Message-Queuing (MQ) Systems

- Message-queuing systems support asynchronous persistent communication. They are also called Message–Oriented Middleware (MOM).
- Applications communicate by inserting messages in queues
  - Queues are provided by the MQ system
  - Applications can only add/retrieve messages to/from local queues (i.e., queues on the same machine or on a nearby machine)
  - Sender’s local queue is called the source queue and the receiver’s queue is called the destination queue
  - The MQ system transfers messages from source queue to destination queue
  - Every message contains the name of the destination queue

MQ Systems (continued)

- A sender is given the guarantee that its message will be eventually inserted in the recipient’s queue
  - No guarantee on when or if the message will be read, which is determined by the recipient
  - Communication is loosely-coupled in time: Sender and receiver can execute independently of each other

Four combinations for loosely-coupled communications using queues.
Basic Interface Offered to Applications

<table>
<thead>
<tr>
<th>Primitive</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Put</td>
<td>Append a message to a specified queue (nonblocking)</td>
</tr>
<tr>
<td>Get</td>
<td>Block until the specified queue is nonempty, and remove the first message</td>
</tr>
<tr>
<td>Poll</td>
<td>Check a specified queue for messages, and remove the first. Never block</td>
</tr>
<tr>
<td>Notify</td>
<td>Install a handler to be called when a message is put into the specified queue</td>
</tr>
</tbody>
</table>

General Architecture of a MQ System

- A MQ system should maintain a mapping of queue names to network locations in order to transfer messages to their destination queue
  - The mapping is maintained in a (possibly distributed) database
General Architecture of a MQ System (continued)

- In many MQ systems, there is no naming service that can dynamically maintain queue-to-location mappings
- One solution is to have each queue manager maintain a copy of the queue-to-location mapping
  - A message can be directly sent to the destination queue manager
  - This is not scalable: every queue manager must be updated when queues are added or removed
- Routers (or relays) can be used to achieve scalability
  - A message is transferred from the source queue to the nearest router, which then forwards the message toward the destination
    - Each router maintains a mapping of destination queue names to next routers
    - This is scalable: Only routers need to be updated when queues are added or removed. Queue managers only need to know where the nearest router is.
      - Note that the number of routers is much smaller than the number of hosts

The general organization of a message-queuing system with routers.
Message Brokers

• A MQ system is often used to integrate different applications into a single coherent distributed information system
• Different applications may have different message formats → Message brokers can be used to handle conversion between different message formats
  — A message broker maintains a repository of rules and programs that can transform a message of one type to a message of another type

MQ Systems vs Email Systems

• Both MQ systems and email systems support asynchronous persistent communication
• There are a number of differences
  — No routing in email systems: messages are sent directly from the source mail server to the destination mail server
  — MQ systems are more general purpose than email
    • MQ systems enable communication between processes
    • Email systems support only end users
Example: IBM’s Websphere MQ

- Sending application put messages in a local queue; receiving application remove messages from a local queue.
- All queues are managed by queue managers, which handle outgoing and incoming messages
- Queue managers and applications can run on the same machine or on separate machines
  - If on separate machines, the application communicates with the queue manager using RPC

General organization of IBM’s message-queuing system.
Message Channels

- Queue managers are pairwise connected through *message channels*
  - A message channel is a unidirectional, reliable connection (e.g., a TCP connection) between a sending and a receiving queue manager, through which queued messages are transported
- At each endpoint of a channel is a *message channel agent* (MCA) responsible for
  - Setting up channels using transport-level communication facilities (e.g., TCP)
  - Wrapping messages in the send queue into transport-level packets and sending them (sending MCA)
  - Receiving packets, unwrapping them, storing unwrapped messages into appropriate queue (receiving MCA)

Message Transfer

- Each message carries a destination address consisting of the *name of the destination queue manager* and the *name of the destination queue*
- Each queue manager has a routing table (maintained manually)
  - A table entry is a pair (destQM, sendQ)
  - Meaning of the entry: If destination queue manager of message $M = \text{destQM}$, then $M$ should be appended to $\text{sendQ}$
  - Note: Each message channel has exactly one send queue, so $\text{sendQ}$ effectively specifies to which queue manager $M$ is to be forwarded
- Aliases can be used to allow applications to use the same logical name for a queue even if the queue manager of that queue changes
  - An alias defined within a queue manager $M1$ is another name for a queue manager $M2$, which is only available to applications interfacing to $M1$
  - Changing the name of a queue manager requires that we change its alias in all queue managers. However, applications are not affected
The general organization of an MQ queuing network using routing tables and aliases.