WebSphere MQ An Introduction

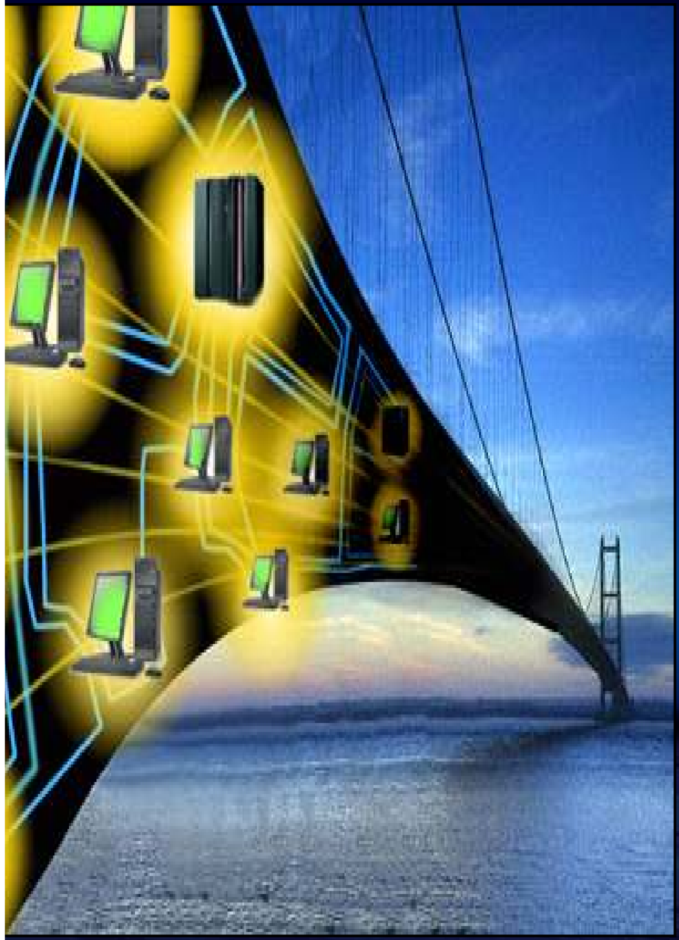
The Basis For The Advanced ESB

Agenda

- Overview of WebSphere MQ (Messaging and Queueing)
- Basic Concepts of WebSphere MQ
  - Messages
  - Queues
  - Queue Managers
  - Channels
  - Programming

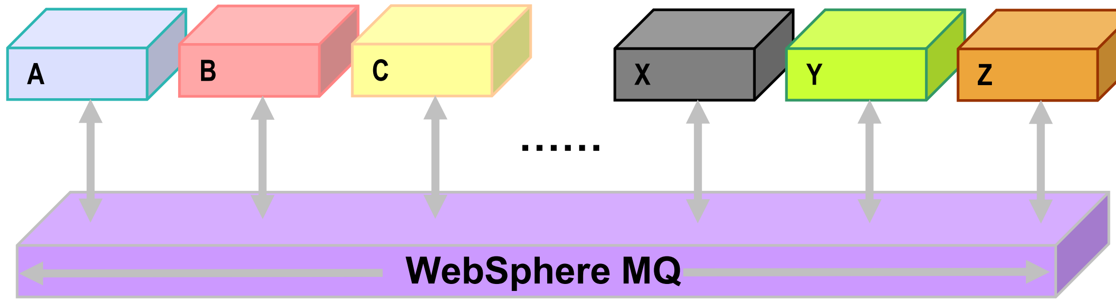
What is WebSphere MQ?

- A proven way of bridging between the components of your Service Oriented Architecture (SOA)
- Like a strong, broad bridge it robustly links your applications and your Web services
- It connects virtually any commercial IT systems
- Helping you to share and exchange critical business information with ease, confidence and security



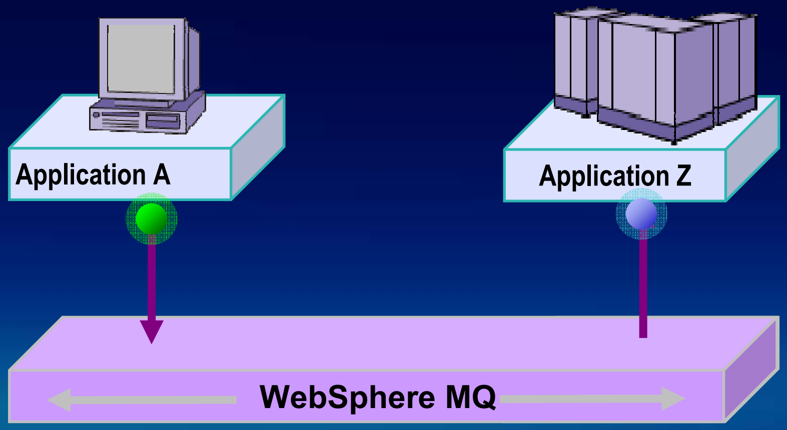
What fundamental problem does WebSphere MQ solve?

How to move information around…



What does WebSphere MQ do?

- Provides messaging services to applications and Web services that need to exchange data and events with:
  - Proven reliability
  - Transactional integrity
  - Consistency
  - Time independence
  - Ease and Speed
  - Flexibility
  - High-performance
  - Security
  - Scalability
  - Auditability

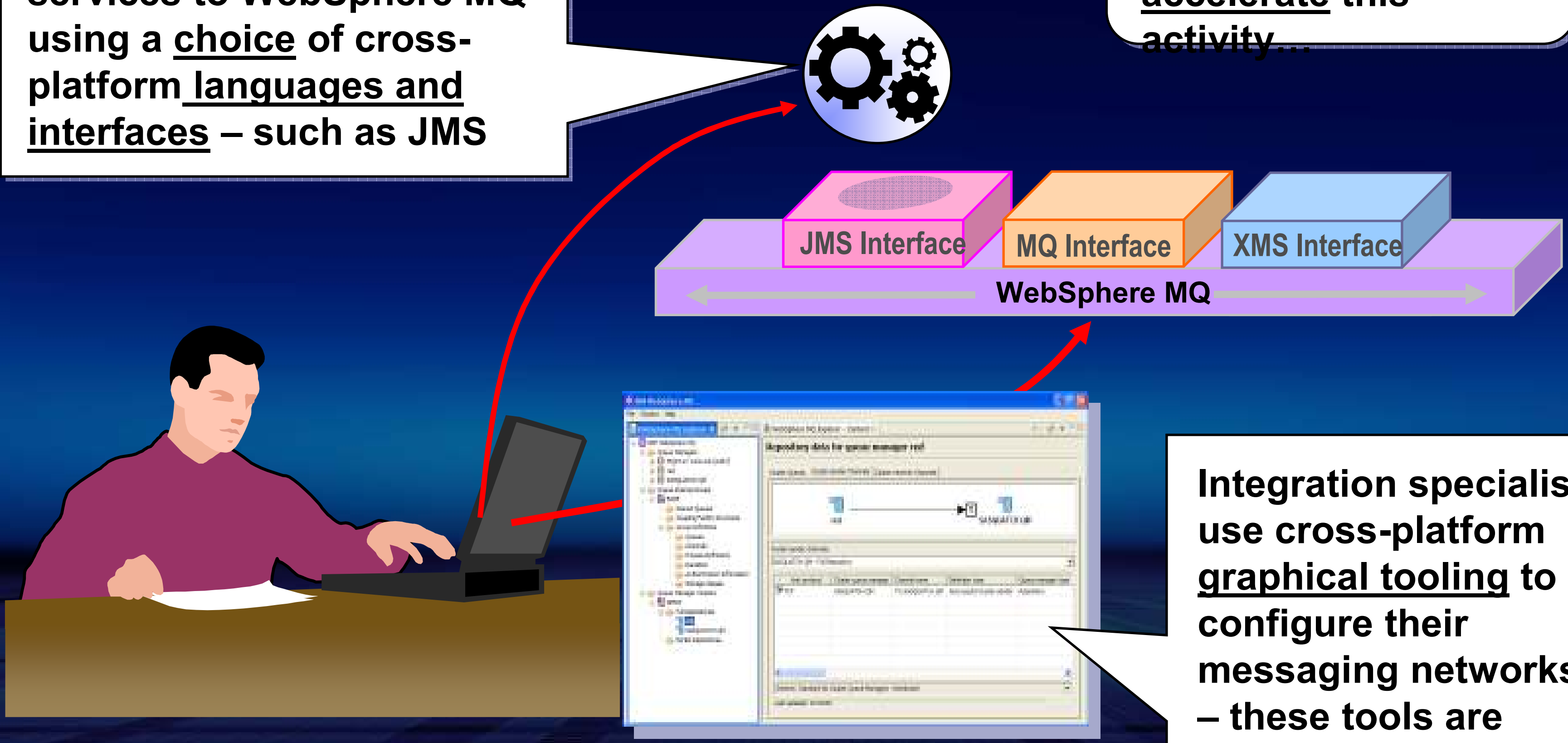


WebSphere MQ is like email for SOA applications …but email you can bet your business on

How do you use WebSphere MQ?

Developers attach applications and Web services to WebSphere MQ using a <u>choice</u> of cross-platform <u>languages and interfaces</u> – such as JMS

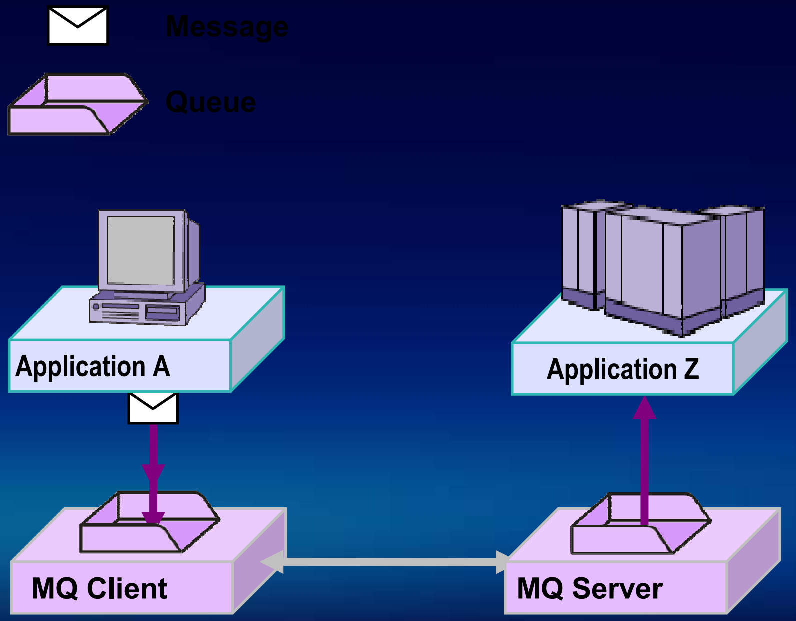
Application and technology adapters <u>accelerate</u> this activity…



Integration specialists use cross-platform <u>graphical tooling</u> to configure their messaging networks – these tools are based on open source Eclipse

How does WebSphere MQ work?

- Messaging services are based on **Queues** that store and forward data based on simple programming commands
- Uses the proven database technique of two-phase commit **transactions** to ensure messages are not lost or duplicated
- Uses **publish/subscribe** to route messages dynamically based on keywords or “topics”
- Uses multi-processor threading and **clustering** to accelerate throughput of messages

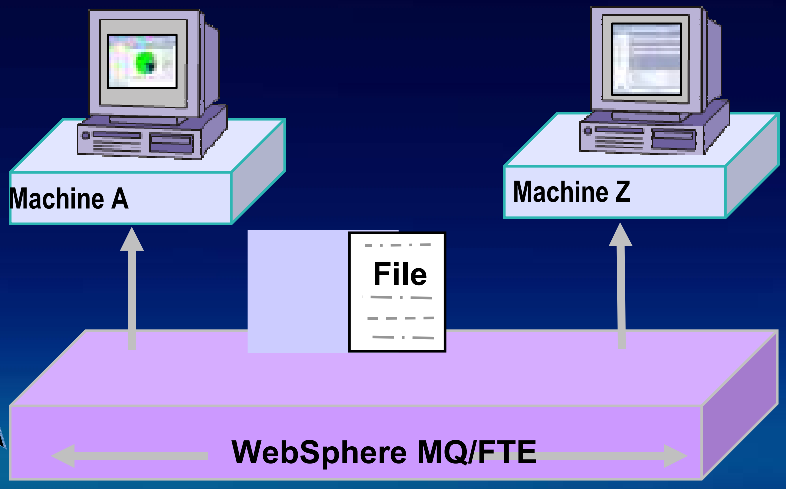


Reliable file transfers with WebSphere MQ/FTE

- Files can be transferred in a <u>reliable</u>, <u>secure</u> and <u>traceable</u> manner across the WebSphere MQ messaging layer using WebSphere MQ/FTE *(Confidential, not yet announced)*

Enables applications to exchanges files:

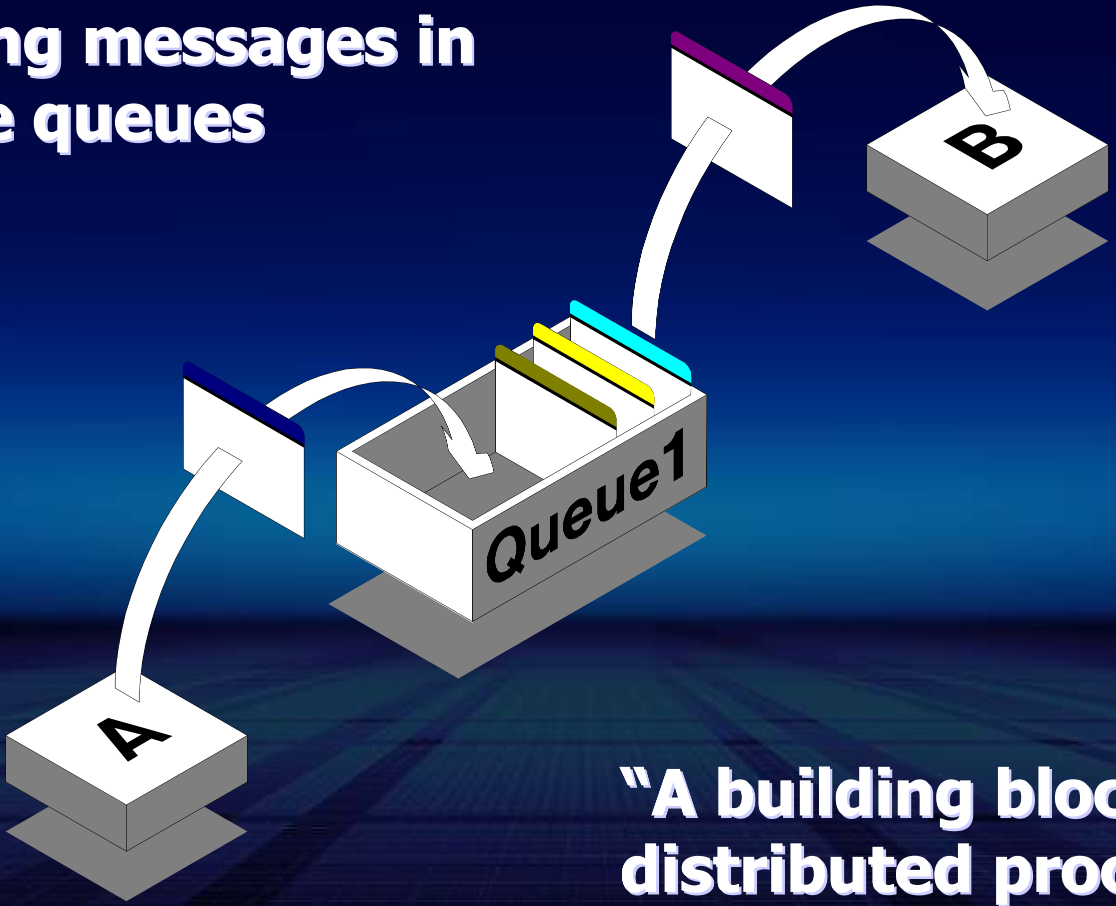
- ***Of any size (KB, MB, GB…)***
- ***Without programming interfaces***
- ***Using powerful graphical tooling in the Eclipse Workbench***
- ***With <u>reliability</u>, leveraging MQ***
- ***With <u>full auditability</u>***
- ***With <u>high-performance</u>***
- ***With code page <u>conversion</u>***
  - ***<u>ASCII ←→ EBCDIC</u>***
- ***With file data <u>compression</u>***
- ***With strong <u>security</u>***
- ***Across <u>many</u> supported MQ environments***



- **Helps reuse of existing file based data**
- **Removes lack of management controls from ad hoc solutions**
- **Improves availability of data**

Elements of Messaging and Queuing

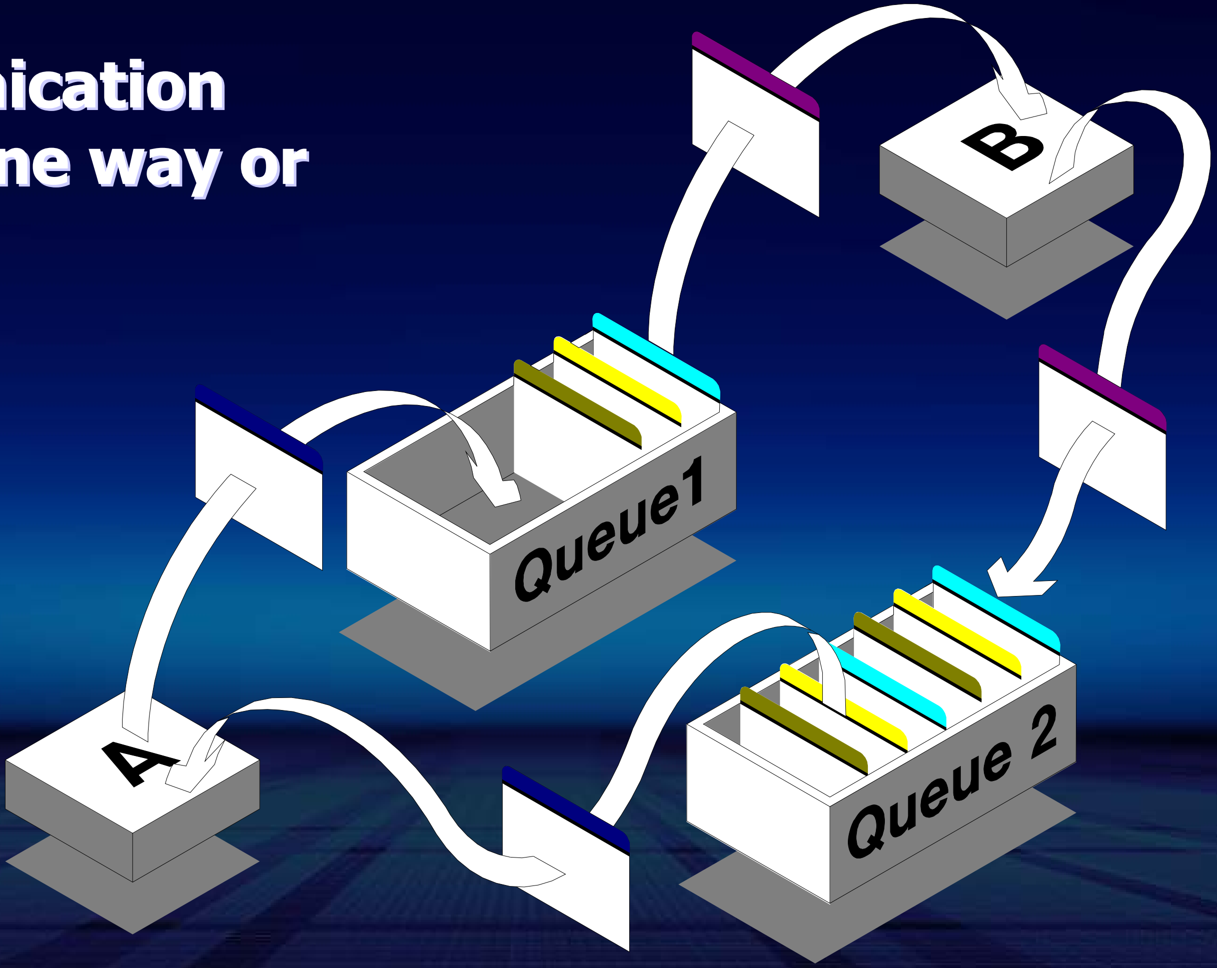
- **Programs communicate by putting messages in message queues**



"A building block for distributed processing"

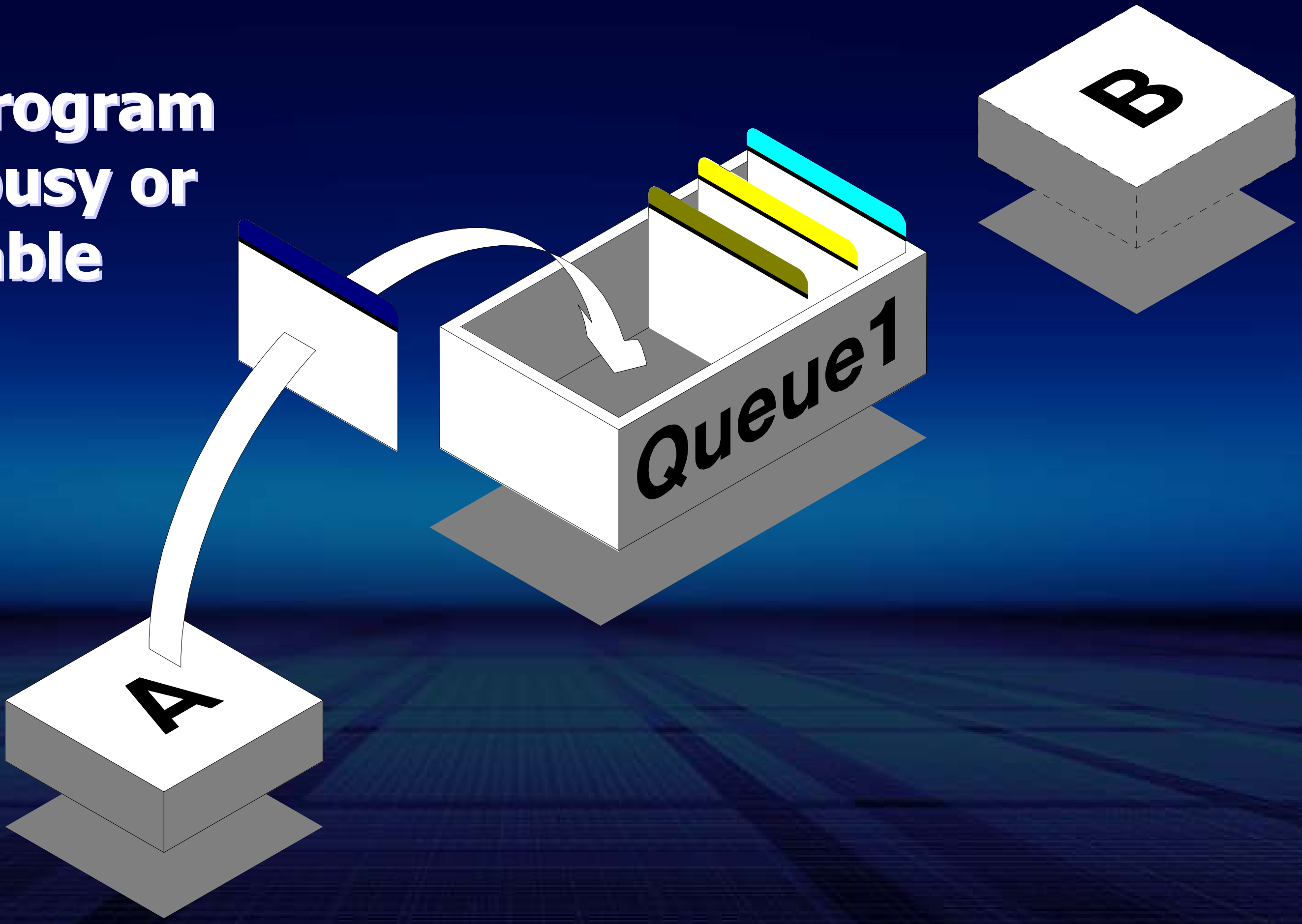
Elements of Messaging and Queuing

➔ **Communication can be one way or two way**



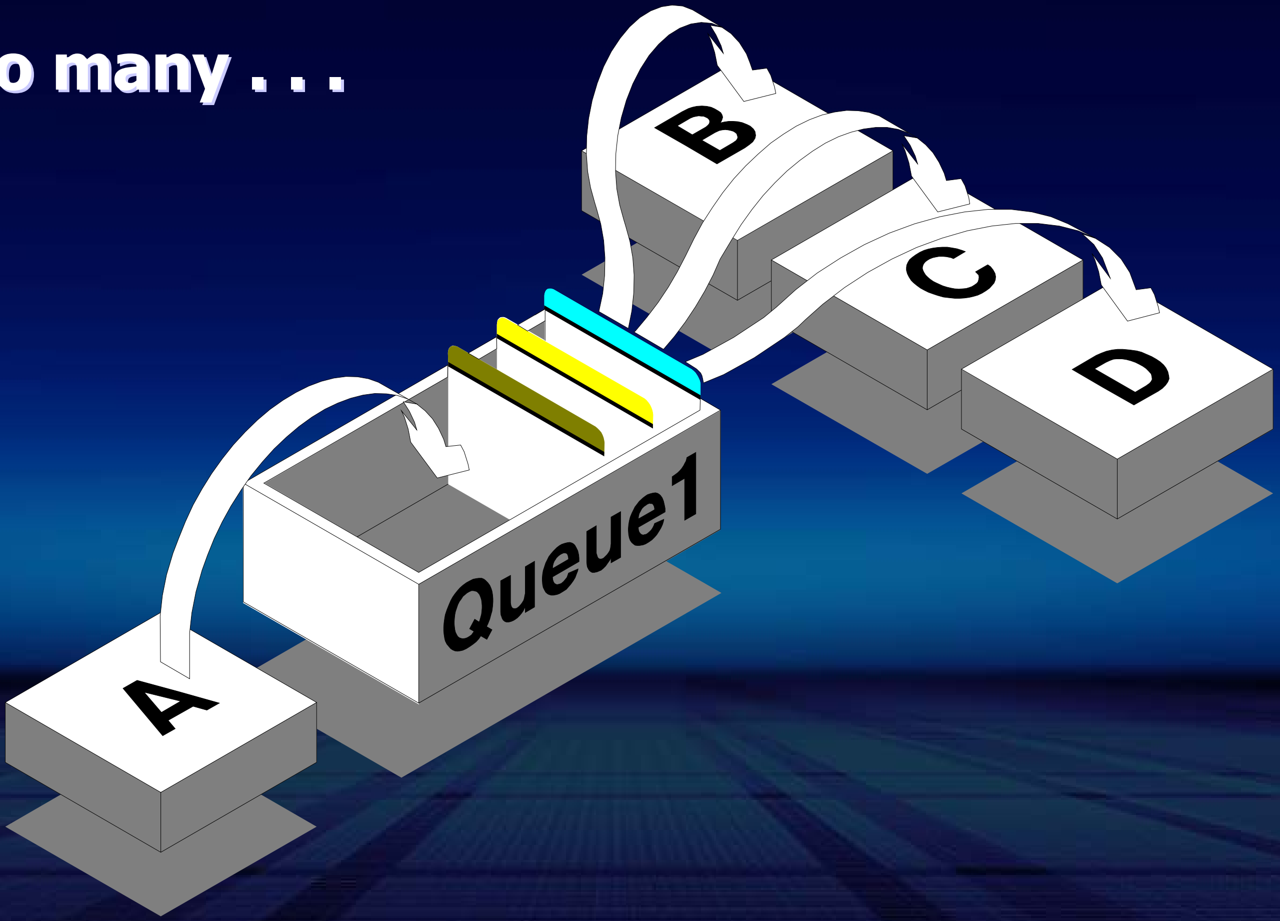
Elements of Messaging and Queuing

- **Either program can be busy or unavailable**



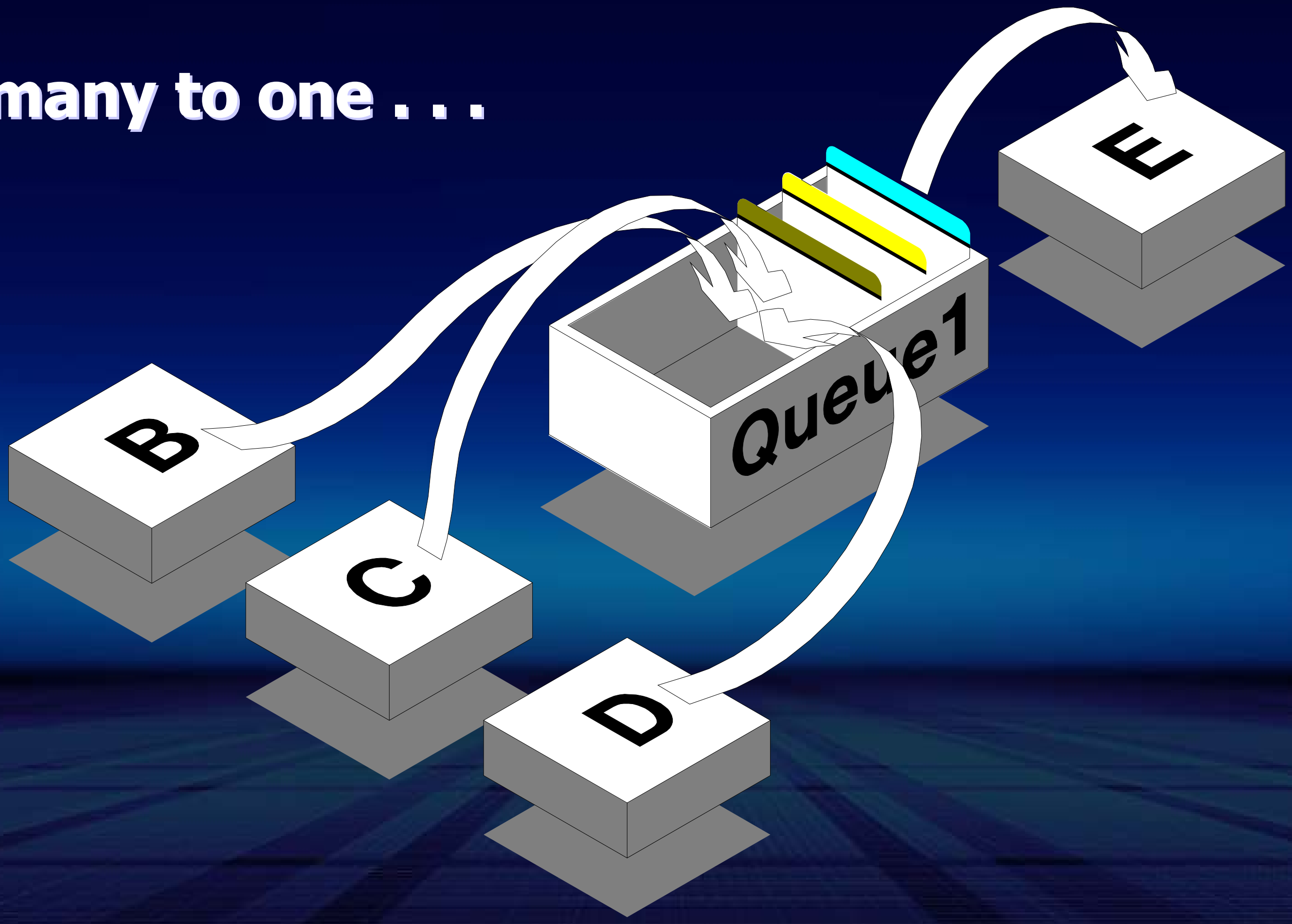
Elements of Messaging and Queuing

➔ **One to many . . .**



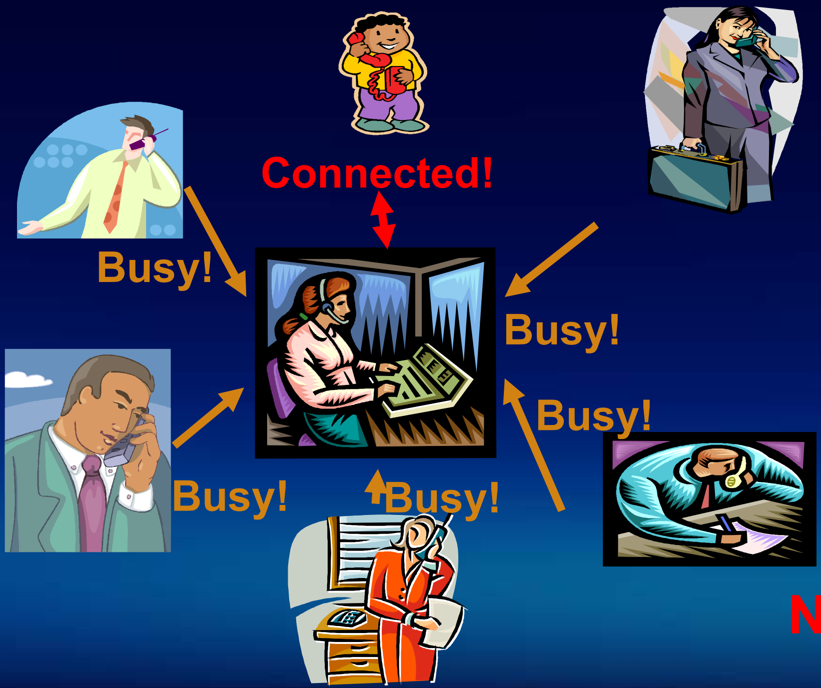
Elements of Messaging and Queuing

➔ **Or many to one . . .**



Synchronous vs. Asynchronous Communications

Synchronous communications = Telephone conversation



YOU can't simultaneously have <u>separate</u> conversations without:

- Experiencing overlapping conversations and losing track of what is going on

or

- Waiting for the other person to finish before responding.

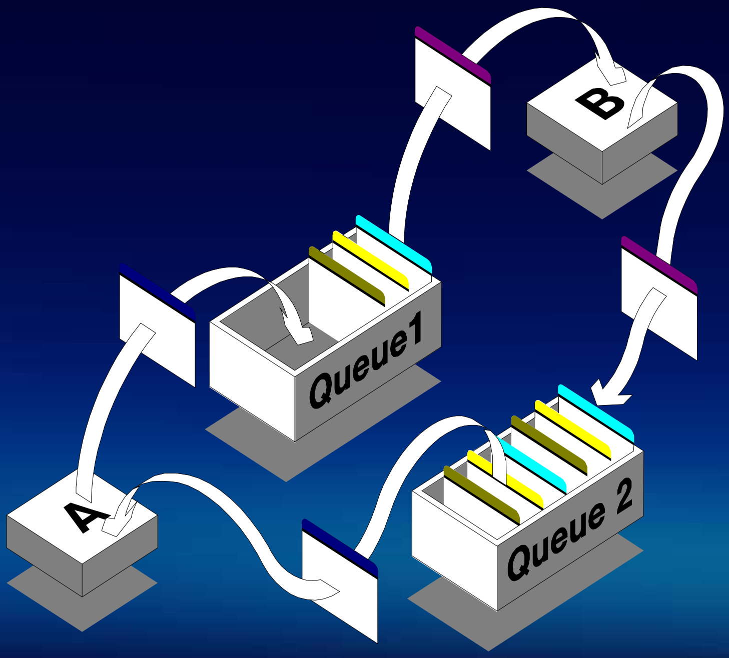
NEITHER CAN YOUR APPLICATIONS!

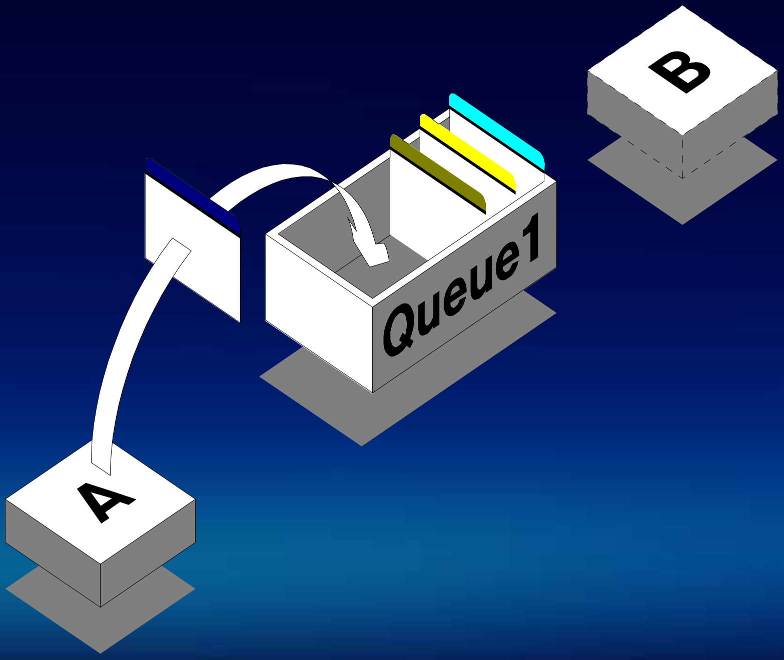
Asynchronous communications = Voicemail / Email

Please leave a message and I will listen to it when I am available.

Asynchronous communications enable better resource utilization thus improving performance; carry on processing until system is available.

Availability Choices





→ **Synchronous**

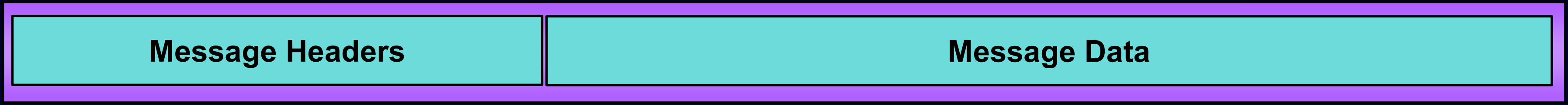
→ **Asynchronous**

The Message

- A message is considered to be the unit of data to be moved from one application to another
- A message is built by an application
- A message is consumed by a different application
- Message can contain **any** kind of data:
  - Binary data
    - A video clip, a song, a photograph, a sensor reading, etc…
  - Text data
    - Raw text
    - XML
  - Structured data (C Structures, COBOL Copybook, Serialized Java objects)
  - The source data is the choice of the application

The structure of an MQ Message

Maximum size of 100 megabytes based on Queue Manager configuration



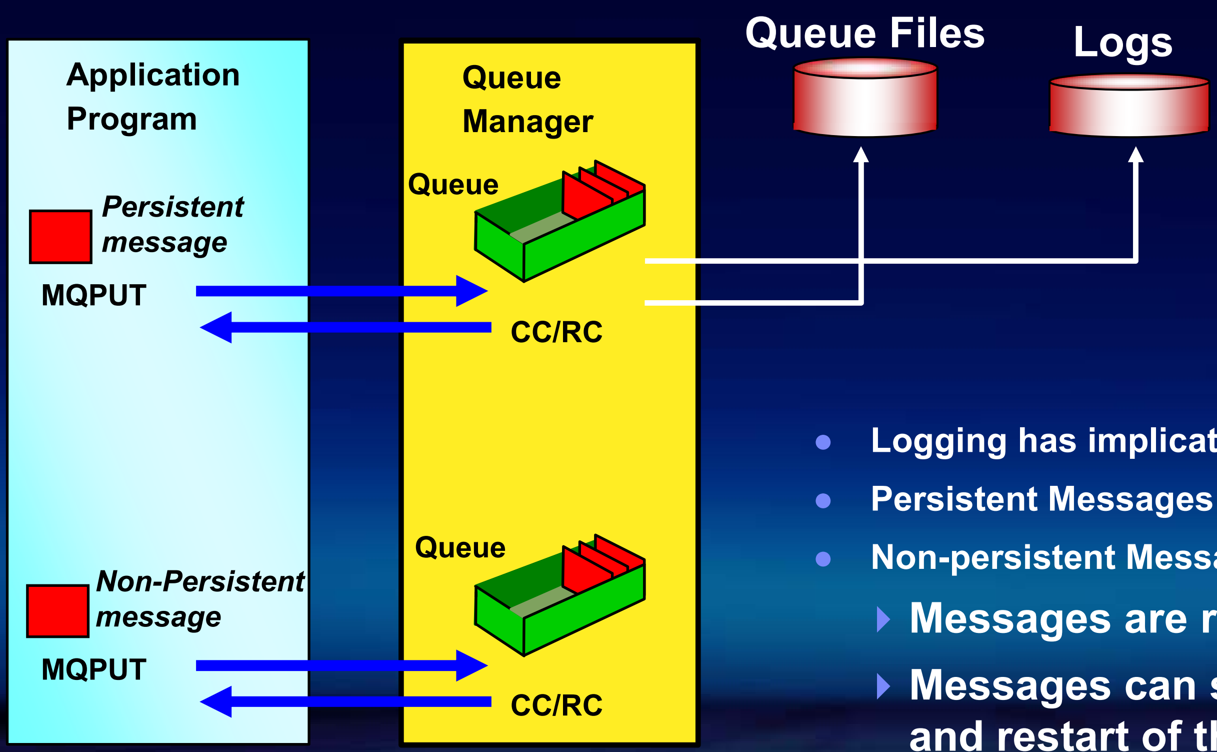
Message Headers

- A Series of Message Attributes
- Understood and augmented by the Queue Manager
  - Unique Message Id
  - Correlation Id
  - Routing Information
  - Reply Routing Information
  - Message Priority
  - Message Persistence
    - Persistent
    - Non-persistent
    - Semi-persistent
  - Message Codepage
  - Message Format
  - Etc…

Message Data

- *Any* sequence of bytes
  - Defined by the sending program
  - Understood by the receiving program
  - NOT meaningful to the Queue Manager
- Can contain any data
  - Structured
    - XML, Tagged, Tagged Delimited, C or Cobol defined, etc.
  - Unstructured
    - Binary
      - A video, a picture, etc.
    - Any content

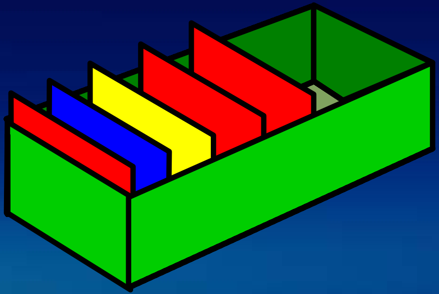
Message persistence



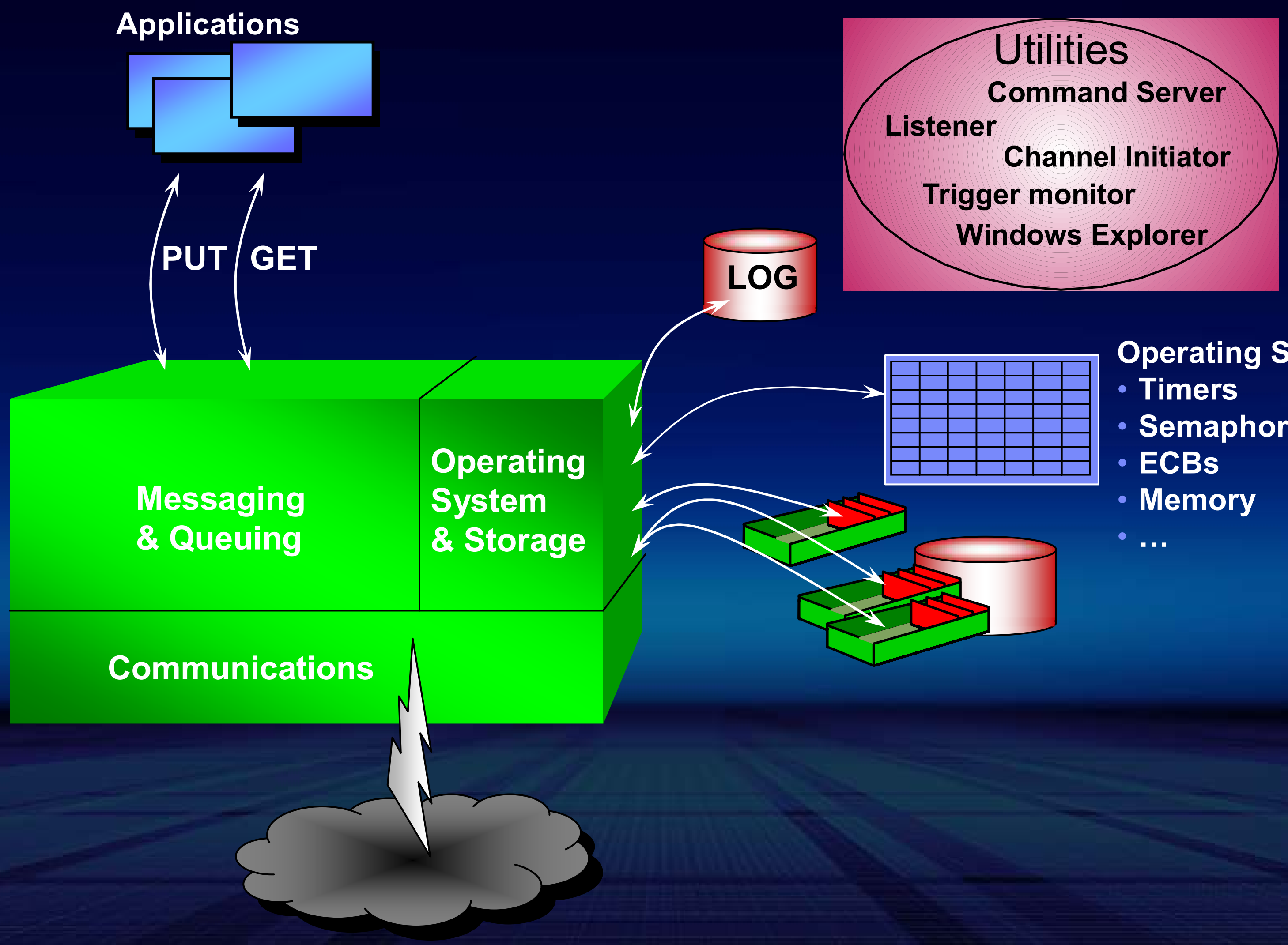
- Logging has implication on performance
- Persistent Messages are always recoverable
- Non-persistent Messages have 2 classes of service:
  - Messages are retained for the life of the Qmgr
  - Messages can survive a normal shutdown and restart of the Queue Manager

What is a Queue ?

- **Messages** are delivered asynchronously to a **Queue**
- **A Place to hold messages**
- **Queue creation**
  - Pre-defined
  - Dynamic definition
- **Message Access**
  - FIFO (first in first out)
  - Priority (FIFO within Priority)
  - Direct
  - Destructive & non-destructive access
- **Parallel access by applications**
  - Managed by the queue manager



What is a Queue Manager?



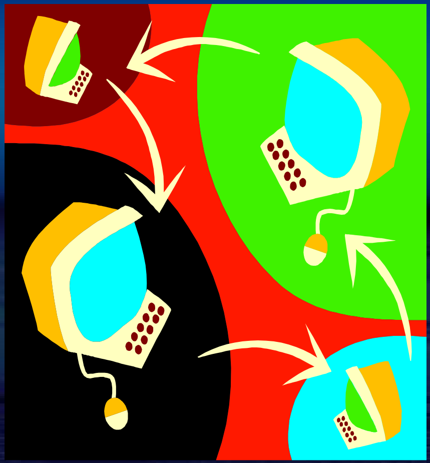
Operating System
- Timers
- Semaphores
- ECBs
- Memory
- …

Channels

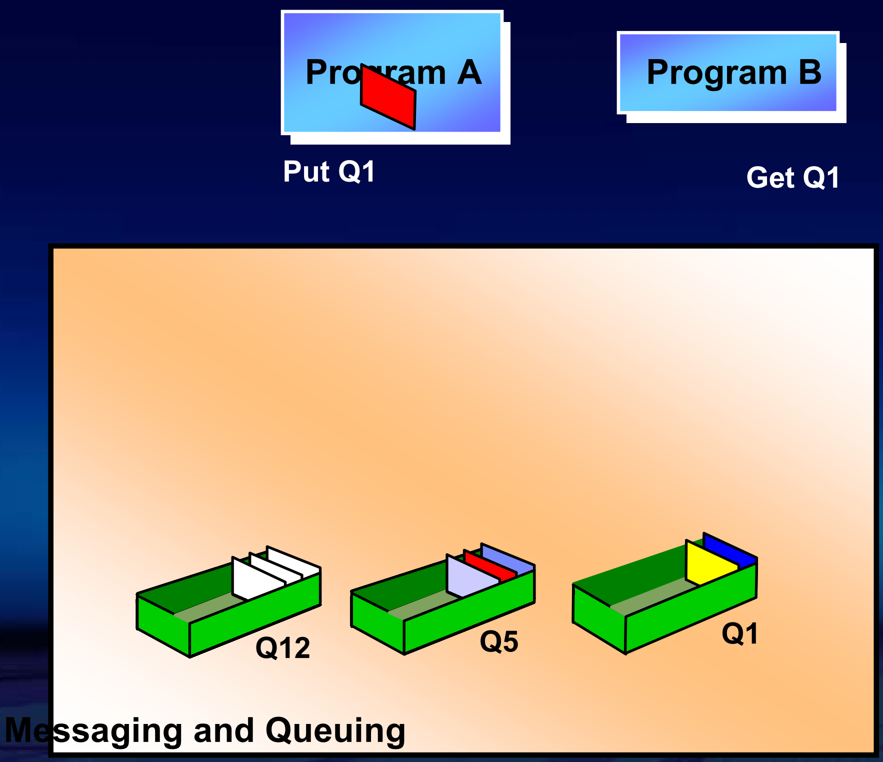
- Queue Manager to Queue Manager
  - Uni-directional
  - Usually defined in pairs for example:
    - One Sender
    - One Receiver
  - Asynchronous
- Client to Queue Manager
  - Bi-directional
  - Defined as a single channel
  - Synchronous

** Note Client to Client communication must go via a Queue Manager

- A building block for the ESB



Reliable, asynchronous communication with WebSphere MQ



Accept Message

- Receive message from application
- Manage “unit of work”

Apply Security (optional)

- Access Control

  (permission to get/put by queue)

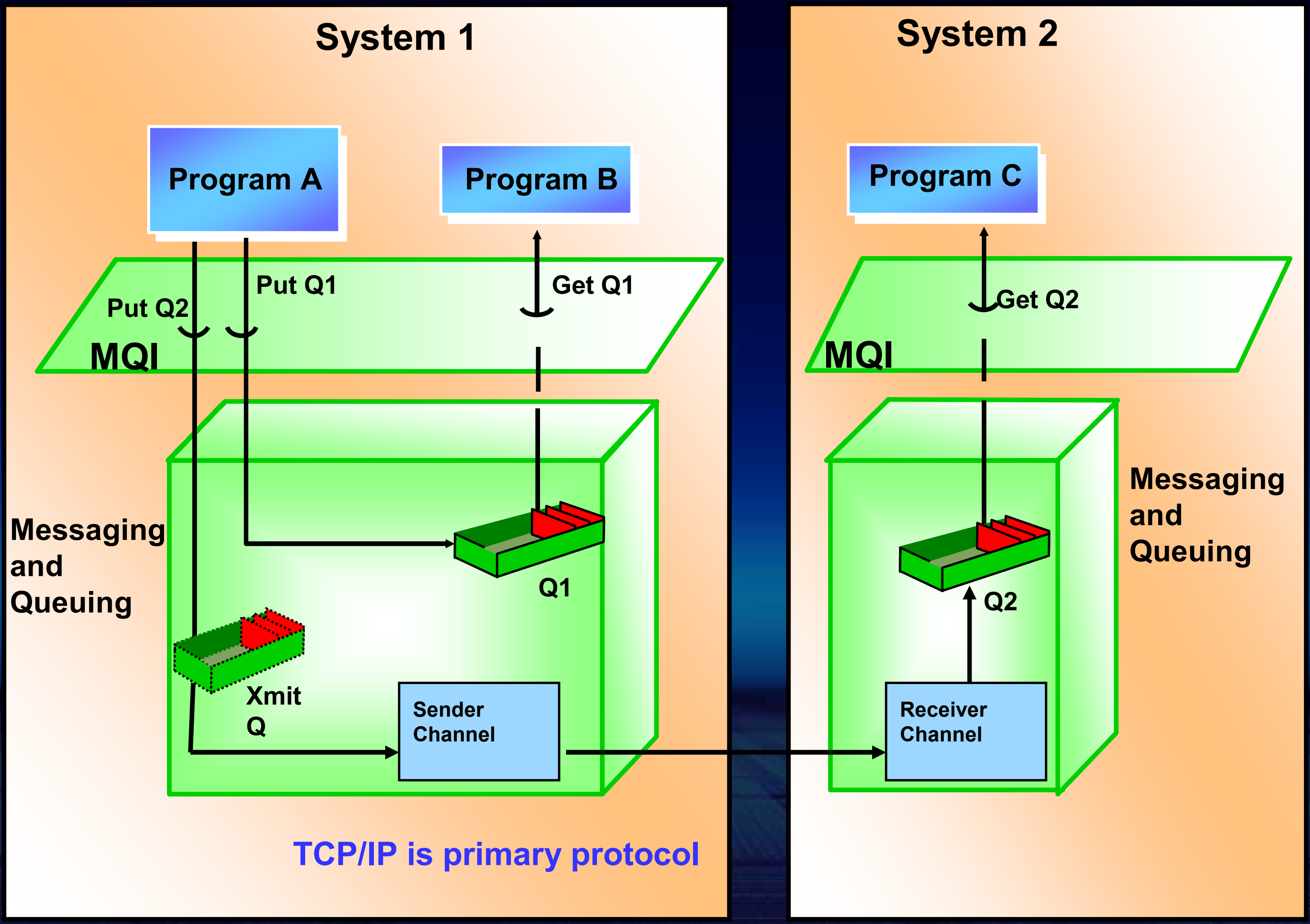
Deliver Message(s)

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

Applications can be transactional

- Messaging can be performed under transaction control
  - Messages can be put or got under a logical unit of work
  - Messages can be committed or rolled back as an atomic unit
- Distributed transactions are supported
  - WebSphere MQ can be XA resource manager
  - WebSphere MQ can be XA transaction manager
- When WebSphere MQ is used as an XA resource Manager
  - A queue and a database operation can be performed under a single logical unit-of-work using commit / rollback logic
  - For example
    - Get a message from a queue and insert into a database with a single commit
  - Most commercial database systems are XA compliant and can be under control of WebSphere MQ as a resource Manager

Applications can use Point to Point



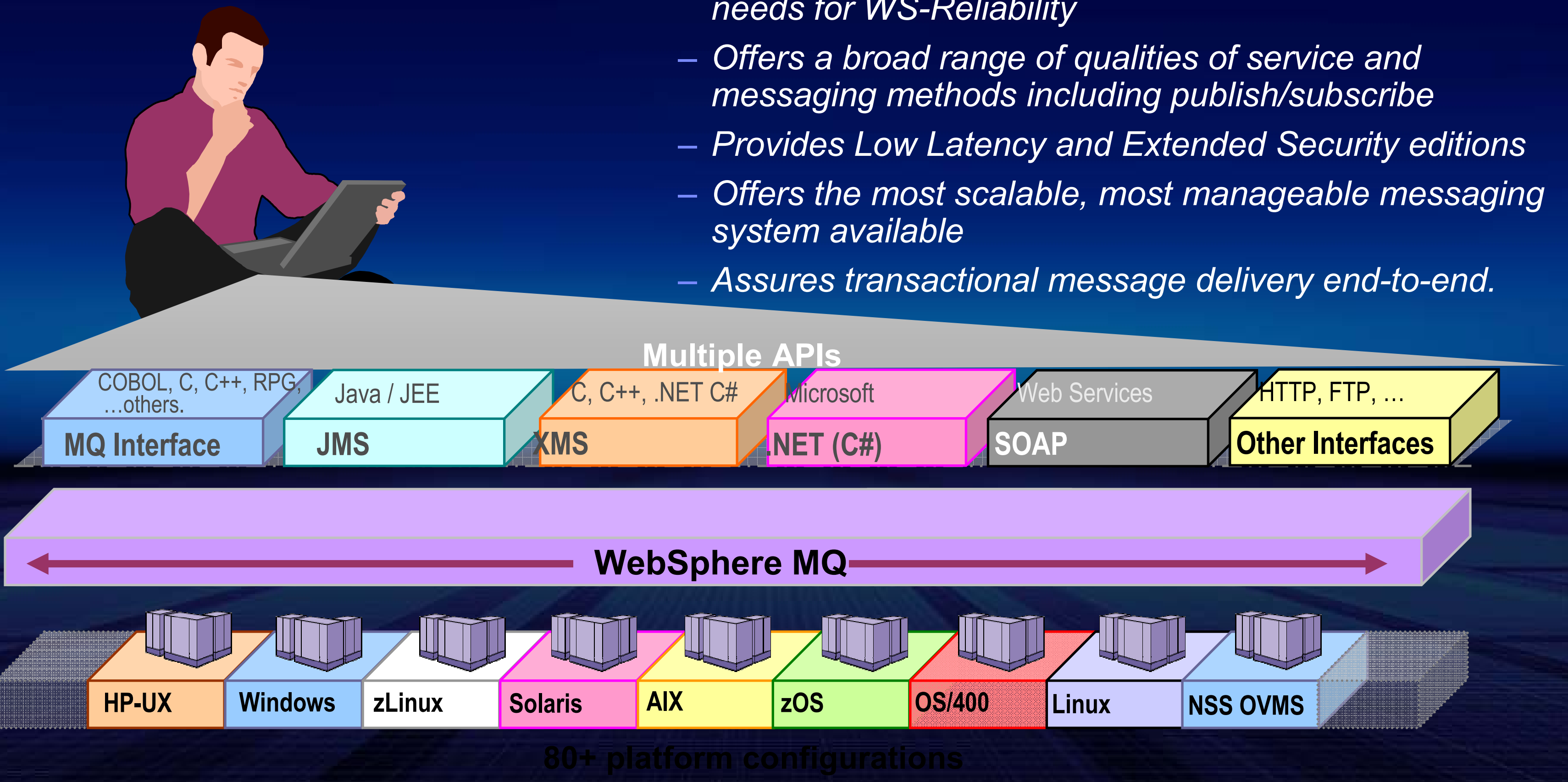
TCP/IP is primary protocol

The solution to Universal Connectivity → IBM WebSphere MQ

WebSphere MQ can dramatically reduce application infrastructure costs by providing a single manageable distributed infrastructure for all application messaging traffic.

Features:

- *Supports the broadest range of APIs, programming languages and OS platforms*
- *Provides the only JMS engine that can be implemented on "any" standards-compliant JEE server*
- *Provides rich web services interfaces meeting customer needs for WS-Reliability*
- *Offers a broad range of qualities of service and messaging methods including publish/subscribe*
- *Provides Low Latency and Extended Security editions*
- *Offers the most scalable, most manageable messaging system available*
- *Assures transactional message delivery end-to-end.*



Smart SOA

WebSphere MQ Enterprise Class Messaging

- Proven Scalability
  - Grow your network **incrementally** one server at a time
- Performance
  - Many clients are moving **millions of messages** per day
- Administer massive networks
  - Cross-platform, remote configuration tooling
  - **Tivoli CAM** for enterprise-wide systems administration
- Support for virtually any commercial IT platform
- MQ for zOS
  - Built from the ground up to exploit zSeries platform
  - Consistent with MQ on distributed platforms
- Clustering on distributed, shared queues on zOS
  - For **High-Availability** and workload balancing
  - Easier to set up than you may think!
- Multi-threading
  - Exploits **multi-processors** for high-speed throughput
- Security
  - Industry-standard **SSL support**
  - Certified for **Common Criteria**
  - Policy-based security with **MQ Extended Security Edition**
- IBM's worldwide 24x7 support

- **90% of the Fortune 100**
- **300 of the Fortune 500**
- **66% of NA and European banks**
- **Banking clients move transactions worth $35 Trillion over MQ**
- **Government clients move 675+ million messages per day over MQ**