WebSphere Application Integration (MQSeries) 家族產品藍圖與創新技術介紹

Charlie Tseng

WebSphere MQ Overview

What does WebSphere MQ provide ?

Reliable, Loosely-coupled, Easy to use

Exactly once message delivery

Loosely-coupled applications

- Time-independent Asynchronous messaging

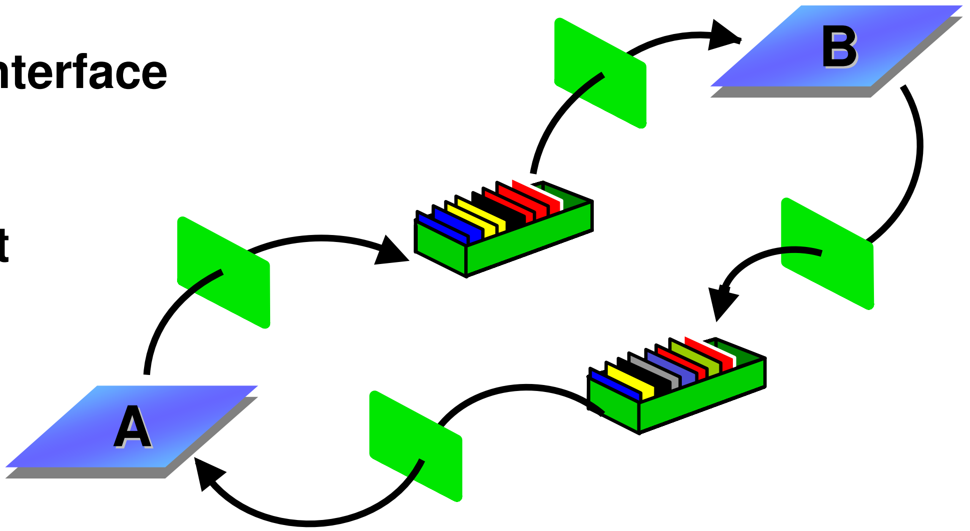
A single, multi-platform API

- Easy to use ... message centric interface
- Network independent

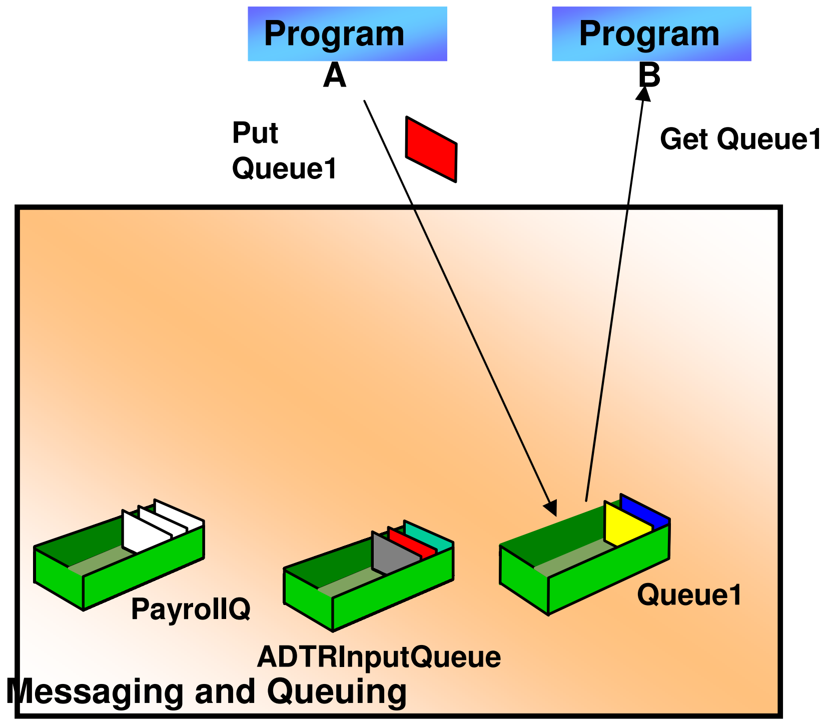
... faster application development

Universal …

runs everywhere



WebSphere MQ example - Local



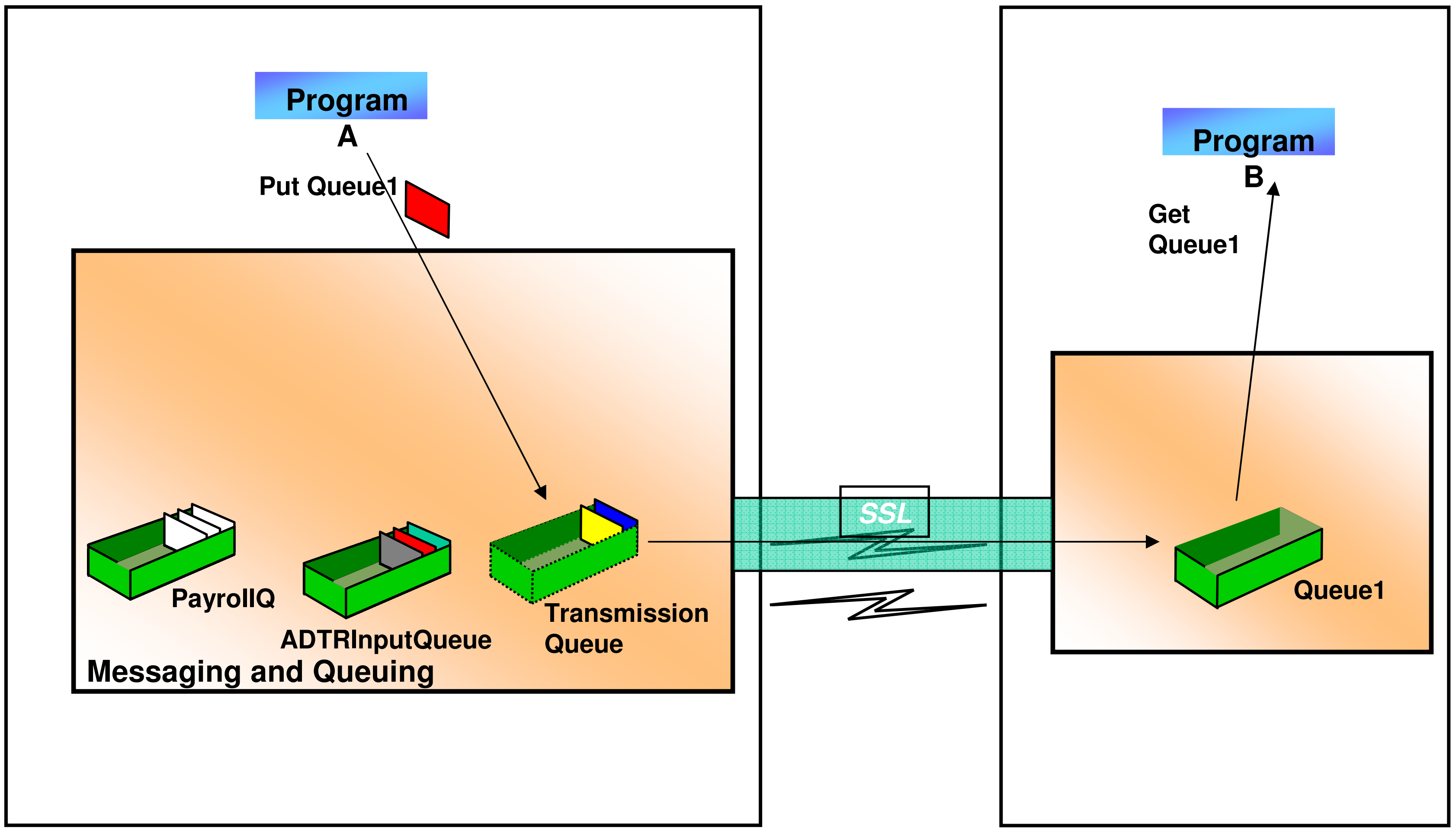
Accept Message

- Receive message from application
- Manage "unit of work"

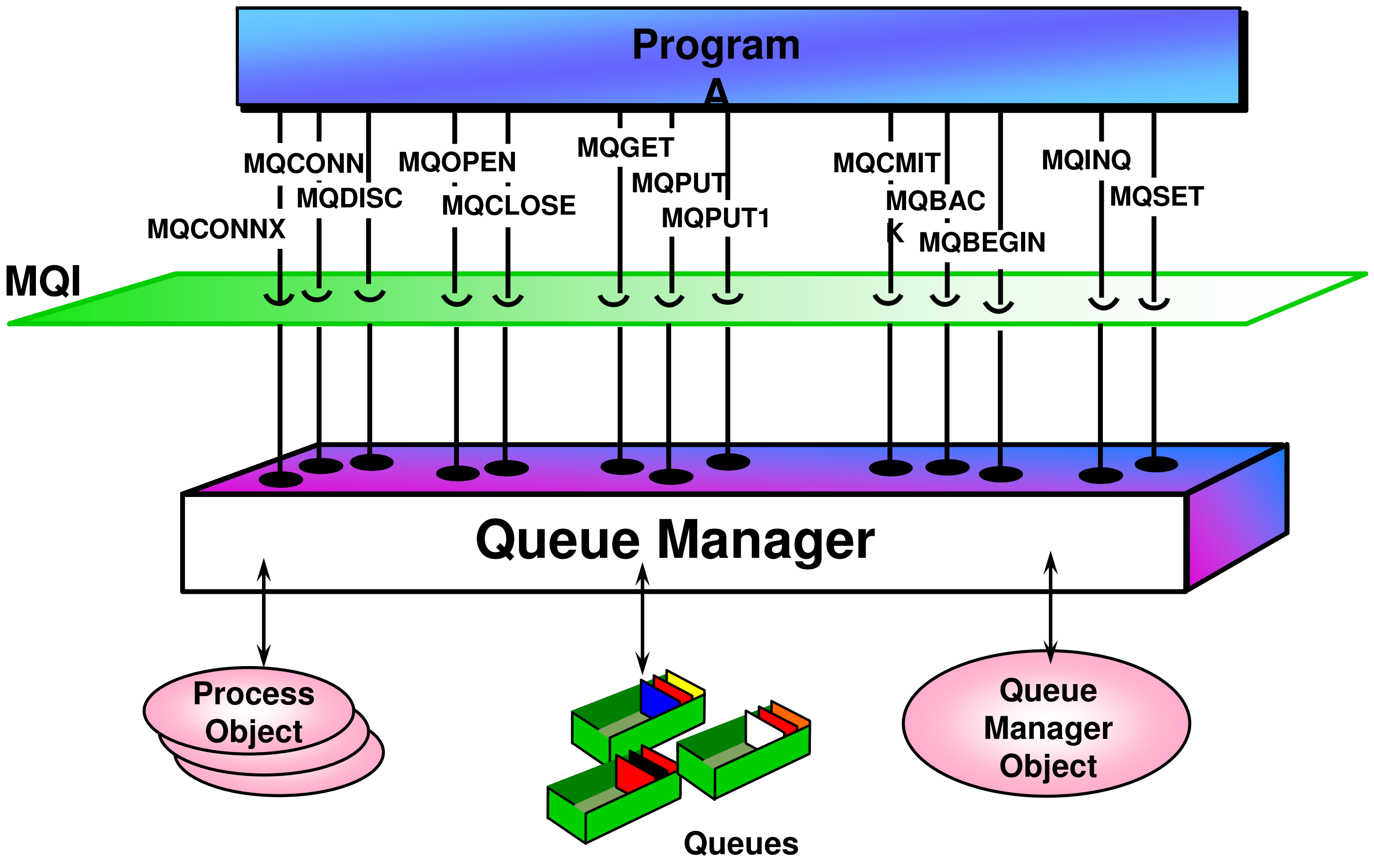
Deliver Message(s)

- Deliver message to application
- Ensure Exactly Once Delivery (even after a failure)
- Manage "unit of work"

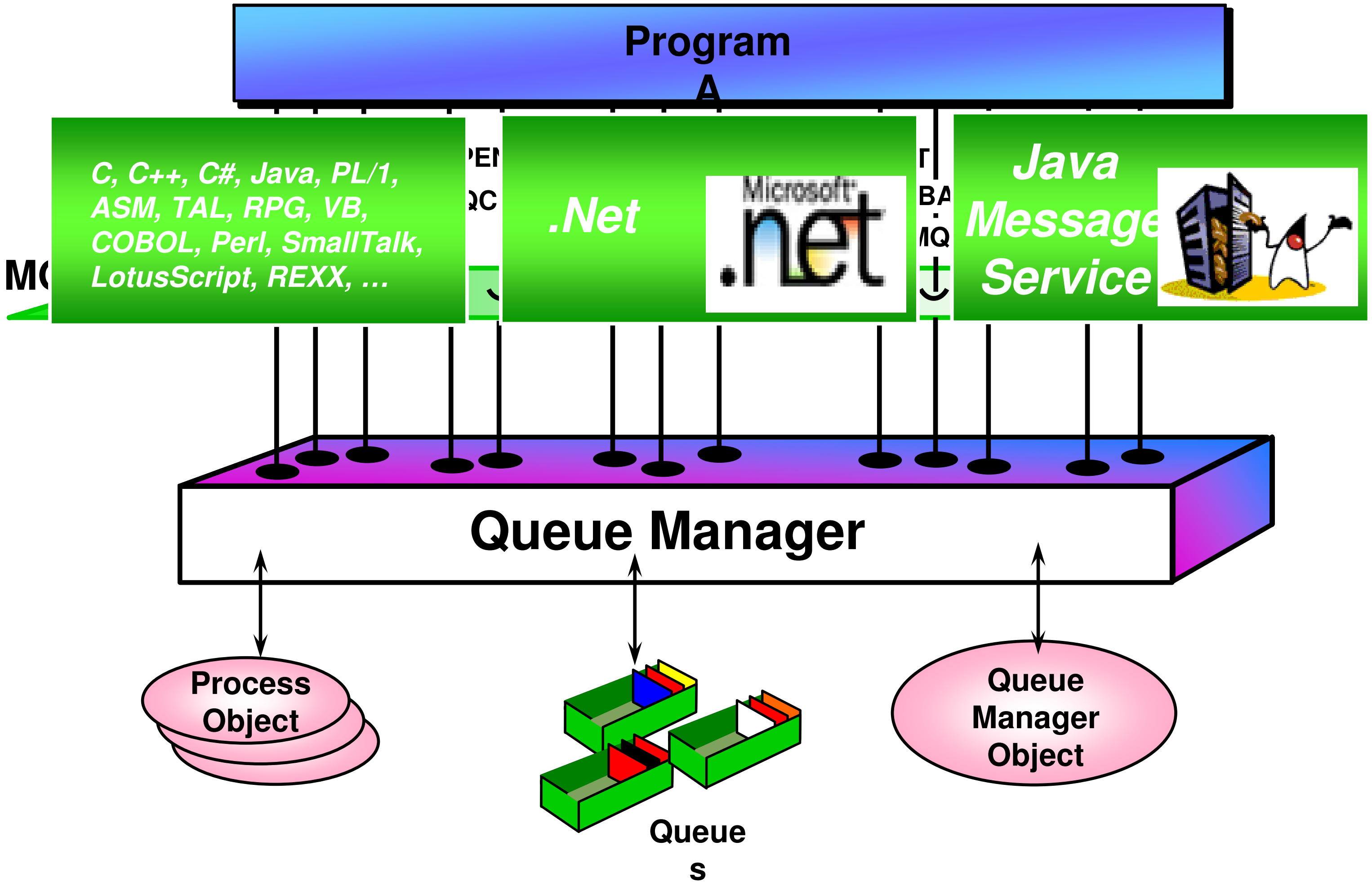
WebSphere MQ example (remote) - WAN



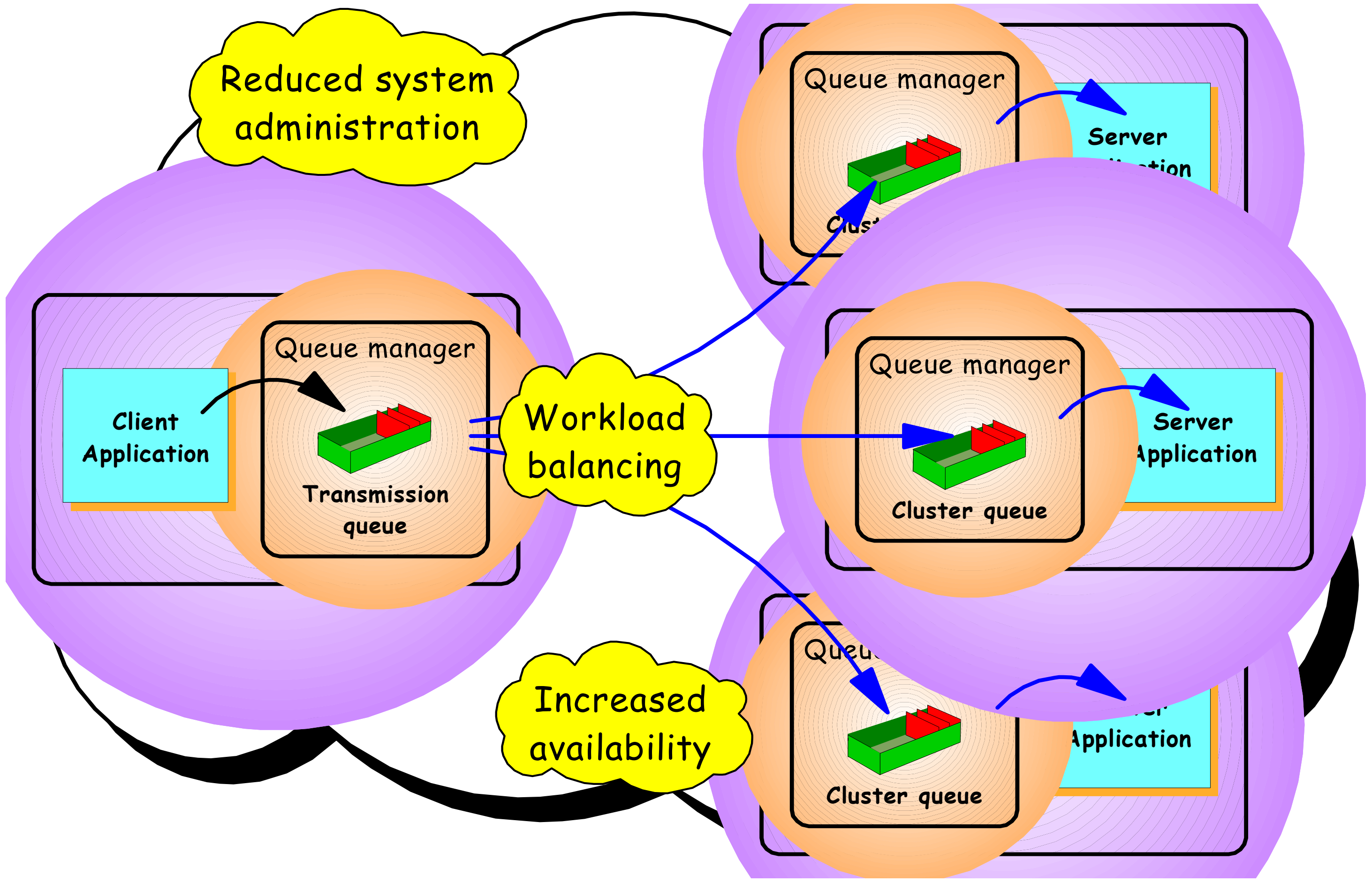
Easy for application programmers to use



Supports popular application environments

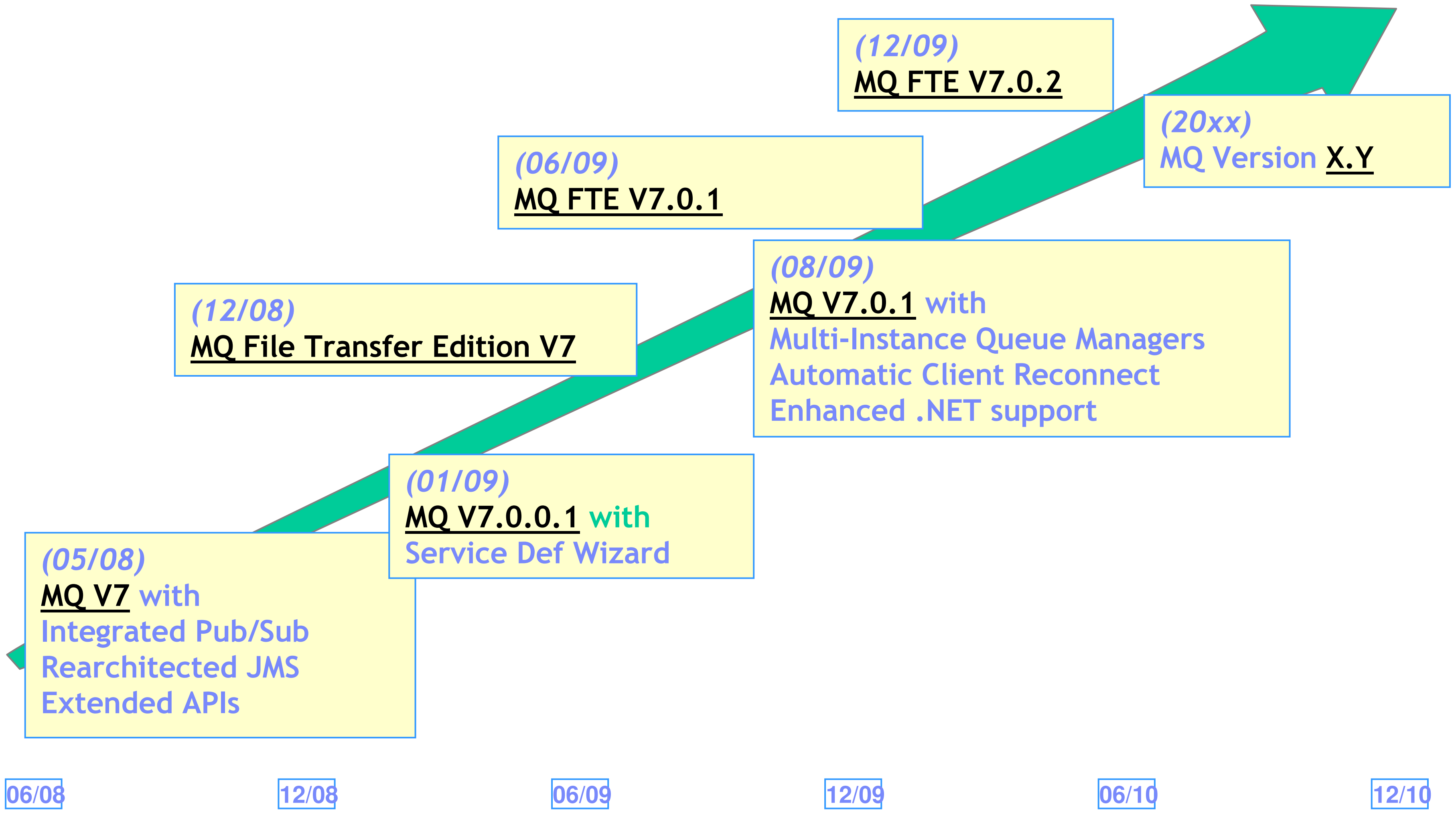


WebSphere MQ Clustering



Installation / Upgrade / Migration

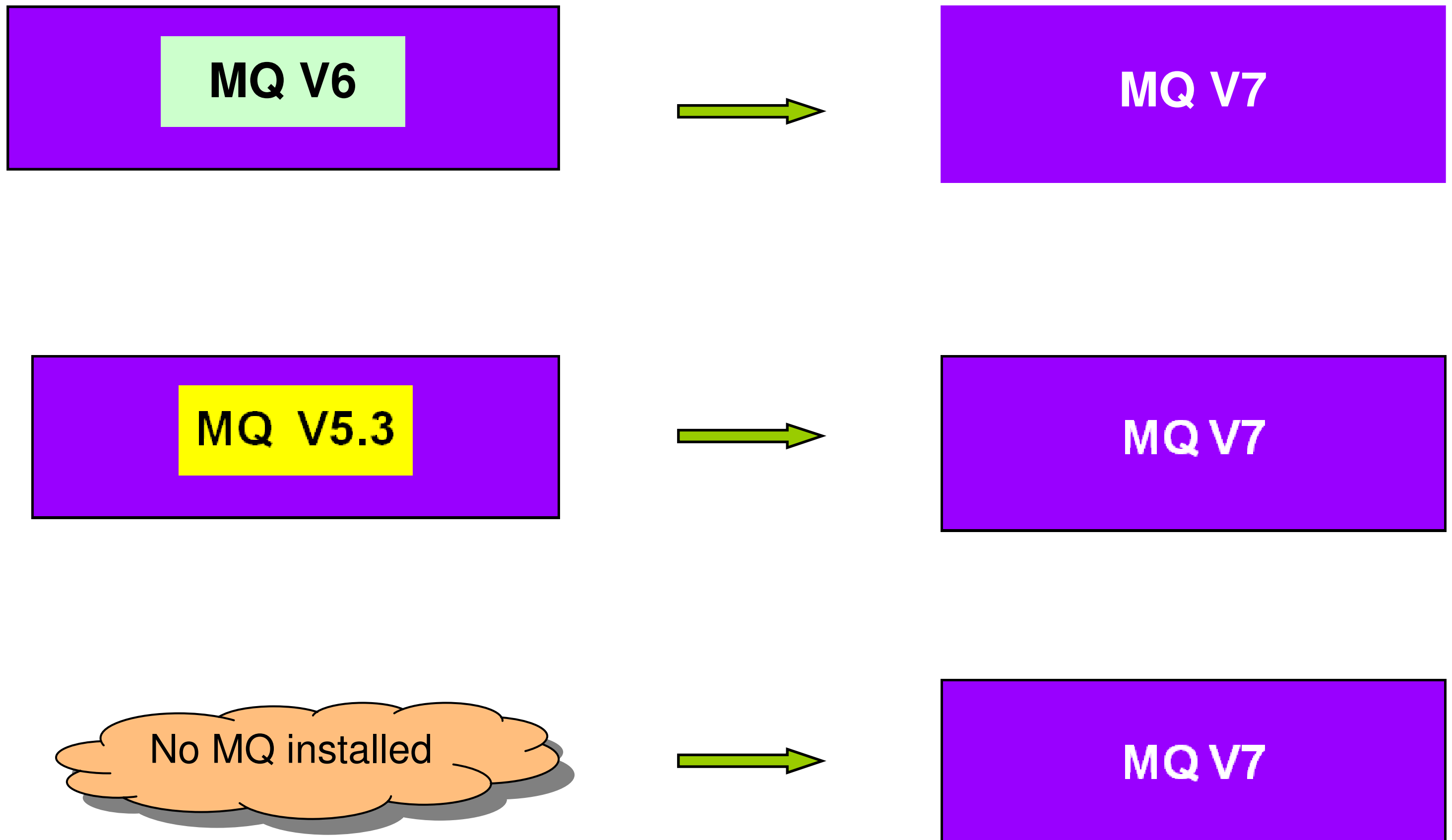
WebSphere MQ Delivery Roadmap



Installation and Delivery

- WMQ V7.0.1 is a modification release on the V7 base
  - Which means limited scope for new objects/attributes
  - Minimises migration aspects
- On Distributed platforms, it is available in two ways
  - A fixpack for upgrade from existing V7 installations (which can be backed out)
  - Now ordering V7 will now get V7.0.1
- Ease of Upgrade / Migrate
  - Easy to migrate from V5.3 or V6 (Installation Wizard)

MQ V7 installation path



IBM WebSphere MQ V7 New Enhancement Summary

- Integrated Publish/Subscribe
- Extended APIs
- Multi-Instance Queue Managers
- Automatic Client Reconnect
- Service Def Wizard

Integrated Publish/Subscribe

Publish/Subscribe

- Point-to-point asynchronous messaging decouples applications
  - But still implies a one-one relationship between sender and receiver
- Publish/subscribe is a further stage of decoupling
  - Sender has no direct knowledge of how many (if any) apps will see a message
  - Link between applications is a Topic, not a Queue
- A natural part of the JMS API
  - Combined both Publish/Subscribe and Point-to-Point styles
- Support for durable and non-durable subscriptions

- V5.3 and V6 (Distributed) included a Publish/Subscribe broker (need MA0C)

Integrated Publish/Subscribe Demo

Extended APIs

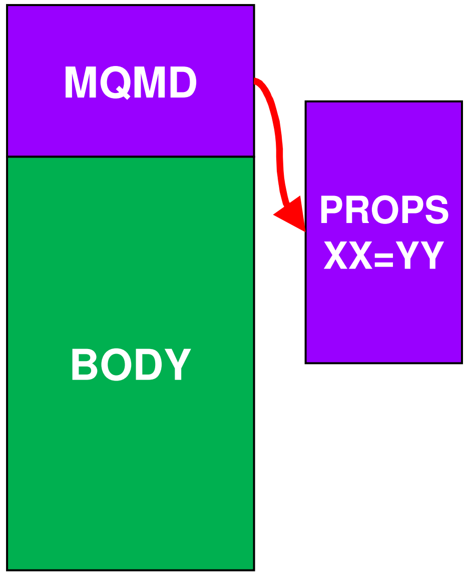
Publish/Subscribe using MQI - Summary

- The verbs used are:-
  - MQOPEN
  - MQPUT
  - MQSUB
  - MQGET
  - MQSUBRQ
  - MQCLOSE

- New structures to accompany new verbs
  - MQSUB – MQSD – Subscription Descriptor
  - MQSUBRQ – MQSRO – Subscription Request Options

Message Properties

- Arbitrary values associated with the message but not part of the body
  - Like a user-extendable MQMD
  - Already part of JMS
- New verbs **MQSETMP** and **MQINQMP**
  - Properties can be integers, strings, boolean, etc.
- Easier to use than RFH2 folders
  - Receiving apps do not see them unless they want
  - No need to parse and skip over message headers
- Appear as RFH2 properties on older queue managers



Other MQI Enhancements

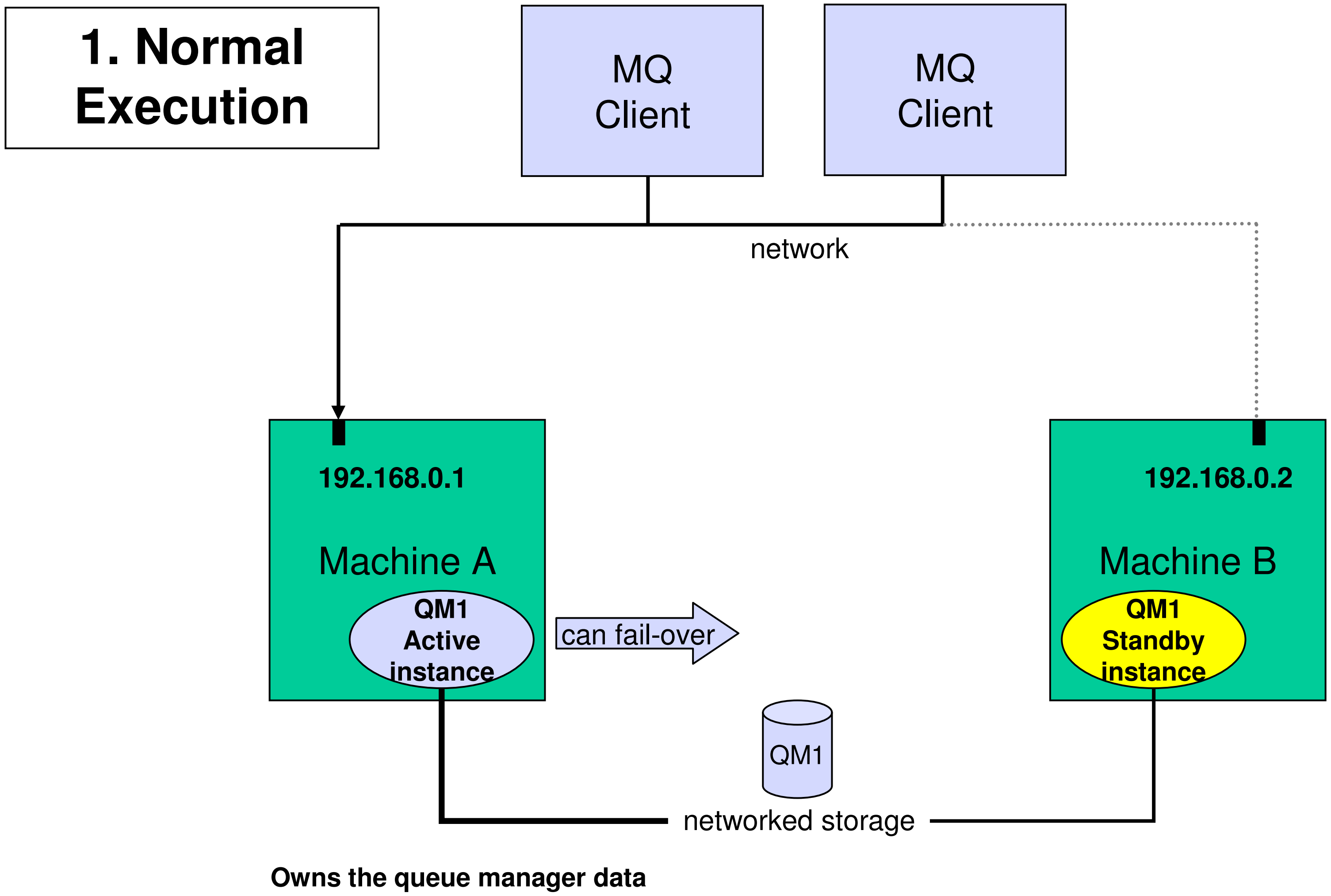
- Asynchronous Message Reception
  - New verb **MQCB** defines a callback function
  - Automatically Invoked when a message arrives
  - No need for MQGET(WAIT) or MQGET(SIGNAL)

Multi-Instance Queue Managers

Distributed Platforms: Multi-instance Queue Managers

- Basic failover support without HA coordinator
  - Faster takeover: fewer moving parts
  - Cheaper: no specialised software or administration skills needed
  - Windows, Unix, Linux platforms
- Queue manager data is held in networked storage
  - NAS, NFS etc so more than one machine sees the queue manager data
- Multiple (2) instances of a queue manager on different machines
  - One is “active” instance; other is “standby” instance
  - Active instance “owns” the queue manager’s files and will accept app connections
  - Standby instance does not “own” the queue manager’s files and apps cannot connect
- Instances share data, so it’s the SAME queue manager

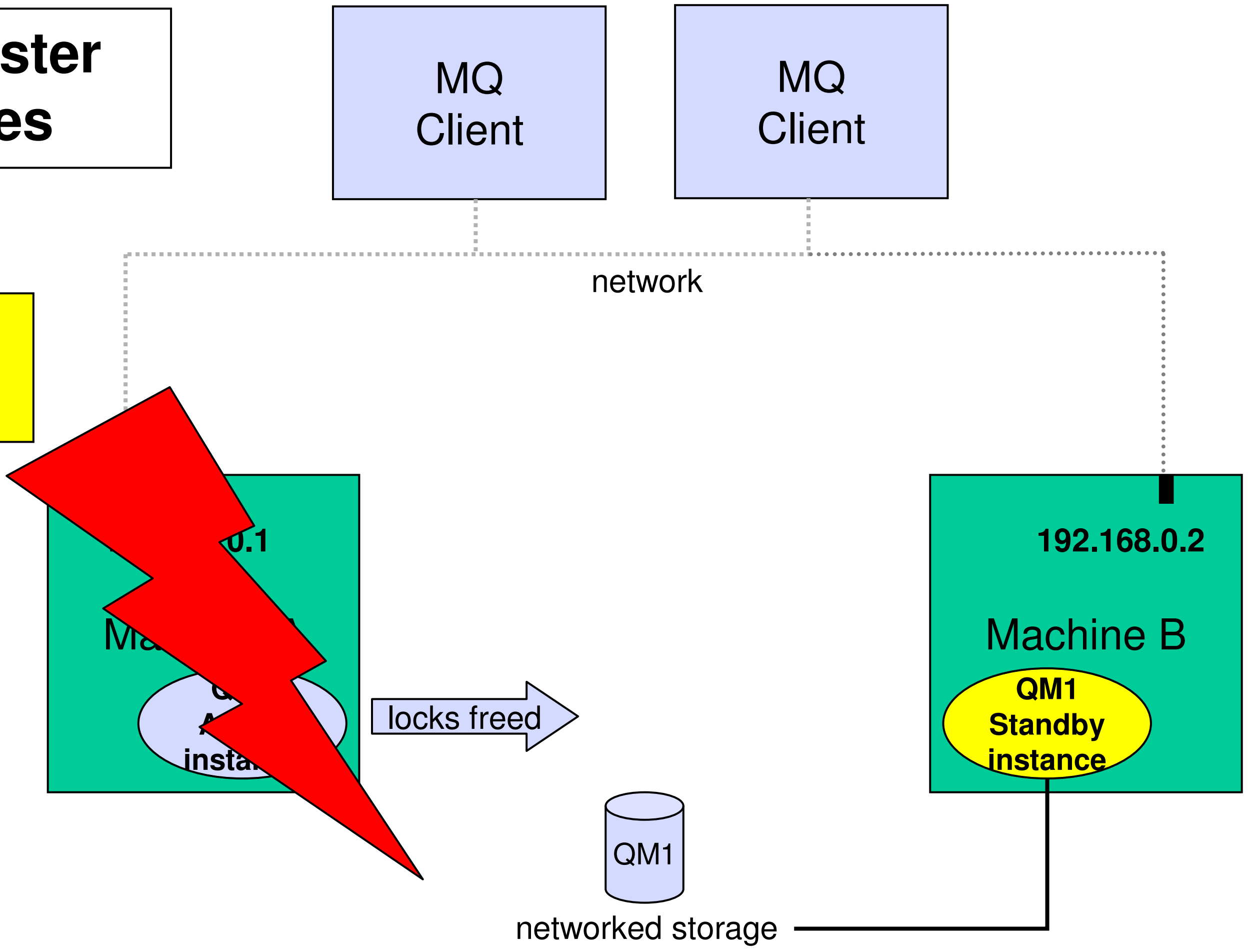
Multi-instance Queue Managers



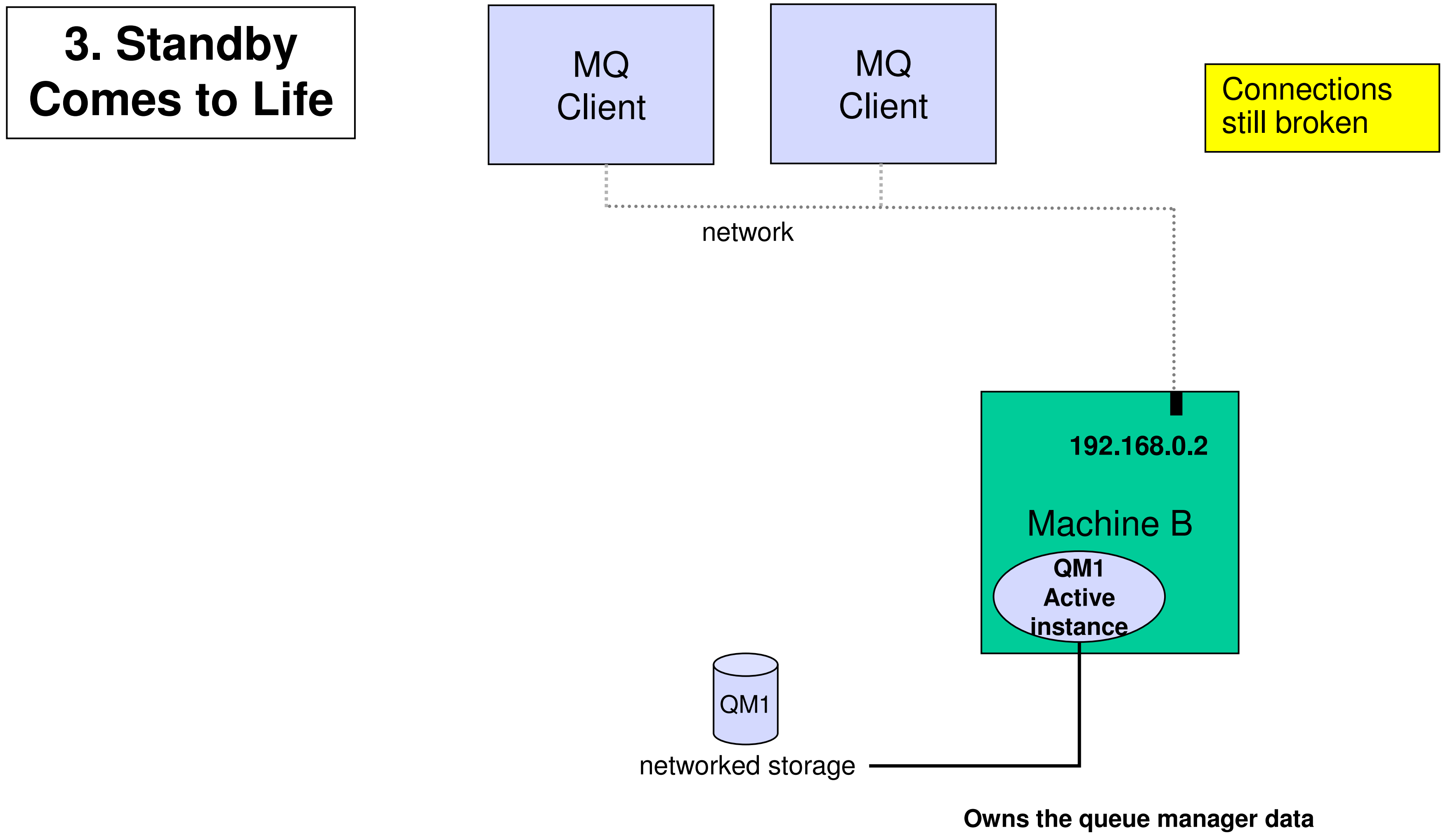
Multi-instance Queue Managers

2. Disaster Strikes

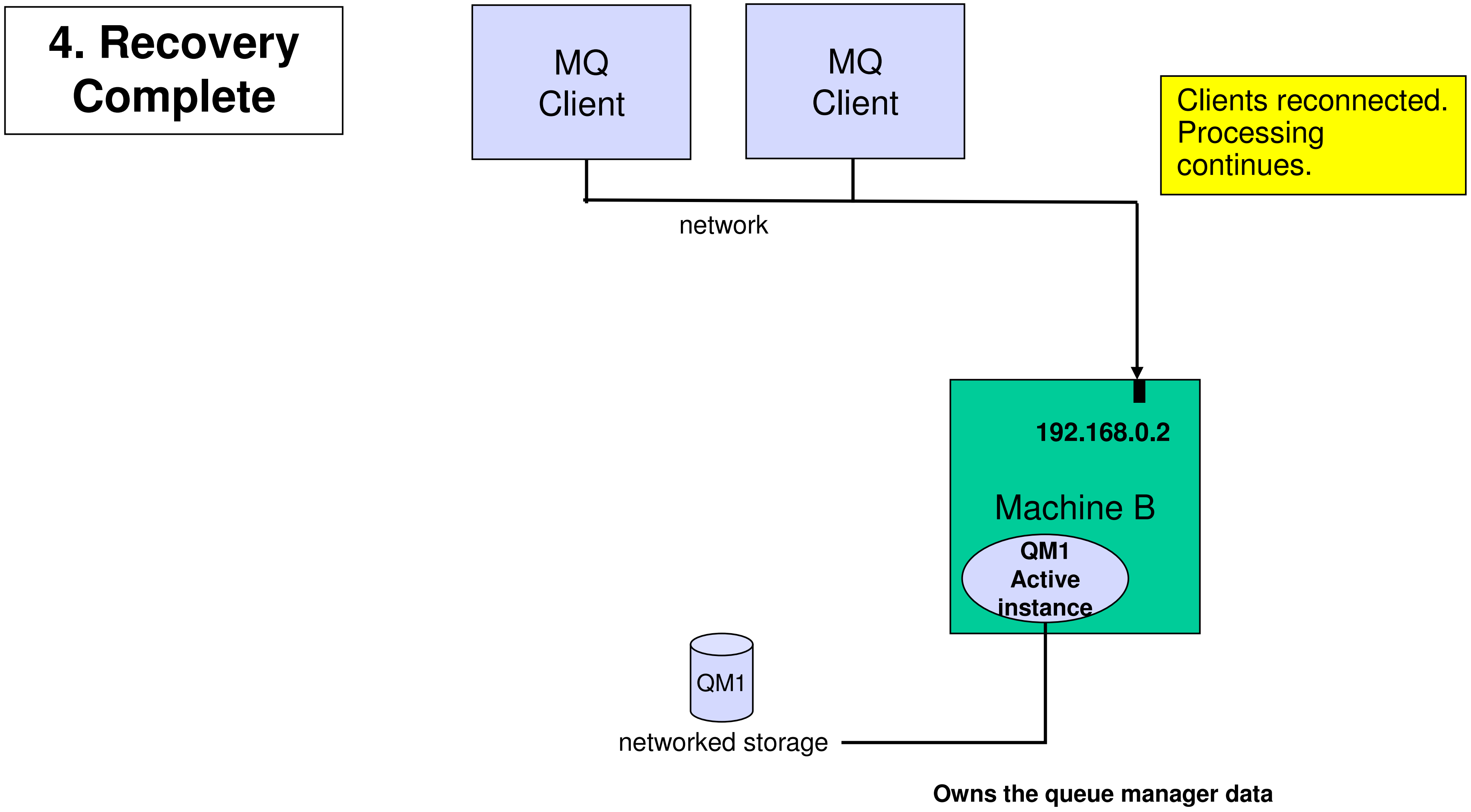
Connections broken from clients



Multi-instance Queue Managers



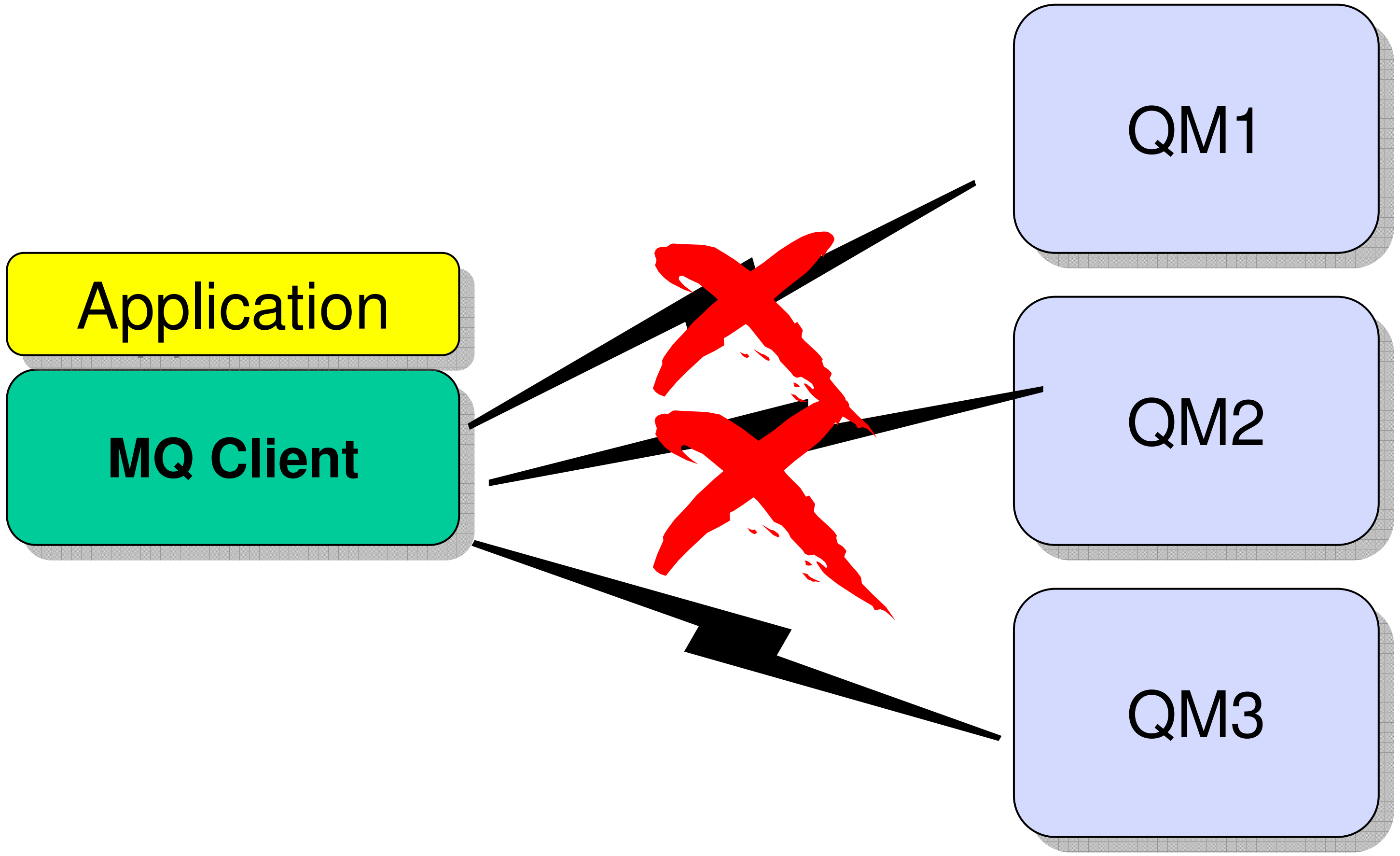
Multi-instance Queue Managers



Automatic Client Reconnect

Automatic Client Reconnection

- Client library provides necessary reconnection logic on detection of a failure
- Hides failure from application code



Automatic Client Reconnection

- Tries to hide queue manager failures by restoring current state automatically
  - For example, if MQPUT returns error, client reruns MQCONN/MQOPEN/MQPUT internally
- Uses the list of addresses in CONNAME to find queue manager
  - MQSERVER environment variable also understands list
  - MQSERVER=SYSTEM.DEF.SVRCONN/TCP/host1(1414),host2(1414)
- Can reconnect to the same or different Queue Manager
- Re-opens queues and other qmgr objects, re-establishes subscriptions

Automatic Client Reconnection: Details

- Enabled in application code or ini file
  - Event Handler callback shows reconnection is happening if app cares
- Tries to keep dynamic queues with same name
  - So replies may not be missed
- Not all MQI is seamless, but majority repaired transparently
  - eg a browse cursor would revert to the top of the queue, non-persistent messages will have been lost during restart, non-durable subscriptions may miss some messages, in-flight transactions backed out, hObj values maintained
- Some MQI options will fail if you have reconnection enabled
  - Using MQGMO_LOGICAL_ORDER, MQGET gives MQRC_RECONNECT_INCOMPATIBLE

IBM WebSphere MQ V7 New Enhancement Summary

- Integrated Publish/Subscribe
- Extended APIs
- Multi-Instance Queue Managers
- Automatic Client Reconnect
- Service Def Wizard

धन्यवाद
Hindi

多謝
Traditional Chinese

ขอบคุณ
Thai

Спасибо
Russian

Gracias
Spanish

Thank You
English

شكراً
Arabic

Merci
French

Obrigado
Brazilian Portuguese

Grazie
Italian

多谢
Simplified Chinese

Danke
German

நன்றி
Tamil

ありがとうございました
Japanese

감사합니다