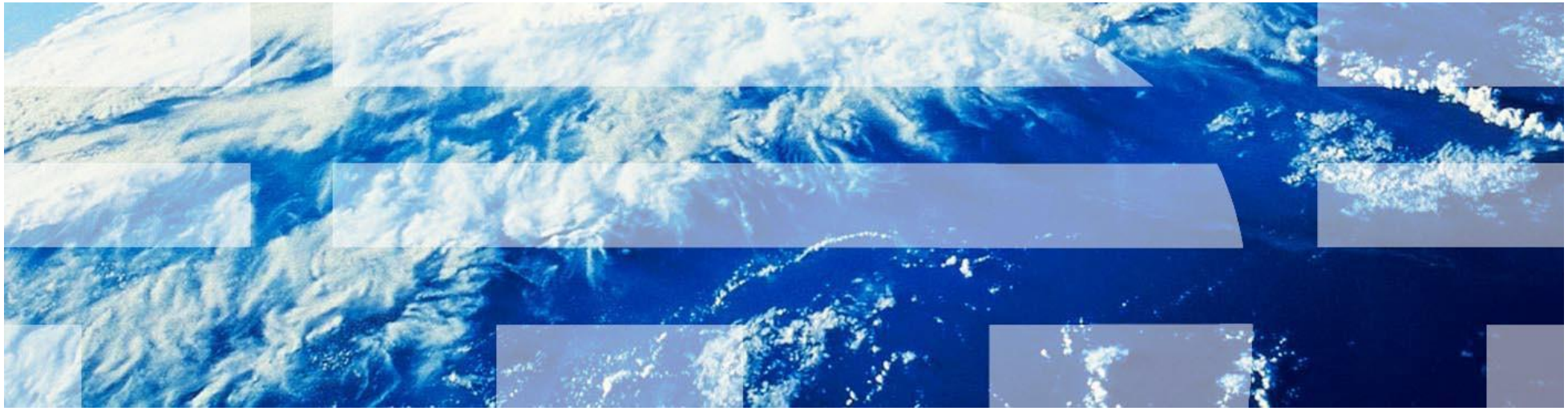
IBM MQ

Introduction and Agenda



Course Agenda

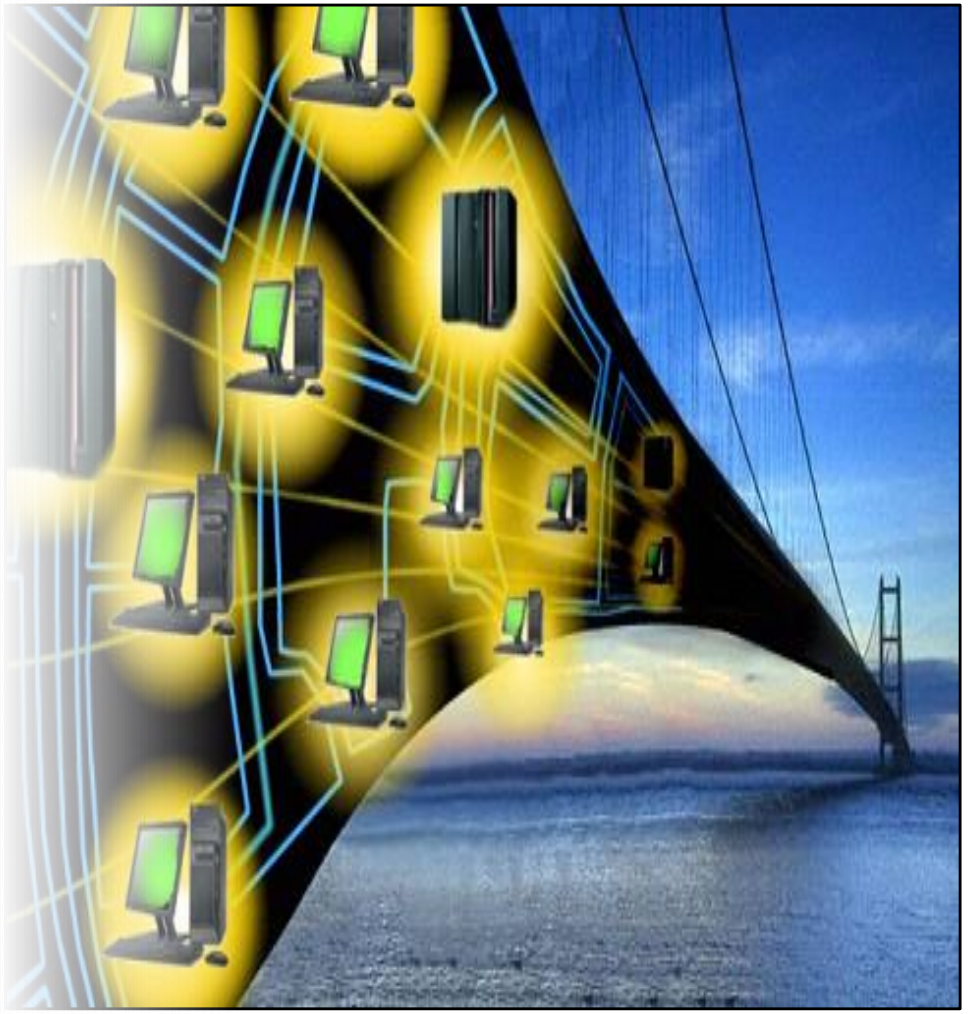
- 10:30 ~ 11:00 : IBM MQ Overview (실습소개) + IBM 통합 모니터링툴 소개
- 11:00 ~ 11:50 : Lab 1 (Lecture – This deck)
- 11:50 ~ 13:00 : Lunch
- 13:00 ~ 15:00 : Lab 1 (Hands on)
- 15:00 ~ 16:00 : Lab 2 (Lecture & Hands on)
- 16:00 ~ 17:30 : Lab 3 (Lecture & Hands on)
- 17:30 ~ 18:00 : Wrap up

Unit Agenda

- What is IBM® MQ? Why use it?
- What are:
  - Messages
  - Queues
  - Queue Managers
  - Channels
- What forms of IBM MQ are available?
- Where can IBM MQ be run?

What is IBM MQ?

- 어떤 유형의 데이터든 메시지로 전송하는 검증된 방법
- 비즈니스가 마이크로서비스와 SOA와 같은 유연하고 재사용 가능한 아키텍처를 구축할 수 있도록 지원
- 가상으로 모든 상업용 IT 시스템을 연결.
- 다양한 컴퓨팅 플랫폼, 애플리케이션 환경, API 및 통신 프로토콜과 함께 작동하여 보안이 강화된 메시지 전달을 제공

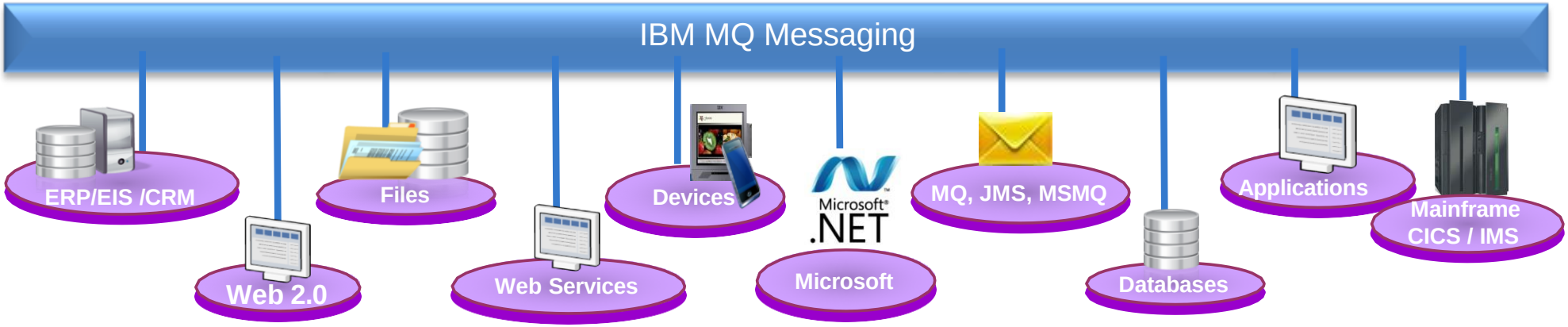


Notes - What is IBM MQ?

Key Takeaways:

- MQ는 신뢰할 수 있는 방식으로 사물을 연결
- 비즈니스와 IT 부서의 숙련된 인력이 핵심 비즈니스 가치 항목에 집중

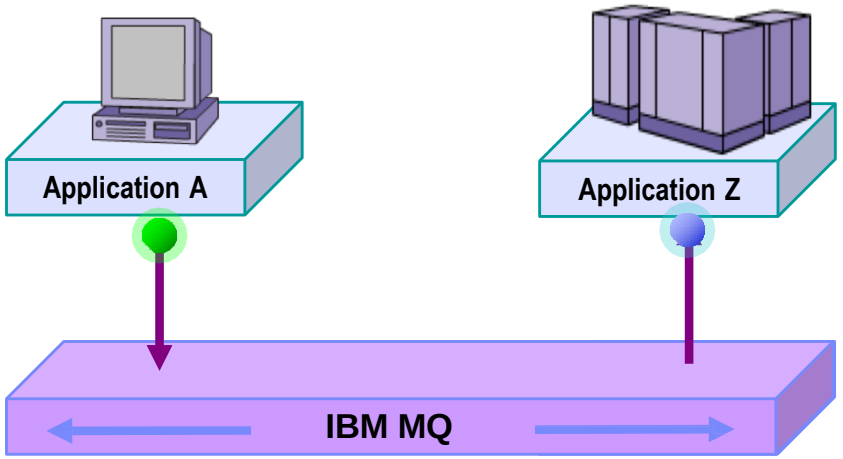
The value of IBM's IBM MQ Messaging capabilities



- 어디에 있든 모든 유형의 데이터에 접근할 수 있는 범용 메시징 패브릭을 제공
  - 비동기 신뢰 메시징
  - 동기 경고, 이벤트, 알림
- 신뢰할 수 있고 안전한 데이터 전달을 통해 다양한 엔드포인트 간의 연결을 지원
  - 애플리케이션 , 서비스 , 모바일 , 센서 데이터
- Topic Space 를 기반으로 관련 정보를 동적으로 실시간으로 전달
  - Publish / Subscribe, Multicast
- 각각의 특정 사용 사례에 최적화된 다양한 전송 방식을 지원
  - MQ는 엔터프라이즈 메시징을 위해 사용되고, MQTT는 모바일, 센서, 사물인터넷(IoT)을 위해 사용되며, MFT는 MQ를 통한 관리 파일 전송을 위해 사용;
  - HTTP/웹소켓은 인터넷 메시징을 위해 사용되며, Advanced Message Security(AMS)는 종단 간 페이로드 보안을 제공

What does IBM MQ do?

- 애플리케이션과 서비스가 데이터와 이벤트를 교환할 수 있도록 메시징을 제공:
  - Proven reliability
  - Transactional integrity
  - Consistency
  - Time independence
  - Ease and Speed
  - Flexibility
  - High-performance
  - Security
  - Scalability
  - Auditability



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

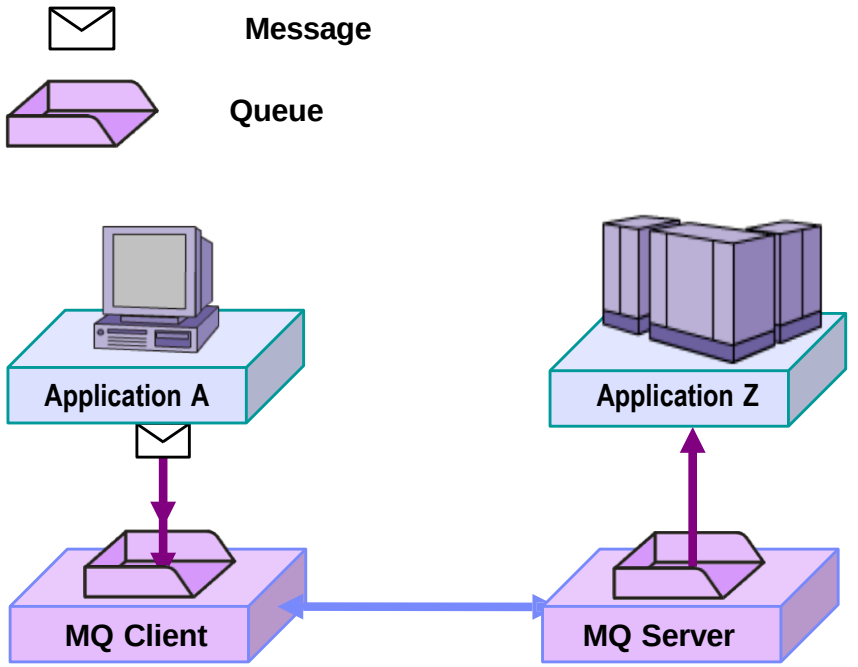
Notes - What does IBM MQ do?

Key Takeaways:

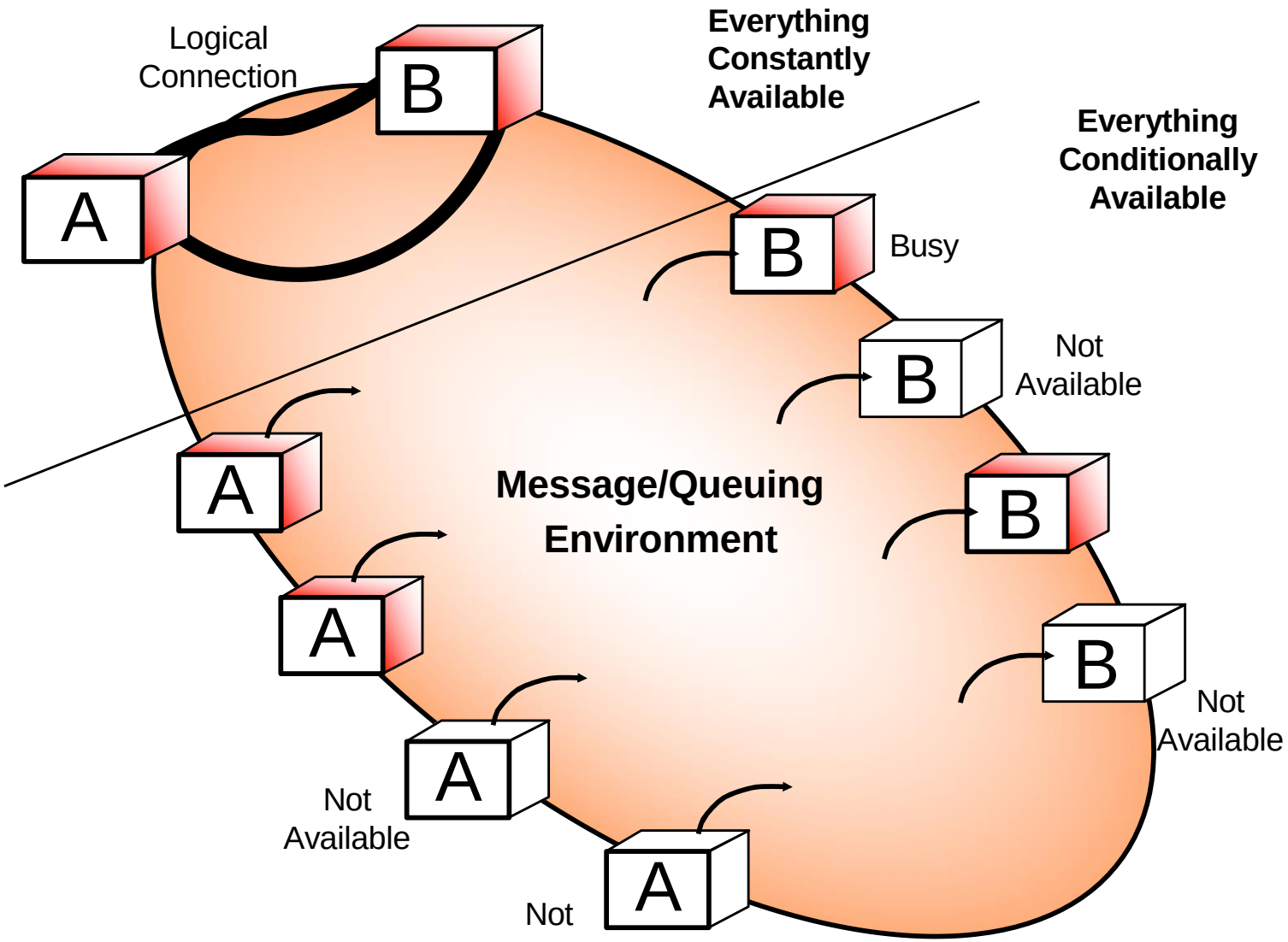
- MQ enables you to connect applications and services together with valuable qualities-of-service.
- Applications can exchange information without tying themselves up – just like email where people communication a-synchronously.

How does IBM MQ work?

- 메시징 서비스는 간단한 프로그래밍 명령을 기반으로 데이터를 **저장하고 전달**하는 큐를 기반
- 메시지가 **손실되거나 중복되지 않도록** 보장하기 위해 검증된 데이터베이스 기술인 2단계 커밋 트랜잭션을 사용
- 키워드 또는 "토픽"을 기반으로 메시지를 **동적으로 라우팅**하기 위해 발행/구독 방식을 사용
- 멀티 프로세서 스레딩과 클러스터링을 사용하여 **메시지 처리량을 가속화**

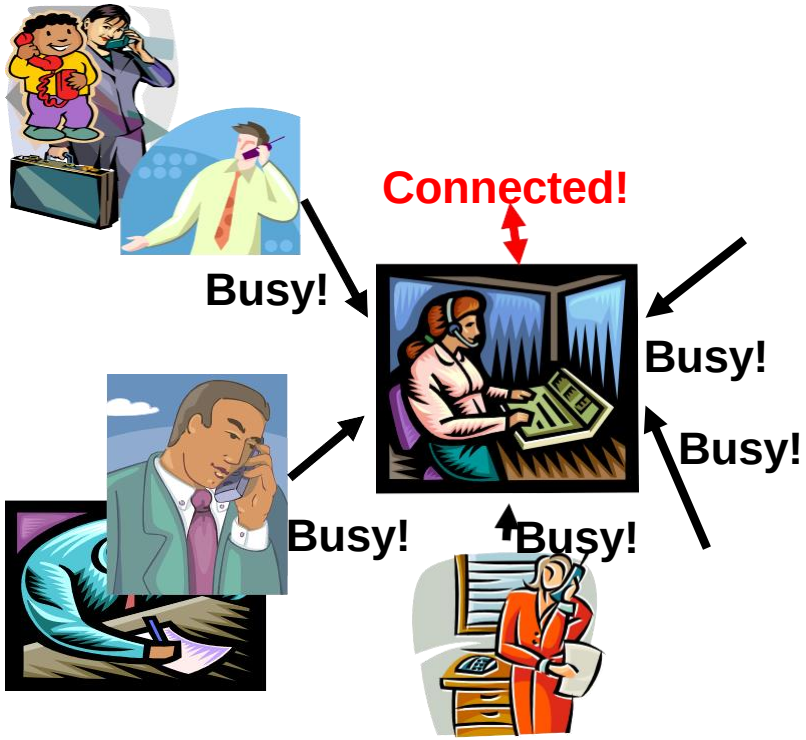


The Value of Loose Coupling



Synchronous vs. Asynchronous Communications

동기 통신 = 전화 통화



YOU can't simultaneously have separate conversations without:

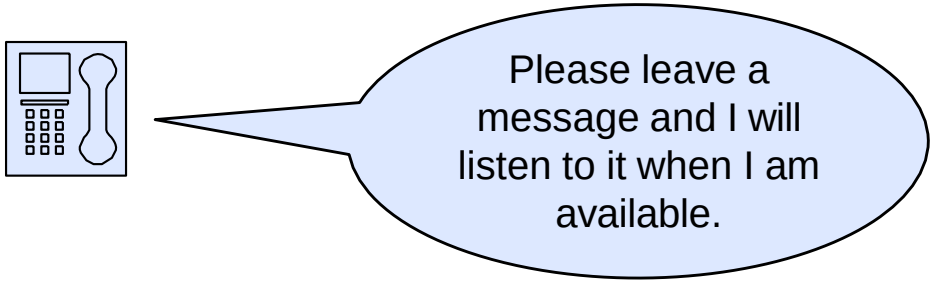
- Experiencing overlapping conversations and losing track of what is going on

or

- Waiting for the other person to finish before responding.

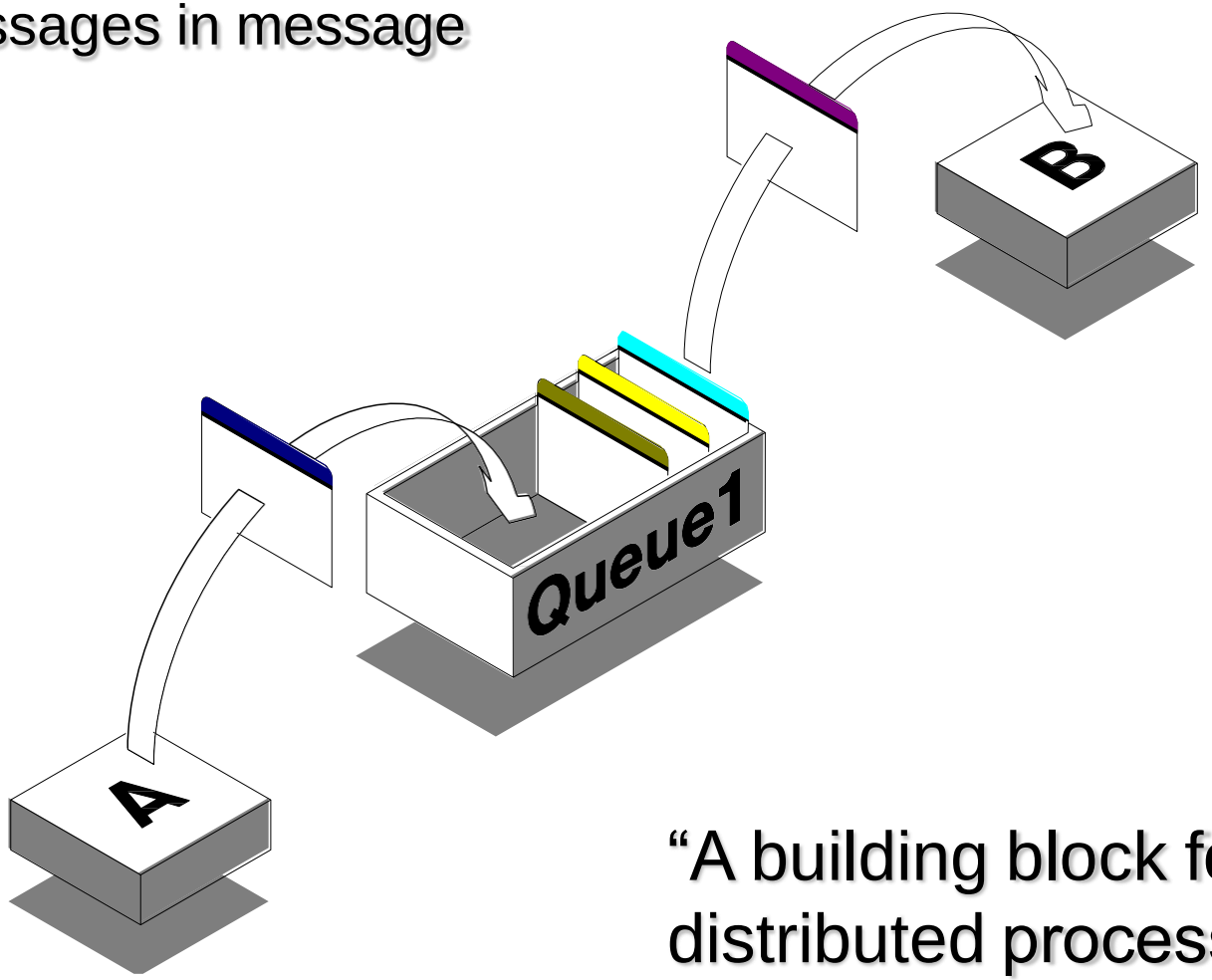
NEITHER CAN YOUR APPLICATIONS!

비동기 통신 = 음성메일 / 이메일



Elements of Messaging and Queuing

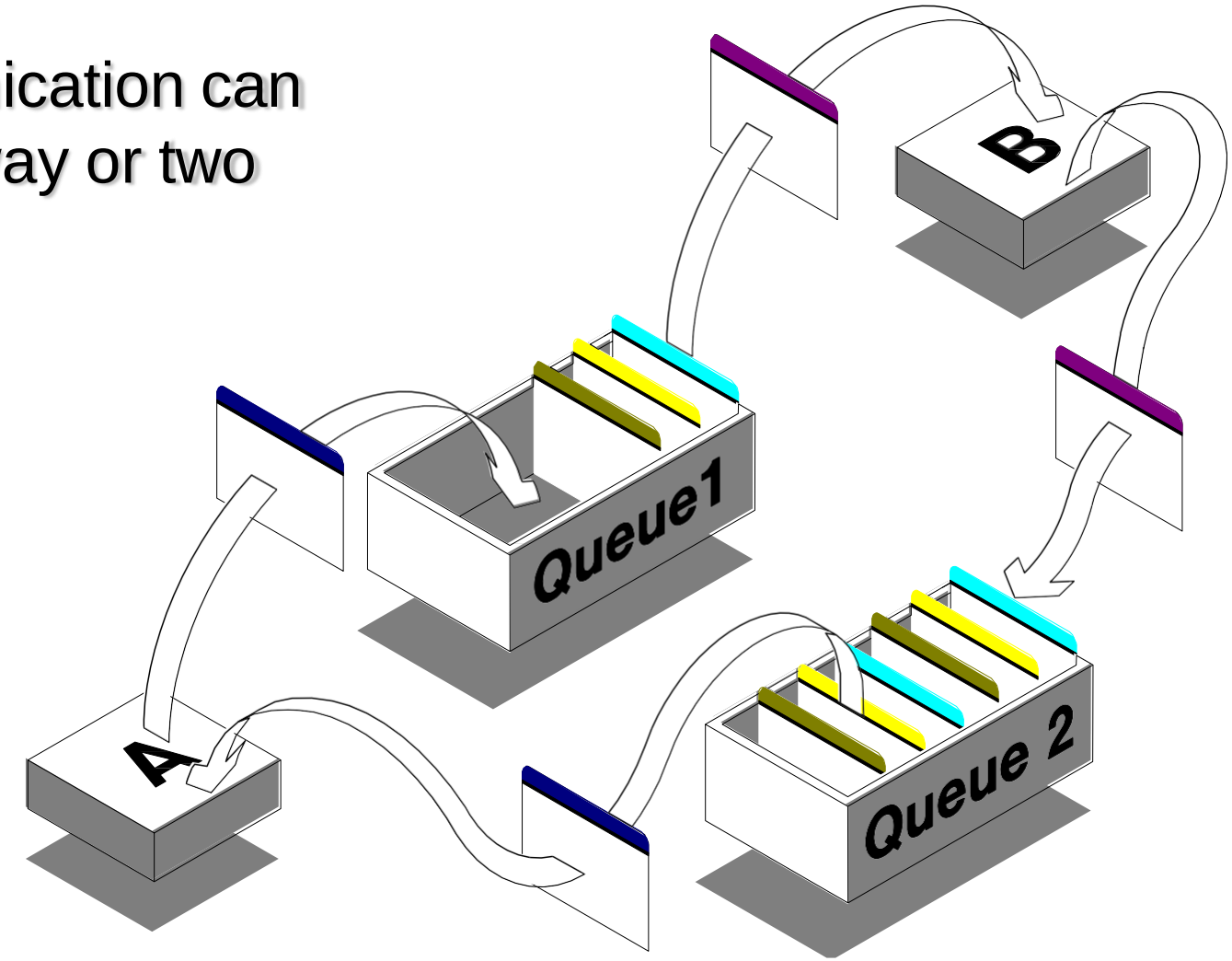
- Programs communicate by putting messages in message queues



"A building block for distributed processing"

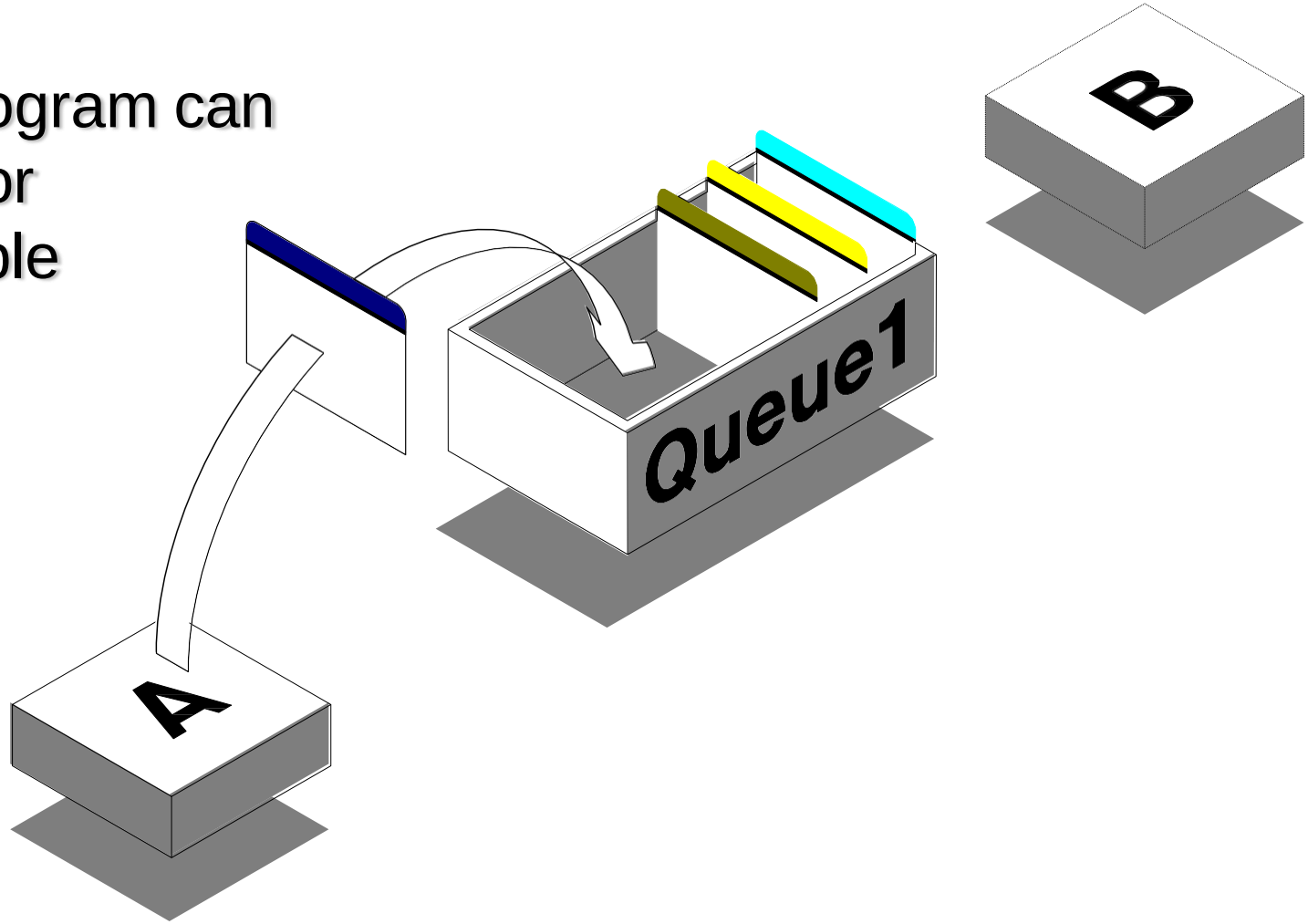
Elements of Messaging and Queuing

- Communication can be one way or two way



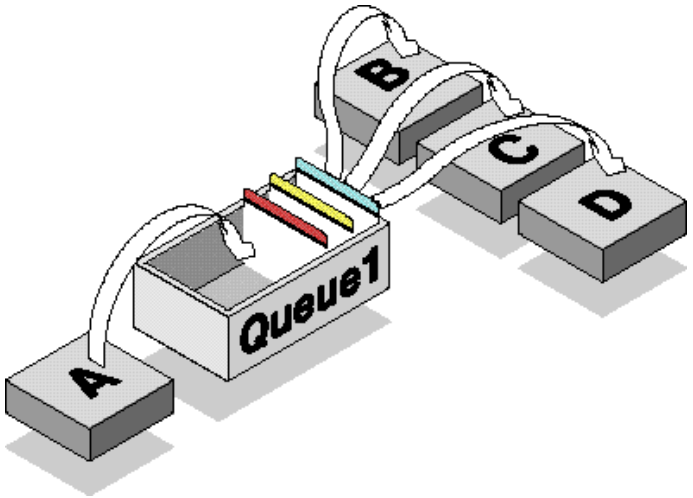
Elements of Messaging and Queuing

- Either program can be busy or unavailable

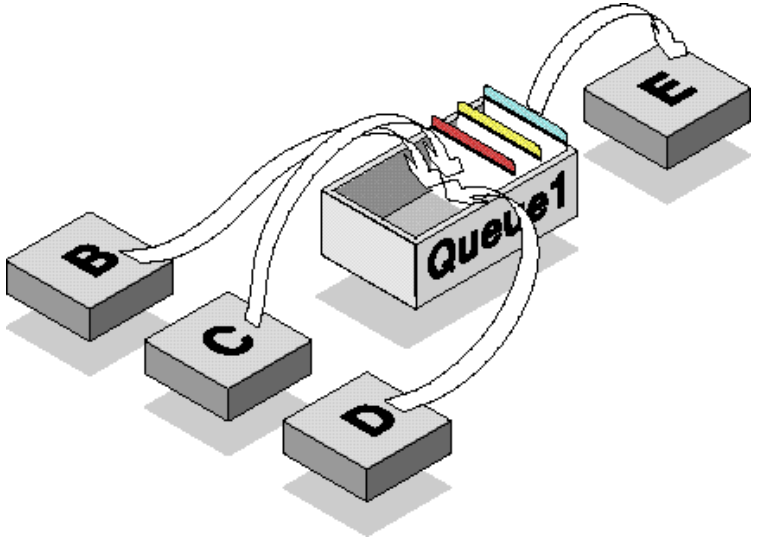


Elements of Messaging and Queuing

There can be a one to many relationship between applications



Or a many to one relationship between applications



What is a Message?

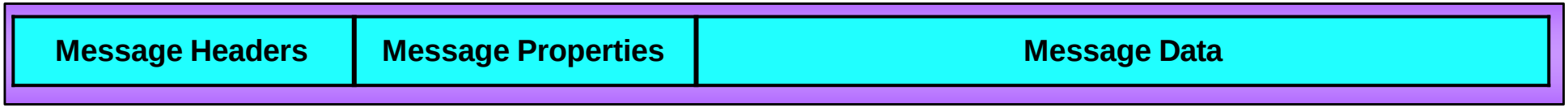
- 메시지는 한 애플리케이션에서 다른 애플리케이션으로 이동되는 데이터의 단위로 간주됨.
- 한 애플리케이션에서 생성 (Build) 됨
- 다른 애플리케이션에서 소비 (Consume) 됨
- 메시지의 데이터 유형 :
  - Binary data
    - A video clip, a song, a photograph, a sensor reading, etc…
  - Text data
    - Raw text
    - XML
  - Structured data (C Structures, COBOL Copybook, Serialized Java objects)
  - 데이터의 형식은 애플리케이션의 필요에 따라 선택 가능

Notes - What is a message?

Having identified two distinct paradigms for data transmission, let us examine what is actually passed between the participants. In our story, the unit of data propagated from one place to another is called “the message”. The message is the unit of data to be moved. It is constructed by one application and destined to be consumed by another. The message can be thought of as a container for data much in the same way as an envelope can be considered the container for the information within.

The data can be of a variety of different formats. The message does not care what kind of data it contains.

The structure of an MQ Message



Message Headers

- Queue Manager 가 이해하고, 확장하는 메시지 속성 집합
  - –Unique Message Id
  - –Correlation Id
  - –Routing Information
  - –Reply Routing Information
  - –Message Priority
  - –Message Persistence
    - Persistent
    - Non-persistent
  - –Message Codepage
  - –Message Format
  - –Etc…

Message Properties

- 메시지와 연관되어 있지만 본문에 포함되지 않은 임의의 값들
- 속성값 유형: integers, strings, boolean, etc.
- 수신애플리케이션은 필요에 따라 확인 가능
- 메시지 간의 관계를 명시적으로 표현할 수 있도록 허용
  - ▶ e.g. Message X is a **REPLY** to Message Y

Message Data

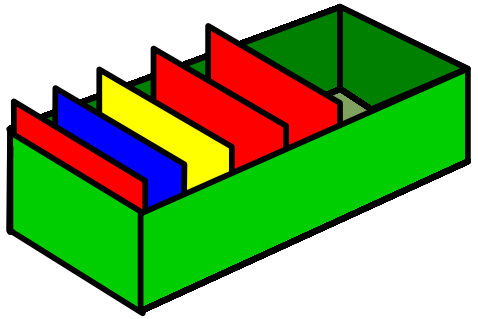
- *Any* sequence of bytes
  - –발신프로그램에서 정의
  - –수신프로그램에서 이해 (Understood)
  - –Queue Manager 에게는 의미가 없음
- 다양한 데이터를 포함
  - –Structured
    - XML, Tagged, Tagged Delimited, C or Cobol defined, etc.
  - –Unstructured
    - Binary
    - Any content

Notes - The structure of an MQ message

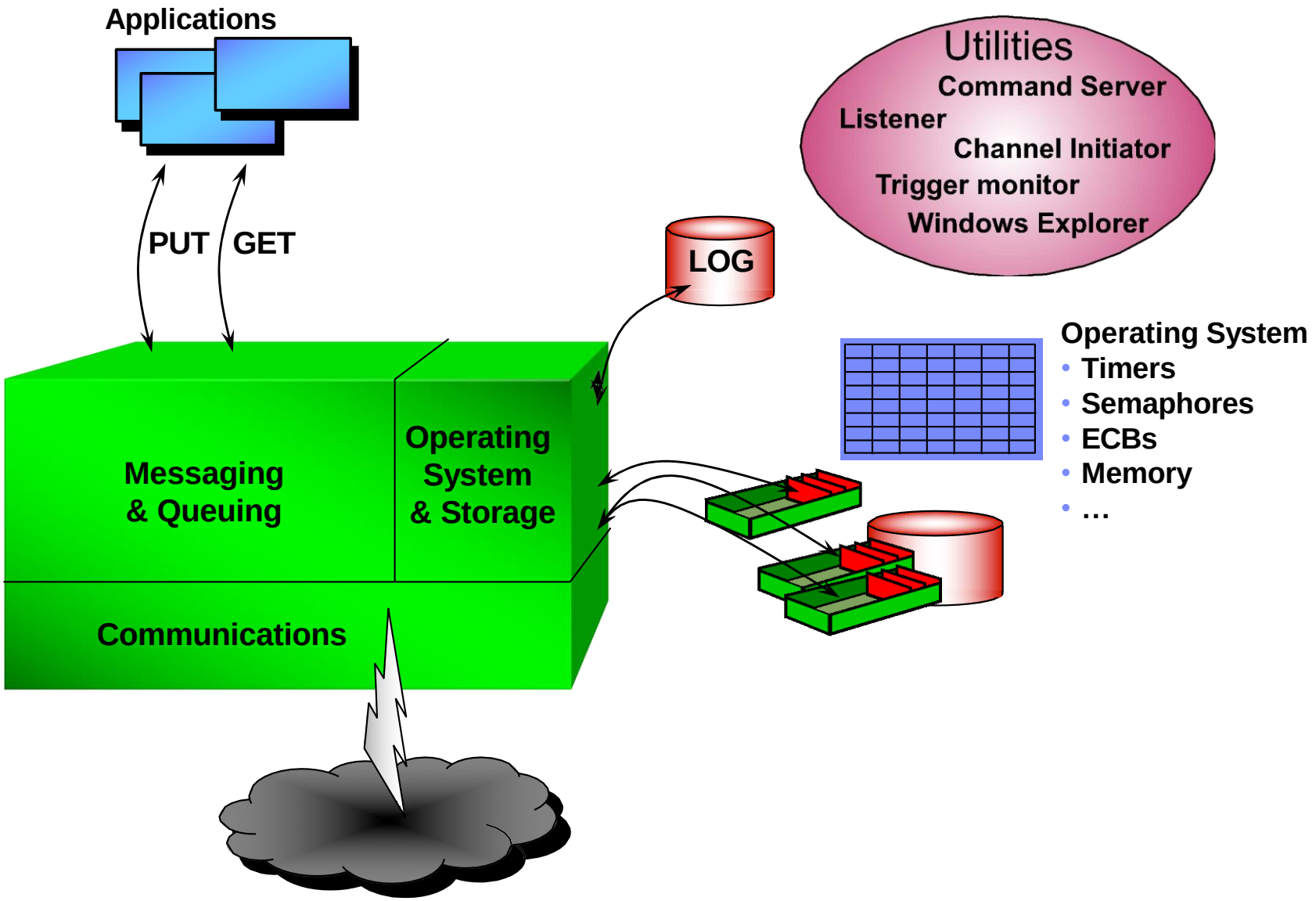
Note that the graphic does NOT depict to scale the relationship of Message headers and Properties to the message body (which can be up to 100MB)

What is a Queue?

- **메시지**는 비동기적으로 **큐**에 전달
- **메시지 보관 장소**
- **Queue 생성**
  - 사전 정의된 큐
  - 동적 정의
- **Message Access**
  - FIFO (first in first out)
  - Priority (FIFO within Priority)
  - Direct
  - Destructive & non-destructive access
- **애플리케이션에 의한 병렬 접근**
  - Queue Manager 가 관리



What is a Queue Manager?



Notes - What is a queue manager?

Explain queue manager components:

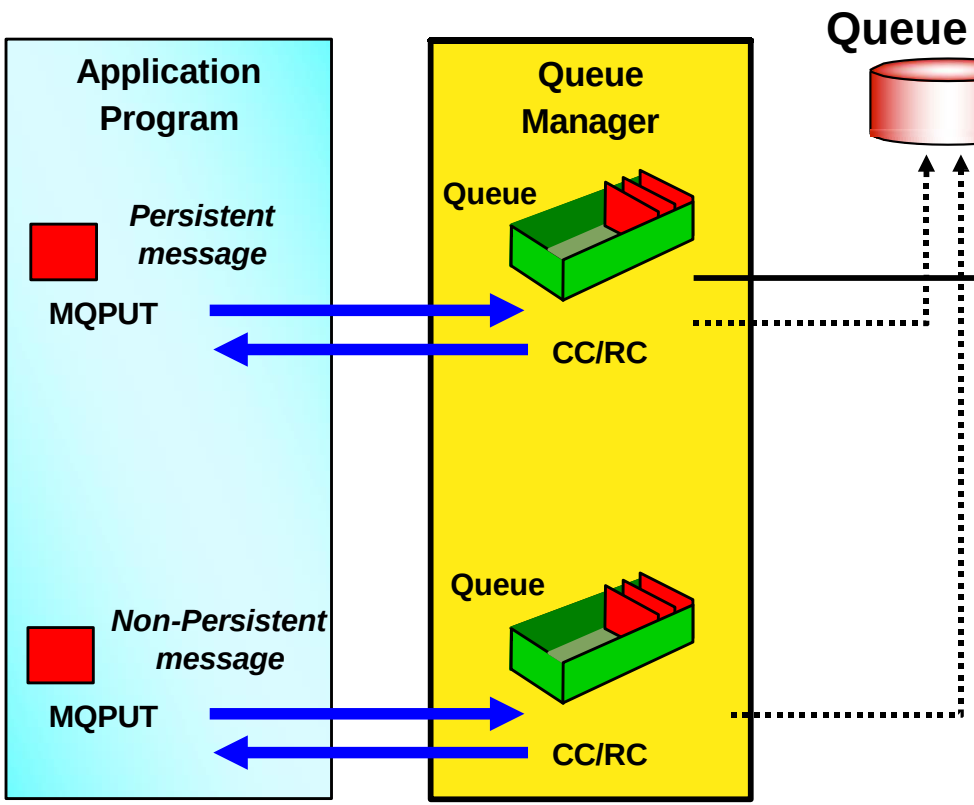
- Messaging and queuing requests are handled by the queue manager kernel. This component is OS independent

- Queue I/O and Logging are handled by DAP (Data Abstraction and Persistence) component. This component interoperates with the OS and file system and therefore is somewhat OS dependent.

- Communications (needed for distributed queuing) is handled by a separate components (listeners, MCAs, etc). This component is protocol dependent (TCP, SNA, etc).

- Also review utilities like Command Server, Trigger Monitors, etc.

How are Messages Persisted?



- 메시지는 큐파일에 기록
- Persistent messages ➔ 로그에 기록
  - Persistent Messages 언제나 복구가능
  - 로깅은 성능에 영향을 미침
- Non-persistent Messages 는 두개의 클래스 서비스를 가짐:
  - 메시지는 Qmgr의 수명동안 유지
  - Qmgr 의 정상종료 및 재시작

Notes - How are messages persisted?

Messages may be written to files, which back the queue. The queue manager will attempt to avoid this by holding the messages in buffers, only spilling to disk if the queue buffers fill.

Persistent messages that have been written (MQPUT) by an application will also **always** be logged - will **never** be lost or discarded (unless expiry specified).

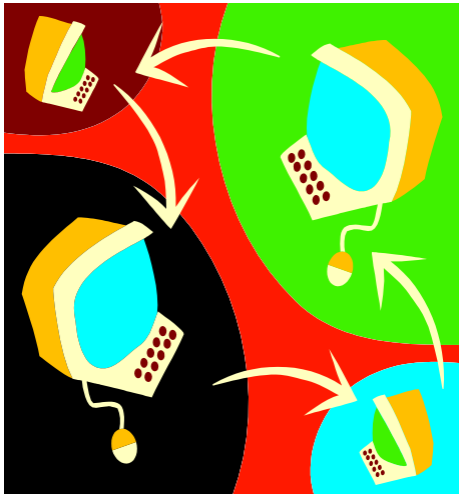
Non-persistent messages may or may not be written to disk. They are **never** logged – can be lost or discarded, even in non-failure situations.

Messages are always put to a **local** queue. Messages destined for remote queue are therefore written to a local intermediate queue (called a transmission queue).

Messages sent by a connected queue manager that cannot be delivered to the target queue are written to a special queue called the Dead Letter Queue (DLQ).

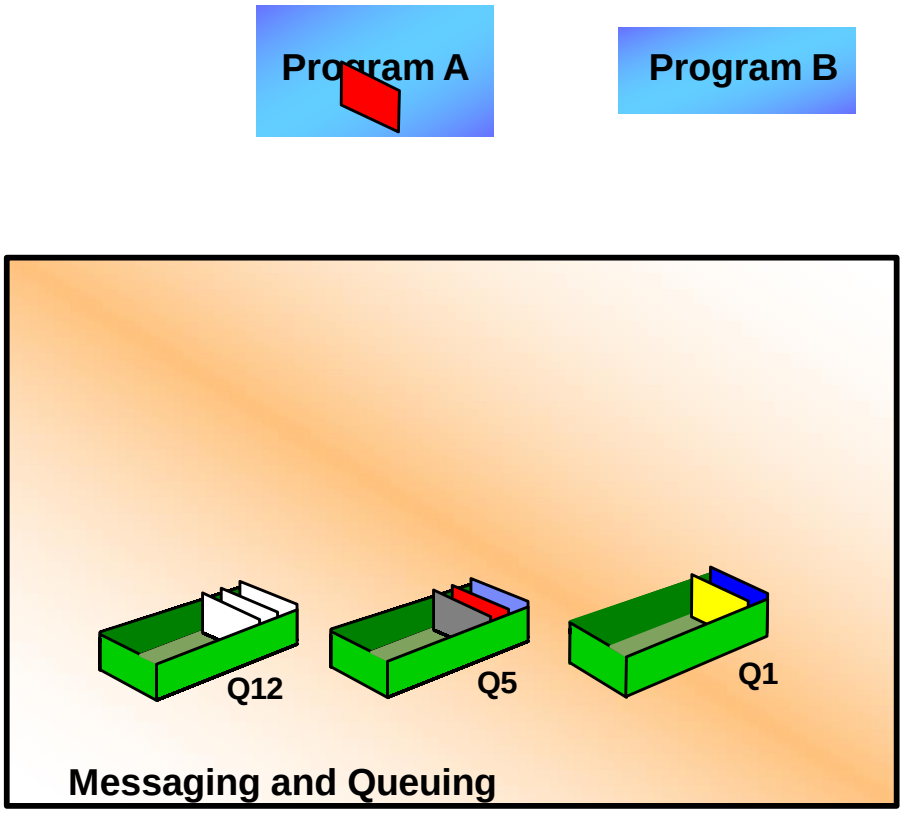
What are Channels?

- Queue Manager 간 통신
  - Uni-directional
  - Usually defined in pairs for example:
    - One Sender
    - One Receiver
  - Asynchronous
- Client 와 Queue Manager간 통신
  - Bi-directional
  - Defined as a single channel
  - Synchronous
- 신뢰할 수 있는 엔터프라이즈 서비스 버스(ESB)의 구성 요소



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

Reliable, asynchronous communication with IBM MQ



메시지 수신
– Accept Message

Receive message from application

Manage “unit of work”

보안 적용
- Apply Security (optional)

Access Control

(permission to get/put by queue or topic)

메시지 전달 –
Deliver Message(s)

Deliver message to application

Ensure Exactly Once Delivery (even after a failure)

Manage “unit of work”

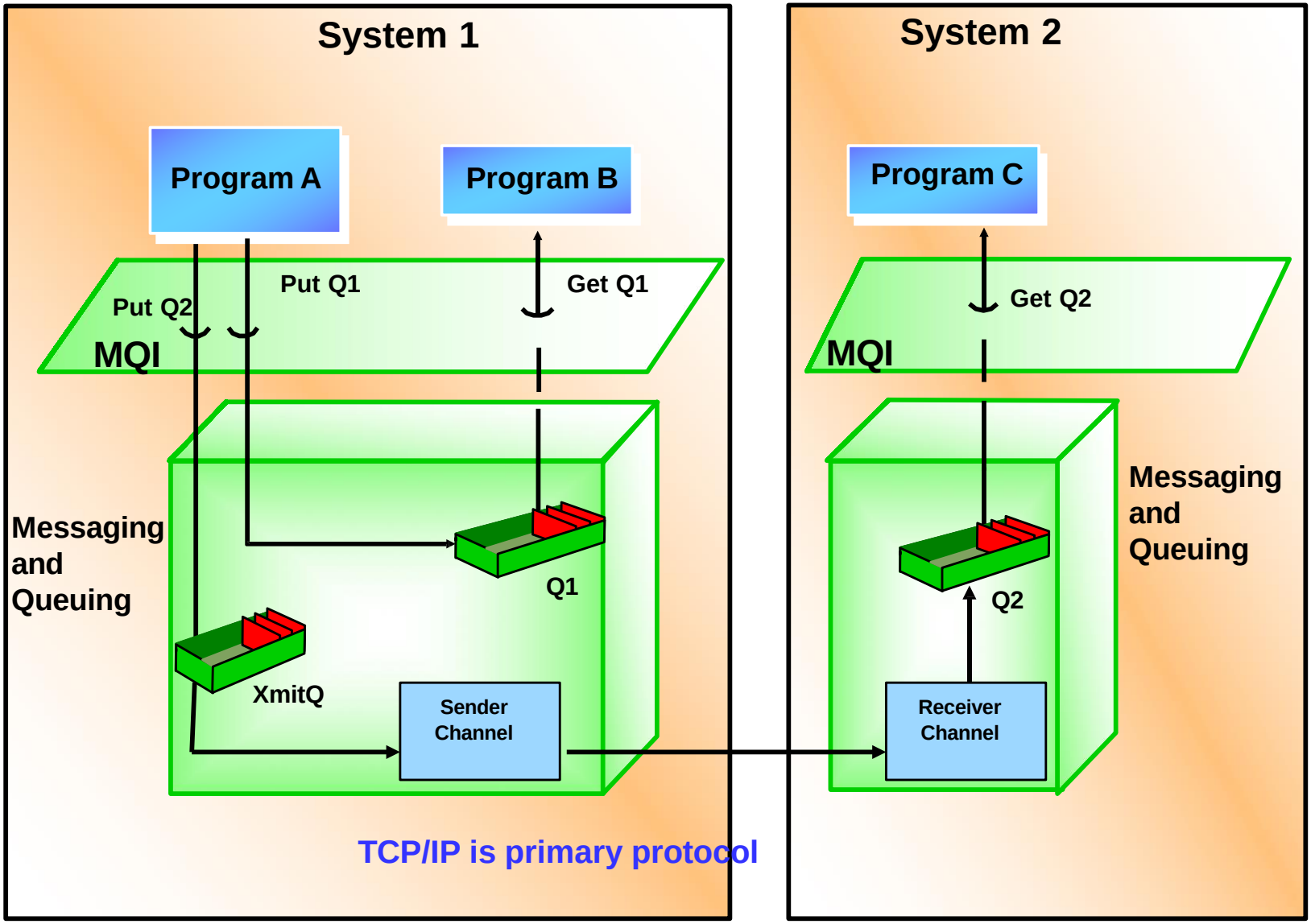
Notes - Reliable, asynchronous communication with IBM MQ

IBM MQ provide Message Oriented Middleware that shields applications from the complexities of network topology (locating each other, even after a move) and network protocols (TCP, SNA). It provides recovery from unavailable applications, network errors and failed applications through the use of message queues.

In this animation, Program A put a message to Q1. Because queuing by default is FIFO, the message is put at the end of the queue.
Program B is the message consumer. Again because queuing by default is FIFO, the messages are consumed from the front of the queue.
There can be any number of queues involved, and neither the queues or the message producers/consumers need to be on the same machine.

Explain the queue managers role in this:
- Receive message from application
- Manage “unit of work” (optional)
- Apply Security (optional)
- Deliver message(s) to application
- Ensure Exactly Once Delivery (even after a failure)

Queues can be Local or Remote



Notes - Queues can be Local or Remote

Program A and B communicate locally via Q1 – no channels involved.

Program A and C communicate remotely via Q2 – channels involved, using an XmitQ. The XmitQ allows Program A to operate as though Q2 were local (because the XmitQ is in fact local).

So neither program need have any awareness of the queue manager topology or the underlying network details.

Developing IBM MQ Applications

- IBM MQ supports a wide range of platforms
  - Windows®, UNIX® , Linux ®, z/OS®, i5OS®, HP NonStop, etc
- With a whole range of programming languages
  - Java, C/C++, C#, .NET, COBOL, Node.JS, TAL, etc…
- And a wide range of Interfaces
  - MQI, JMS, .NET, REST, MQ Light, etc…
- A large number of sample programs are provided to show how to develop MQ applications for various languages and environments, for example:
  - Put sample amqsput
  - Get sample amqsget
  - Browse sample amqsgbr
  - Publish sample amqspub
  - Subscriber sample amqssuba
  - …and many others

Notes - Developing IBM MQ Applications

MQ는 다양한 플랫폼, 서로 다른 인터페이스, 여러 언어에서 사용할 수 있기 때문에 제품과 함께 다수의 샘플 프로그램이 제공됩니다. 이 슬라이드에는 그 중 일부가 나열되어 있습니다.

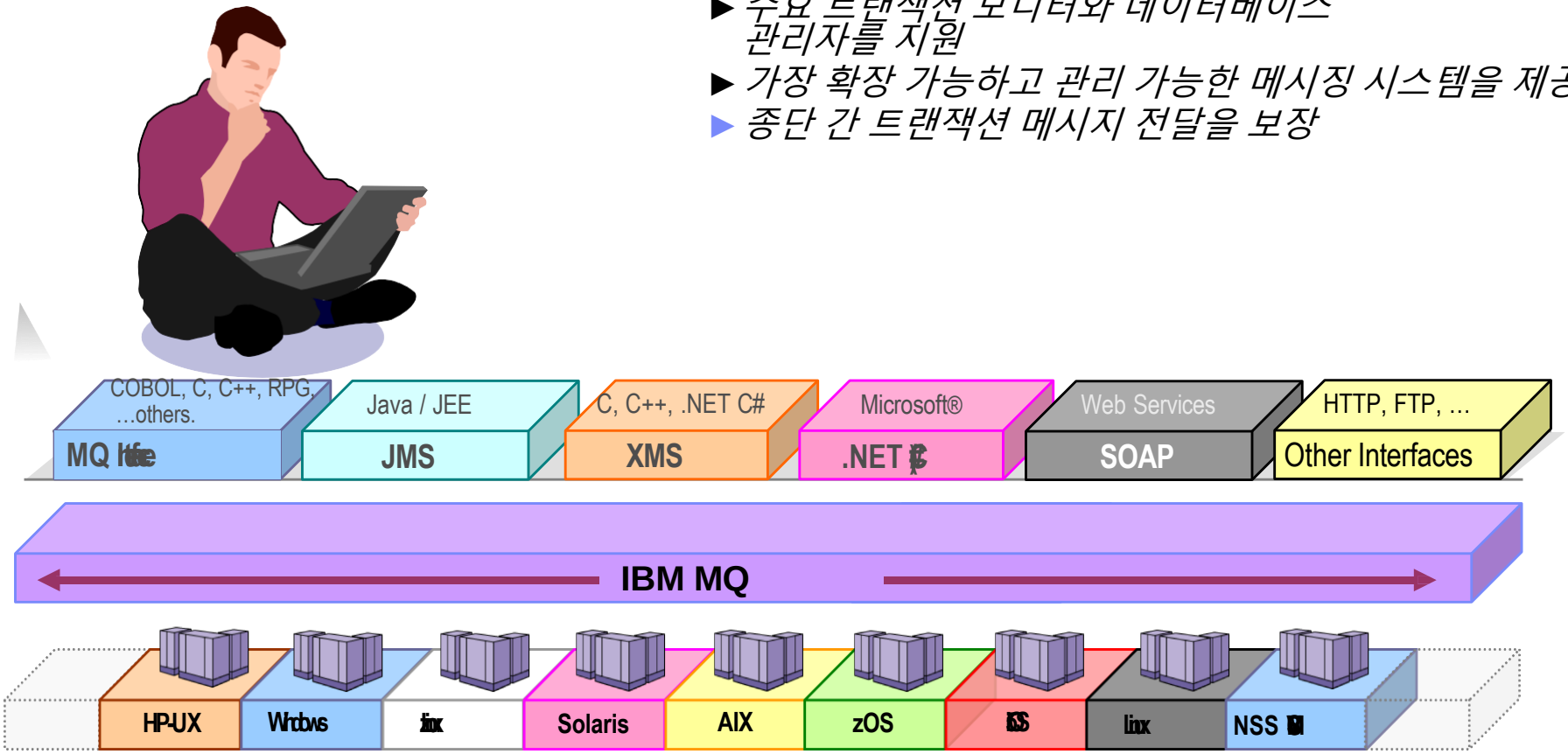
이 샘플들은 다양한 방식으로 MQ에 접근하는 프로그램의 동작을 보여주는 데 사용할 수 있으며, 소스 코드는 학습 도구로 활용되어 자신만의 환경과 언어에서 MQ 애플리케이션을 개발하는 방법을 더 잘 이해할 수 있도록 도와줍니다.

The solution to Universal Connectivity ➔ IBM MQ

IBM MQ는 모든 애플리케이션
메시징 트래픽에 대해 단일 관리
가능한 분산 인프라를 제공함으로써
애플리케이션 인프라 비용을 크게
절감

Features:

- *IBM MQ는 가장 광범위한 API, 프로그래밍 언어 및 운영 체제 플랫폼을 지원*
- *"모든" 표준 준수 JEE 서버에서 구현할 수 있는 유일한 JMS 엔진을 제공*
- *발행/구독을 포함하여 다양한 서비스 품질과 메시징 방법을 제공*
- *주요 트랜잭션 모니터와 데이터베이스 관리자를 지원*
- *가장 확장 가능하고 관리 가능한 메시징 시스템을 제공*
- *종단 간 트랜잭션 메시지 전달을 보장*



80+ platform configurations

Notes - The solution to Universal Connectivity ➔ IBM MQ

Key Points:

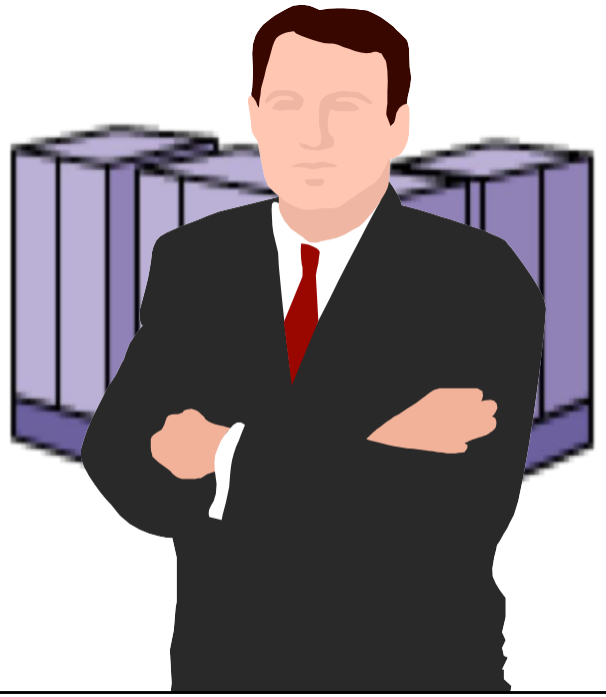
There is nothing else quite like MQ in the Market today. It is the market leader in Message Oriented Middleware. It is The Universal Communicator

MQ's key strength is its breadth:

- With a whole range of programming languages – including Java, C/C++, C#, .NET, COBOL,…
- A wide range of Interfaces from MQI to JMS, XMS and .Net. Other interfaces that can be used are to HTTP & FTP
- This support means that applications can be connected without modification
- Also virtually any IT commercial platform – including native z/OS is supported.
- This means your customers can integrate anything they have and use the skills they already have…

IBM MQ Enterprise Class Messaging

- 검증된 확장성
  - Grow your network **incrementally** one server at a time
- Performance
  - Many clients are moving **millions of messages** per day
- 강력한 관리기능 (Administer massive networks)
  - Cross-platform, remote configuration tooling
  - for enterprise-wide systems administration
- 모든 IT 플랫폼에 가상으로 접근 가능
- MQ for z/OS
  - Built from the ground up to exploit zSeries platform
  - Consistent with MQ on distributed platforms
- Clustering on distributed, shared queues on z/OS
  - For **High-Availability** and workload balancing
  - Easier to set up than you may think!
- 멀티 스레딩
  - Exploits **multi-processors** for high-speed throughput
- 고가용성 : Multi-instance queue managers and replicated data queue managers (Linux only)
  - Software high-availability option
- Security
  - Industry-standard **SSL support**
  - Certified for **Common Criteria**
  - Policy-based security with **MQ Advanced Message Security**
- 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**

Notes - IBM MQ Enterprise Class Messaging

MQ has proven scalability that allows you to grow you adoption as and when your business requires.

Many of our customers are processing 100’s of messages a second and 100s Millions for messages a day

No other messaging oriented middleware product supports as many commercial IT systems as MQ just ask an analyst

Clustering transparently allows dynamic work load distribution and failover without altering your MQ applications

MQ is designed to exploit threading models on different operating systems
Rich support for security

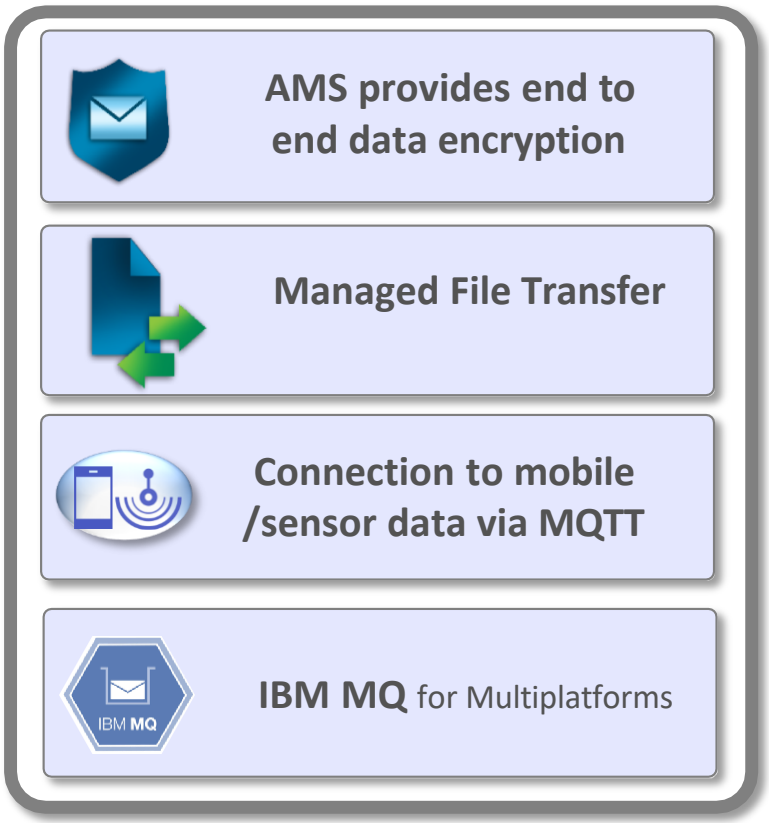
And when you have a problem – you want to make sure you have world class first class support.

What forms of MQ are available?

- IBM MQ for multiplatforms
- IBM MQ Advanced –IBM MQ for multiplatforms + 추가기능, Managed File Transfer, Advanced Message Security, MQ Telemetry Transport
- MQ Appliance –IBM MQ 가 실행되는 물리적 어플라이언스 MQ Advanced firmware 내장

What is MQ Advanced and what is included?

Components of the MQ Advanced offering:



Advanced Message Security(AMS)는 메시지 내용을 종단 간 암호화하여 모든 형태의 침입, 공격 또는 우발적인 노출로부터 민감한 데이터를 보호하며, 애플리케이션 변경이 필요하지 않습니다

Managed File Transfer(MFT)는 신뢰할 수 있고 안전하며 감사 가능한 파일 전송을 제공하여 수동 프로세스의 필요성을 줄이고, 공유 MQ 인프라와 도구를 사용하여 실패 분석 시 낭비되는 시간을 줄이는 데 도움을 줍니다. MQ Advanced에서는 무제한 MFT 에이전트를 제공

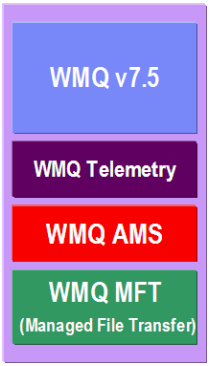
MQTT는 센서, 모바일 및 사물인터넷(IoT) 디바이스를 기존의 엔터프라이즈 애플리케이션 및 서비스와 빠르고 쉽게 연결하도록 지원

온프레미스, 클라우드 또는 하이브리드 환경에서 애플리케이션, 시스템 및 서비스 간에 비즈니스 데이터를 빠르고 신뢰할 수 있게 교환할 수 있도록 지원

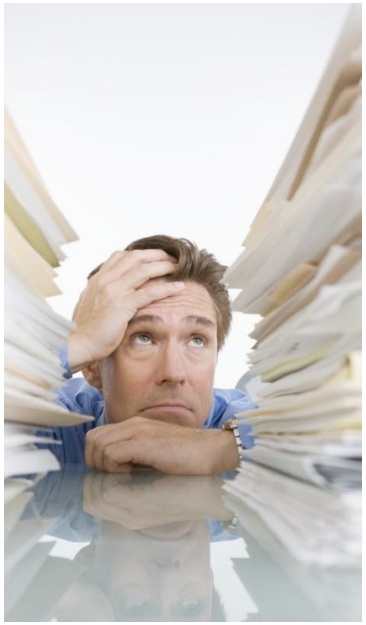
IBM MQ Advanced - AMS (Advanced Message Security)

- Secures sensitive or high-value MQ messages
  - Privacy via message content encryption
  - It leverages digital certificates (X.509) and Public Key Infrastructure (PKI) to protect MQ messages
- Detects and removes rogue or unauthorized messages before they are processed by receiving applications
  - Authentication via certificate *above and beyond* operating system
  - Authorization to queue *above and beyond* MQ OAM or SAF
- Verifies that messages are not modified in transit
  - Message Integrity via digital signature of message content
- Protects messages not only when they flow across the network but when they are at rest in queues
- Messages from existing MQ applications are transparently secured using “interceptors”
  - No application changes are necessary
- No pre-requisite products other than MQ

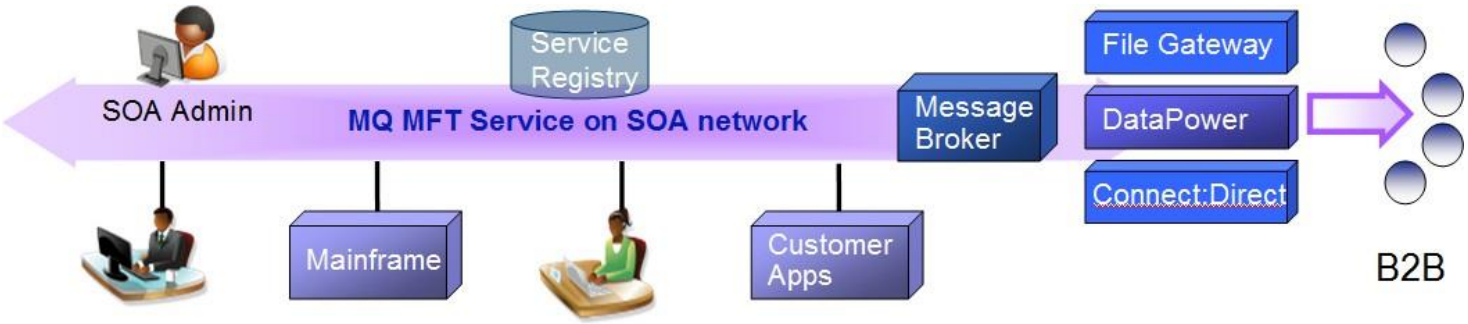
IBM MQ Advanced - Managed File Transfer



- IBM MQ 인프라와 기술을 활용
- 신뢰성 있는 파일전송
- 추적 가능한 파일전송
- 자동화 및 통합



- Auditable
- Reliable
- Secure
- Automated
- Centralized
- Any file size
- Integrated
- Cost Effective



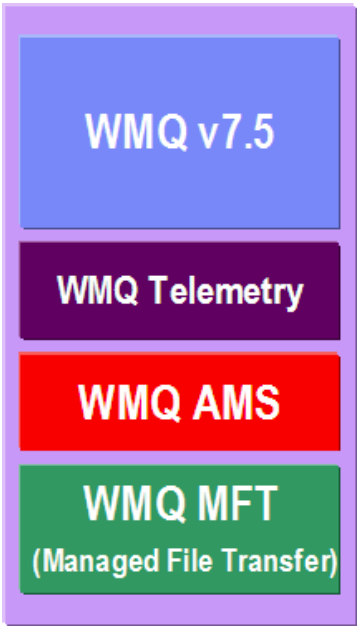
Notes - IBM MQ Advanced - Managed File Transfer

MQ MFT –

Reliability leveraging the MQ transport
Multi-purpose – use for messages and files
Auditable with logging subsystem that tracks transfer at source and at destination for audit purposes
Centralized monitoring, control and configuration
No need to program – no need to use APIs
Graphical tooling enabling remote configuration
Command line interface – for advanced users
Powerful scripting support – enabling complex multi-step transfers using XML and Ant
Flexible backbone – move files from anywhere to anywhere in network
Massive files – larger than MQ messages
Integration with MQ-enabled apps and ESBs
Automatic file character conversion
Security of file payload using SSL
Visual transfer status reporting
Support for range of platforms

IBM MQ Advanced - MQTT

- 신뢰성 있는 전송 제공
- 효율적인 기능 제공
  - Push notification: no polling required
  - Publish/Subscribe: single message to many
- 배터리와 대역폭 절약
- 보안 연결 지원



IBM MQ Messaging

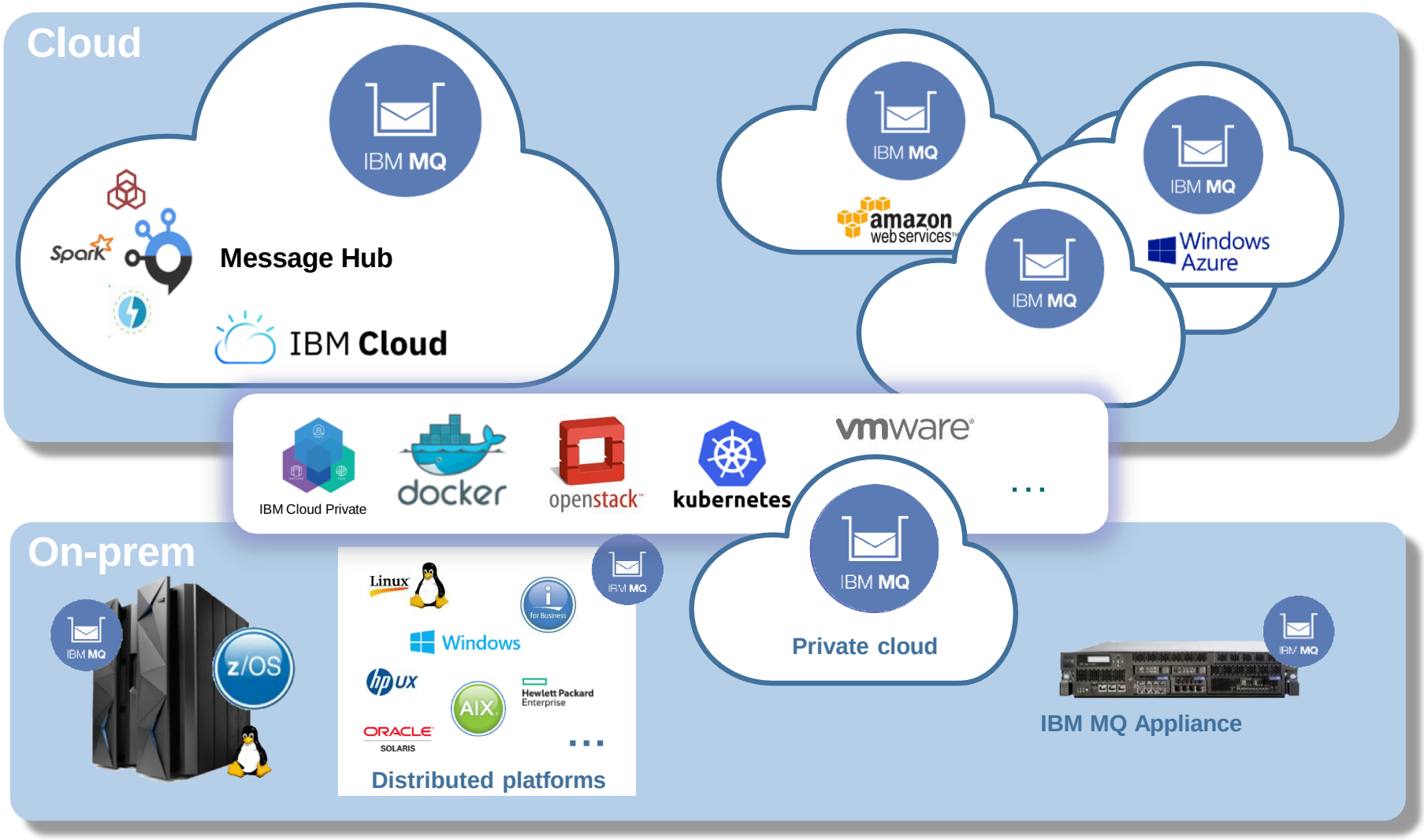
Anytime, anywhere business transactions

Connecting the enterprise back-end services in a secure and scalable manner with Visibility, Security & Management of Mobile platform

The IBM MQ Appliance

- 확장성 , 보안성 및 신뢰성
- 빠른 구축과 낮은 총 소유 비용
- 내장된 고가용성 및 재해 복구 기능
- 고속 대용량 SSD 지원
- 메시징 허브로 이상적
- 익숙한 사용자 경험

Where can I run IBM MQ? Everywhere!!!



Notes - Where can I run IBM MQ? Everywhere!!!

Where do you want to run MQ? We talked about some of the different options for MQ. As you can see, MQ is very flexible, and whether your business is cloud-based, is not cloud-based, or is something in between, MQ is a valuable tool. You can run it as software, on an appliance, and push workloads to the cloud. If you want to develop applications in the cloud, we recommend exploring Message Hub, in our Bluemix cloud. MQ also runs on the z/OS mainframe.

Multiple APIs and Protocols

다양한 API 와 프로토콜 지원을 통한 상호 운용성 제공

APIs: **MQI**, **JMS**, **MQ Light**, **REST** …

Protocols: **MQ**, **AMQP**, **MQTT**, **HTTP**

폭넓은 애플리케이션 스타일 지원

IBM MQ 의 장점
* 유연성
* 확장성
* 표준 준수

APIs and Protocols

MQI
Exposes the full set of MQ capabilities
Uses the MQ protocol

MQ
MQ's highly reliable and performant messaging protocol

JMS 2.0
Supports the full JMS API for use in many JSE or JEE environments
Uses the MQ protocol

MQTT
Supports the open standard MQTT API and protocol
Open source Eclipse Paho clients

MQ Light
A simple pub/sub messaging API
Uses the AMQP 1.0 protocol

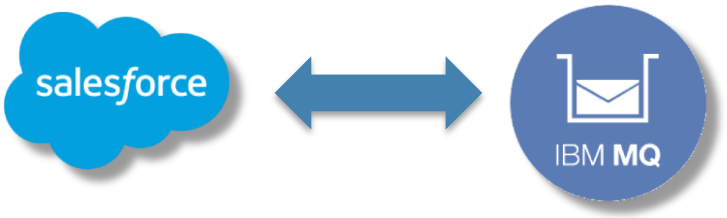
AMQP
Support for AMQP 1.0 enables support for open source clients such as Qpid Proton

REST
A very simple but secure messaging API over REST
…

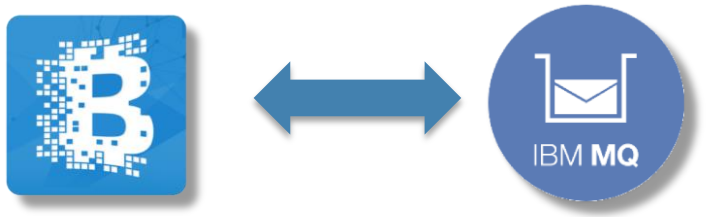
Bridging MQ with external systems

As well as connecting a wide array of applications directly to an MQ system, there are a growing set of bridges and connectors between MQ and external systems

Salesforce
Integrate MQ's publish/subscribe with Salesforce. Exchange **Salesforce events** and **MQ publications** using the MQ Bridge for Salesforce with no need for your backend applications to connect to Salesforce directly.



Blockchain
Use MQ messages to **query** and **update** a Blockchain ledger. Connects to Hyperledger Fabric networks in IBM Cloud and locally. Supported for use with **V9.0.x MQ Advanced** queue managers



Kafka
IBM MQ **sink** and **source** connectors are currently being openly developed by IBM and provides **as-is**, allowing you to connect your MQ systems with your Kafka clusters
www.confluent.io/product/connectors