Integration of Distributed Applications in Message Oriented Environment
Business Needs

- Typical Business Information System consist of numerous services and applications, that are supposed to work together in order to deliver positive business results.
- When business grows, information systems that empowers it cannot easily grow as well (demanding unpredictable costs in time and money).
- Companies that can effectively use IT systems to adopt the changing business conditions will gain significant competitive advantage.

“For enterprises, crucial issue is how to integrate applications, not whether to integrate!”
Known problems

In enterprise, different applications need to exchange data and information between them, which generate variety of problems. Often, business applications communicate only via direct access to database, which allow them to skip business logic rules (if they are not implemented with stored procedures and functions on database server).

Question: HOW TO INTEGRATE DIFFERENT

• Operating systems?
• Programming languages?
• Application platforms
• Database management systems (DBMS)?

Question: HOW TO

• Increase flexibility?
• Decrease costs of ownership?
"Distributed applications are programs that run on more than one computer and communicate through a network, or at a single machine. Some distributed applications are actually two separate software programs: the back-end (server) software and the front-end (client) software."

Indiana University Information Technology Services

**Ingredient needed for Integration:**

**Messaging infrastructure** to allow different systems to communicate through a shared set of interfaces
Synchronous or Asynchronous Communication?

Example: Synchronous Communication

Remote Procedure Call communication
- Remote objects and RMI

::characteristics::

- server blocks client until response is sent
- easy testing and debugging
- no need for middleware software
- all processing resources and network must be up and running

Scenario 1.

Example: Asynchronous Communication

Message Oriented Communication
- Communication based on MOM
(Message Oriented Middleware)

::characteristics::

- communication less sensitive to network unavailability
- communication mechanisms assures safe message delivery
- platform independent
- need middleware software
What is Middleware Software?

“In a distributed computing system, **middleware** is defined as the software layer that lies between the operating system and the applications on each site of the system”

ObjectWeb consortium
(http://consortium.objectweb.org)

• **The concept of middleware appeared as networked systems became increasingly dependent on sophisticated protocols and architectures.**

• **It is an area of computer systems architecture that gained wide recognition from about 2004 onwards.**
Object Oriented Middleware

**Object Oriented Communication is based on RPC:**

- A **Remote Procedure Call (RPC)** is a protocol that allows a computer program running on one computer (host) to cause code to be executed on another computer.
- When the code is written using object-oriented principles, RPC is sometimes referred to as *remote invocation* or *remote method invocation*.

**Examples of RPC oriented protocols:**

- **CORBA** (Common Object Request Broker Architecture)
- Microsoft **DCOM** (Distributed Component Object Model)
- **RMI** (Java Remote Method Invocation).

**::characteristics::**

- telephone like communication (synchronous)
- programmers need to have specific knowledge to create distributed application
- simple and easy to use exception handling
- request/response communication
The primary advantage of a message based communications protocol is the ability to store, route or transform the message as it is being delivered.

Characteristics:

- **Storage**: persistent; non persistent
- **Transformation**: message format transformation with message transformation tools
- **Routing**: unicast; broadcast, multicast
Data Exchange Based on MOM

Middleware

Client Software

Visual C++ Application

Middleware Client Software

IBM Websphere MQ
Microsoft Message Queue Server (MSMQ)
Tibco TIB/ETX
...

Connection

DATABASE SERVER

Middleware Client Software

COBOL Application

Unix OS
MOM ADVANTAGES

MOM (Message Oriented Middleware) utilizes some great improvements over traditional (RPC) application integration strategies:

- Simple Application Development model
- Communication less sensitive to network unavailability
- Communication mechanisms assures safe message delivery
- Easy integration of independent and different computer platforms (Linux, Windows, Macintosh OS..)
- Fault-tolerant communication applicable to long geographic distances
- Messages can be prioritized and load-balanced
- Less data overhead in communication (lower cost)
- Less demanding design on communication system (works well on slower network connection)
- Messaging integrates heterogeneous systems without sacrificing flexibility
MOM DISADVANTAGES

- lack of standard (all the major vendors have their own implementations, each with its own API and management tools)
- Sometimes, messages cannot easily represent data to exchange between hosts
- Commercial middleware software can be pricey
- MOM client software must be installed and configured on every client machine (more human labor required)
WHEN TO USE MOM?

• What are our application-to-application latency requirements?
• Do we need fault tolerance if a node or network fails?
• What kind of service are we supporting with MOM:
  • Peer-to-peer? Client-server? One-to-many?
• Do we need to connect applications hosted on different platforms, and written in different programming languages?
• Will extra component in the architecture (MOM) generate problems?
Products and Vendors

**Commercial software**
- IBM WebSphere MQ
- Microsoft Message Queue Server (MSMQ)
- Oracle Advanced Queuing (AQ)
- BEA Systems MessageQ
- Arjuna Messaging
- ..

**Open source software**
- ObjectWeb JORAM
- Open Source Message Queue (OSMQ)
- ..
CONCLUSION

Good Message Oriented Middleware implementations provide a high-level applications interface, quality of service guarantees, and a host of services such as security, message queuing, and directory support that are necessary for "industrial-strength" distributed communications.