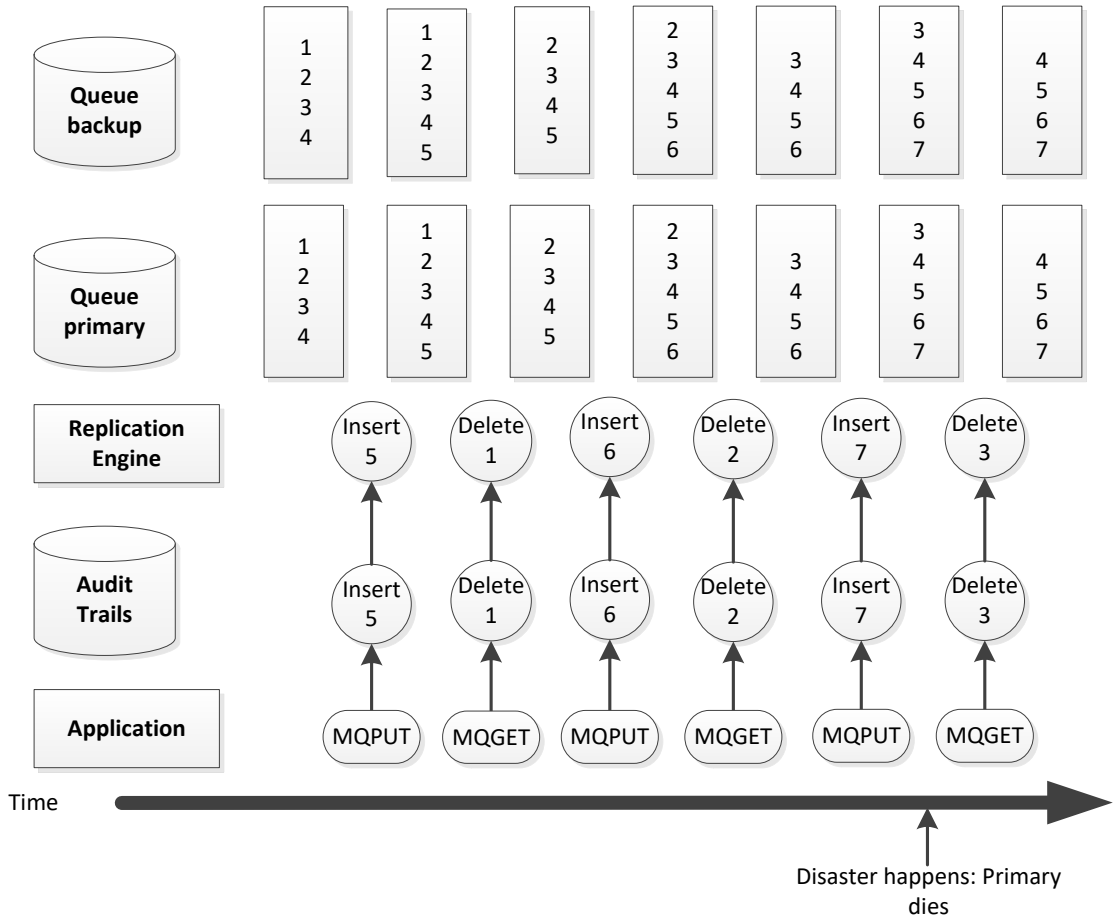
MQ 8 HA and DR

MQ NonStop Disaster Recovery: Backlog Consequences

- Messages may be lost
  - No surprise, as these were not transported in replication stream
  - Messages 6 and 7 do not appear on backup
- There may be duplicate messages
  - Duplicates will typically occur on remote systems
  - Caused by MQGETS from transmit queues not yet replicated
  - If message 2 was already transported to remote system, it may appear as duplicate!
- Channels may need to be reset
  - Message sequence number updates in audited files may be lost
- Note:
  - RDF/Shadowbase/TMF give local consistency, but not across MQ network
  - MQ has its own distributed consistency model

MQ 8 HA and DR

MQ NonStop Disaster Recovery: ZDL/ZLT



MQ 8 HA and DR

MQ NonStop Disaster Recovery: ZDL/ZLT

- Technology keeps primary and backup in synch
- Synchronisation happens at ENDTRANSACTION time
- MQ “sees” a message (or a delete) only after ENDTRANSACTION
- Technology keeps MQ in synch – even within MQ network
- But: ENDTRANSACTION may take significantly longer
- No testing done so far with this technology
- Not our job – not part of the product offering!

MQ 8 HA and DR

MQ NonStop Disaster Recovery: Never do This

- Create queue managers on backup system
  - May cause unresolvable conflicts
  - If needed, create a second installation in a different location on same system
- Start queue manager on backup system
  - With RDF lots of errors (files exclusively open by RDF)
  - With Shadowbase may result in unresolvable conflicts
  - MQ does not know about primary or backup system
- Concurrently start same queue managers on primary and backup
  - Technically not prevented, when RDF is stopped
  - Can seriously confuse other MQ instances
  - Can do serious damage to your MQ cluster
- Change any .ini files on backup
  - After takeover first starte queue managers, then .ini can be changed
  - Start of queue manager will make .ini files up to date
- Move around .ini files
  - The shadowing technology is based on change timestamps
  - These may not be as desired when moving files around
- Use non-default location for mqs.ini
  - You can do that, but will then still need AutoSynch
  - In general this is a somewhat dangerous option

MQ 8: runnscnf Demystified

A NonStop Specific Tool

- The tools covers three areas of NonStop specific activities:
- Configuring NonStop objects
- Viewing physical queue file names
- Online queue file migration

MQ 8: runnscnf Demystified

Understand the concept – then the tool!

- Class: A class is just a name for a collection of related objects
  - A class exists if and only if there is at least one object in the class
  - There is no need to manage classes in any way
  - Classes are pre-defined by the product

- Object: An object is just a “thing” having properties
  - Each object belongs to a class
  - An object exists if and only if there is at least one configured property
  - There is no need (or way) to define or manage objects
  - Some objects are pre-defined (like CurrentQmgr), others are user defined
  - Objects can be just about anything: queues, channels, processes etc.
  - Each object is identified by its class and its name

- Property: An attribute defined for an object
  - The class of an object defines which properties are available for the object

MQ 8: runnscnf Demystified

Understand the concept – then the tool!

- runnscnf can be used when no queue manager is running
- Concept: Configure first, then start queue manager
- Nothing **must** be configured, but many things **can** be changed
- MQ always has (and uses) defaults if nothing else given
- Biggest areas of interest:
  - Certain process types (scalability)
  - Queues (distributing load to different disks)
- Classes for processes (among others):
  - CacheManager
  - SetSignalManager
- Not everything runnscnf shows, can be changed
  - Example: Processes register their names when they start, so that MQ can find them
- A preferred primary and backup CPU can be configured for process pairs
- A list of allowed CPUs can be configured for processes
- The allowed CPUs are strictly obeyed – primary and backup only if available

MQ 8: runnscnf Demystified

Understand the concept – then the tool!

runnscnf does not actively create something, it only describes what to do, if MQ itself creates an object

```
runnscnf Command Interface
5724-H72 (C) Copyright IBM Corp. 1994, 2014.

NSCNF>class CacheManager
CLASS set to CacheManager
NSCNF>object CACHE.MAN.1
OBJECT set to CACHE.MAN.1
NSCNF>set primaryCPU 4
Property PrimaryCPU set to 4

NSCNF>set backupCPU 5
Property BackupCPU set to 5

NSCNF>class queue
CLASS set to Queue
NSCNF>object Q.1
OBJECT set to Q.1
NSCNF>set CacheManager CACHE.MAN.1
Property CacheManager set to CACHE.MAN.1
```

MQ 8: runnscnf Demystified

Understand the concept – then the tool!

- Effects:
- If a queue named Q.1 is created with runmqsc, non-persistent messages for this queue will be managed by cache manager CACHE.MAN.1
- CACHE.MAN.1 will be started only if and when needed
- After creating Q.1 and putting some non-persistent messages, the process name for CACHE.MAN.1 shown in runnscnf

MQ 8: runnscnf Demystified

Understand the concept – then the tool!

```
NSCNF> list ('*','CA*','*')
-------------------------------------------------------
Class:                  CacheManager
Object:                 CACHE.MAN.1
Property:               BackupCPU
Value:                  5
-------------------------------------------------------
Class:                  CacheManager
Object:                 CACHE.MAN.1
Property:               PrimaryCPU
Value:                  4
-------------------------------------------------------
Class:                  CacheManager
Object:                 CACHE.MAN.1
Property:               ProcessName
Instance:               1
Value:                  $X0RKX:1315513661
```

MQ 8: runnscnf Demystified

Understand the concept – then the tool!

From TACL:

```
59> status $x0rkx

System \CS5

Process          Pri PFR %WT Userid  Program file  Hometerm
$X0RKX   X 4,517 158     017  47,1   amqcache      $ZTN0.#PTWNQ6L
                      Swap File Name: $OSS01.#0
$X0RKX   Y 5,958 158     013  47,1   amqcache      $ZTN0.#PTWNQ6L
                      Swap File Name: $OSS01.#0
```

Thank you!

Questions?

Dr. Werner Alexi
CS Software GmbH
Schiersteiner Straße 31
D-65187 Wiesbaden
+49 611 890 85 55
alexi@cs-software-gmbh.de