Message-Oriented Middleware for Enhancing Web Services interoperability

CAS 747: Software Architecture and Reverse Engineering
Institutor: Kamran Sartipi
by Jiacong Zhang (Kevin)
March 21st, 2006

Outline

- What is Middleware?
- What is Message-oriented Middleware?
- What is the difference between PRC and MOM?
- What is Message Queues and Messaging Model?
- What is Service-Oriented Architecture?
- How can MOM concept help to enhance Web Service Interoperability?
- Conclusion

What is Middleware?

- A software layer

- Sitting above the network operation system layer and below the application layer

- Abstracting the heterogeneity of the underlying environment

- Providing a platform to simplify programming and managing distributed application

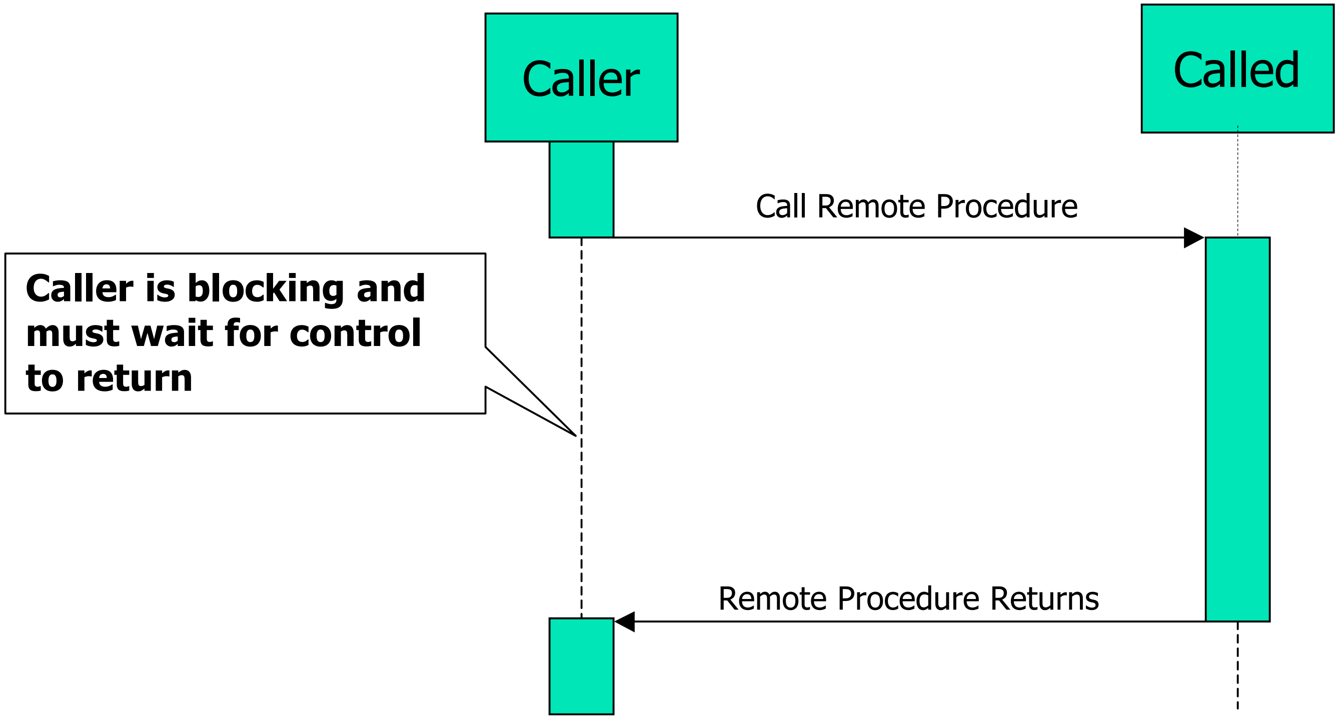
- An integration and interoperability of applications and services running on heterogeneous computing and communications devices

Middleware Platform

- Offering higher-level distributed programming models that extend the native OS network environment
- Allowing developers to develop distributed applications like stand-alone applications
- Masking the heterogeneity of networks, hardware, operation system, even programming languages
- CORBA, DCOM, RMI, WEB SERVICE

Interaction Models(1)

- Synchronous Communication

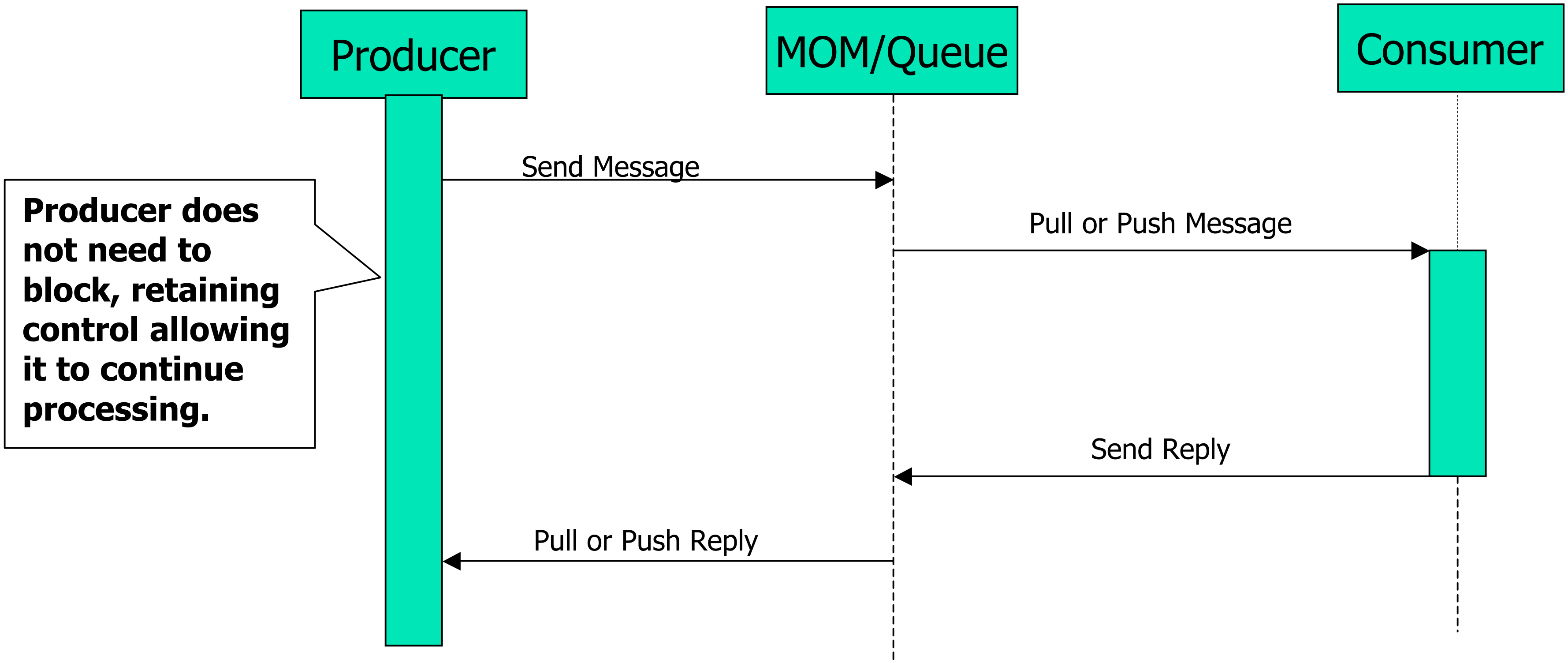


Caller is blocking and must wait for control to return

System do not have processing control independence; they rely on the return of control from the called system.

Interaction Models(2)

- Asynchronous Communication

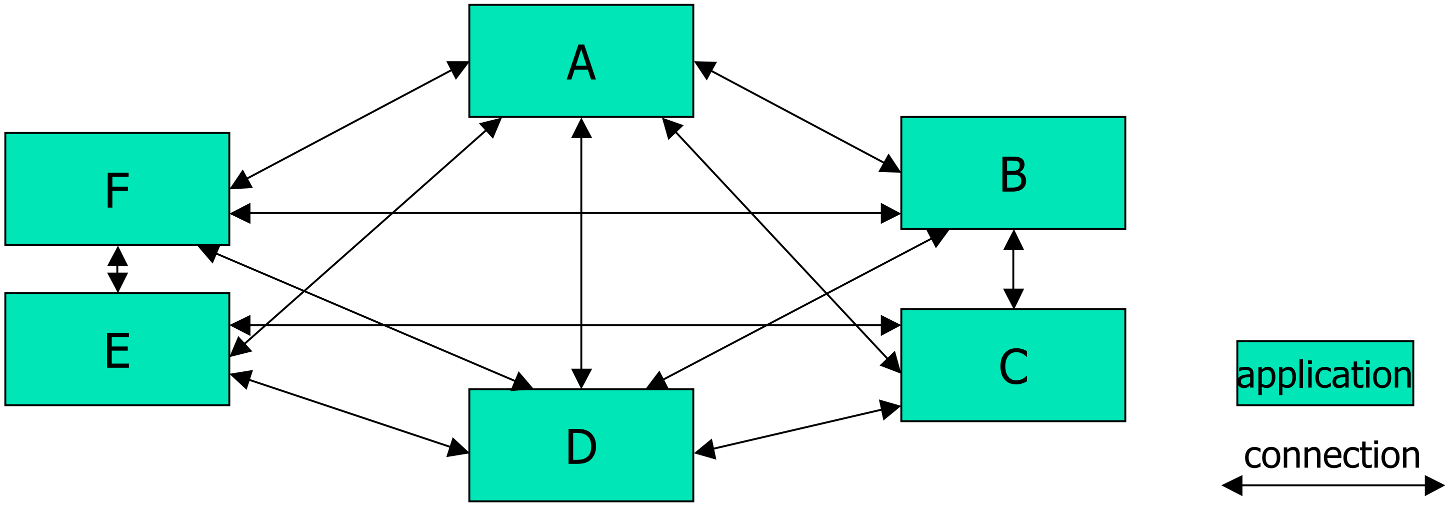


Producer does not need to block, retaining control allowing it to continue processing.

Participants can continue processing, regardless of the state of the other participants.

Remote Procedure Call (RPC) (1)

- A fundamental concept of distributed computing
- Allowing 2 processes to interact and making them believe they are in the same process space
- Based on synchronized model, similar to a local procedure call

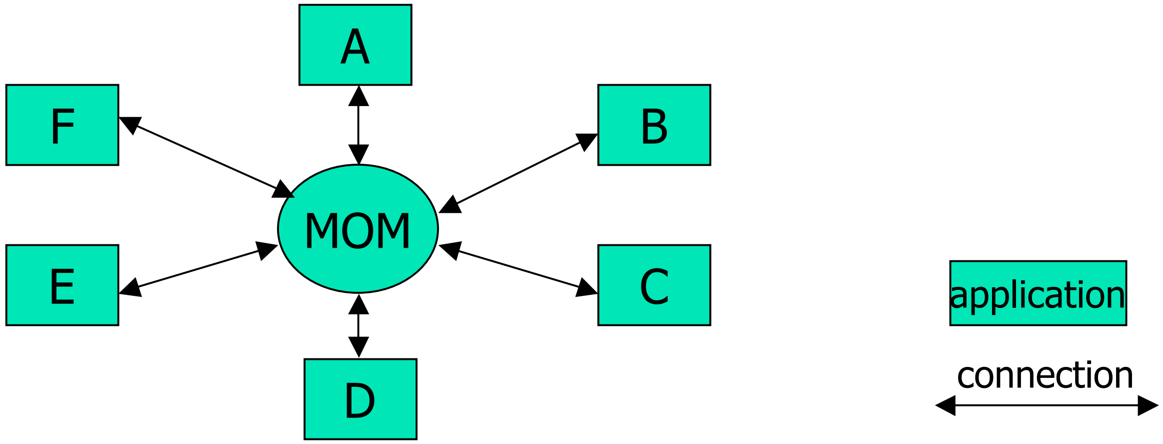


Remote Procedure Call (RPC) (2)

- Coupling
  - Designed to work on object or function interface
  - Producing tightly coupled system
  - Invasive mechanism of distribution
- Reliability
  - Providing little or no guaranteed reliable communication
  - Any failure outside the application can affect the reliable transportation
- Scalability
  - Affecting performance where some participating subsystem do not scale equally
  - Using more bandwidth
- Availability
  - Interdependent, requiring all the simultaneous availability of all subsystems
  - Service outrange or system upgrading can cause error for the whole system

Message-Oriented Middleware (MOM) (1)

- Based on asynchronous model
- Participants not required to block and wait on a message send
- Supporting long time message delivery (minutes)
- Offering service based approach to interprocess communication



Message-Oriented Middleware (MOM) (2)

- Coupling
  - Injecting a layer between senders and receivers
  - Acting as an intermediary to exchange messages between senders and receivers
  - Loose coupling between participants, highly cohesive, decoupled system deployment
- Reliability
  - A store and forward mechanism for message persistence
  - Preventing loss of messages
  - Configurable reliability to guarantee a message will be delivered
- Scalability
  - Decoupling the performance characteristics of the subsystems from each other
  - Allowing simple and effective load balancing
  - Supporting large scalable enterprise-level system (e.g handling 16.2 milion concurrent queries per hour and over 270,000 new order requrests)
- Availability
  - Not requiring simultaneous or “same-time” availability of all subsystems
  - Improving response time because loose coupling between MOM participants

When to use MOM or RPC

- RPC model is ideal for strongly typed/Object-Oriented system with tight coupling, compile-time semantic checking and more straightforward system implementation

- MOM model is ideal for the distributed system, which is geographically dispersed deployments with poor network connectivity and stringent demands in reliability, flexibility, and scalability

Message Queues

- A fundamental concept within MOM
- Storing messages, a destination where messages may be sent to and received from
- Many configurable attributes (e.g. queue's name, queue's size, save threshold of queues)
- MOM platforms support multiple queue types with different purpose (e.g. Public Queue, Private Queue, Temporary Queue, Journal Queues, Bridge Queue, Dead-Message Queue)

Common MOM Services

- Message Filtering
- Transactions
- Guaranteed Message Delivery
- Message Format
- Load Balancing
- Clustering

Service-Oriented Architecture (1)

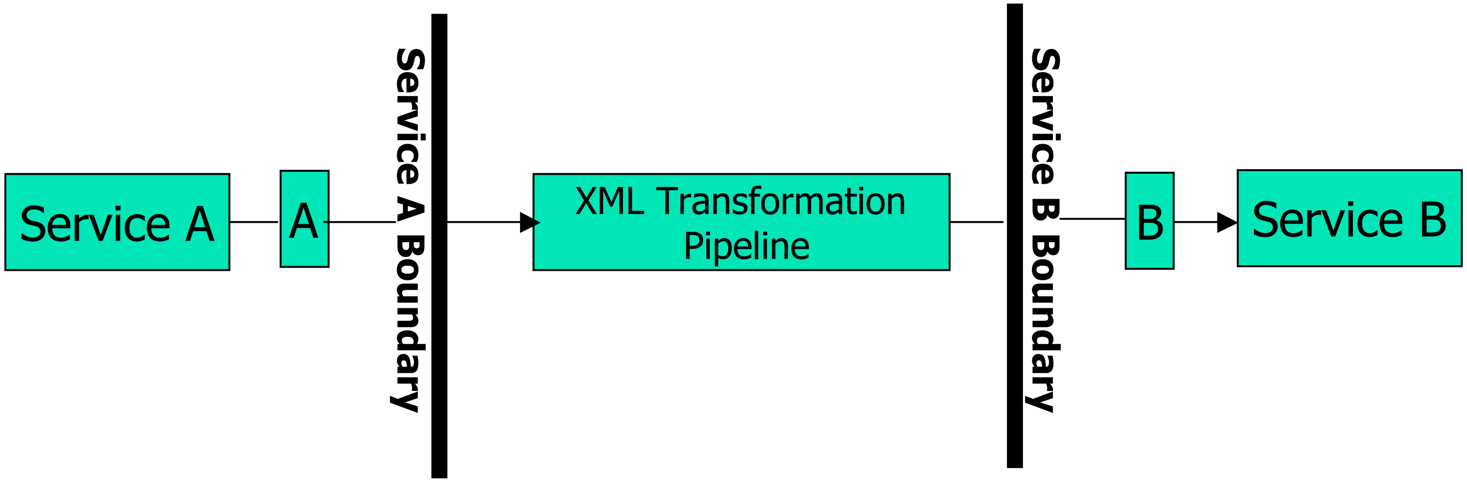
- We want a nonproprietary architectures to utilize the benefits of reusable software components to cut the software cost.
- MOM
  - Creating highly open and flexible systems that allow seamless integration of subsystems
  - Solving many transport issues with integration

MOM cannot solve the problems such as data representation, format and structure, so additional techniques needed.

- XML
  - A programming language and platform-independent format for representing data
  - Using eXtensible Stylesheet Language (XSLT) to transform xml data format
- Web Service
  - Platform and language independent standards defining protocols for heterogeneous system integration
  - Interfaces that allow programs to run over public or private networks with standard protocols (SOAP)
  - B2B/enterprise application integration tool, a evolution of the traditional RPC
- SOAP
  - A mechanism for exchanging structured and typed information
  - SOAP messages can be bound with a transport mechanism (HTTP, SMTP, JMS)

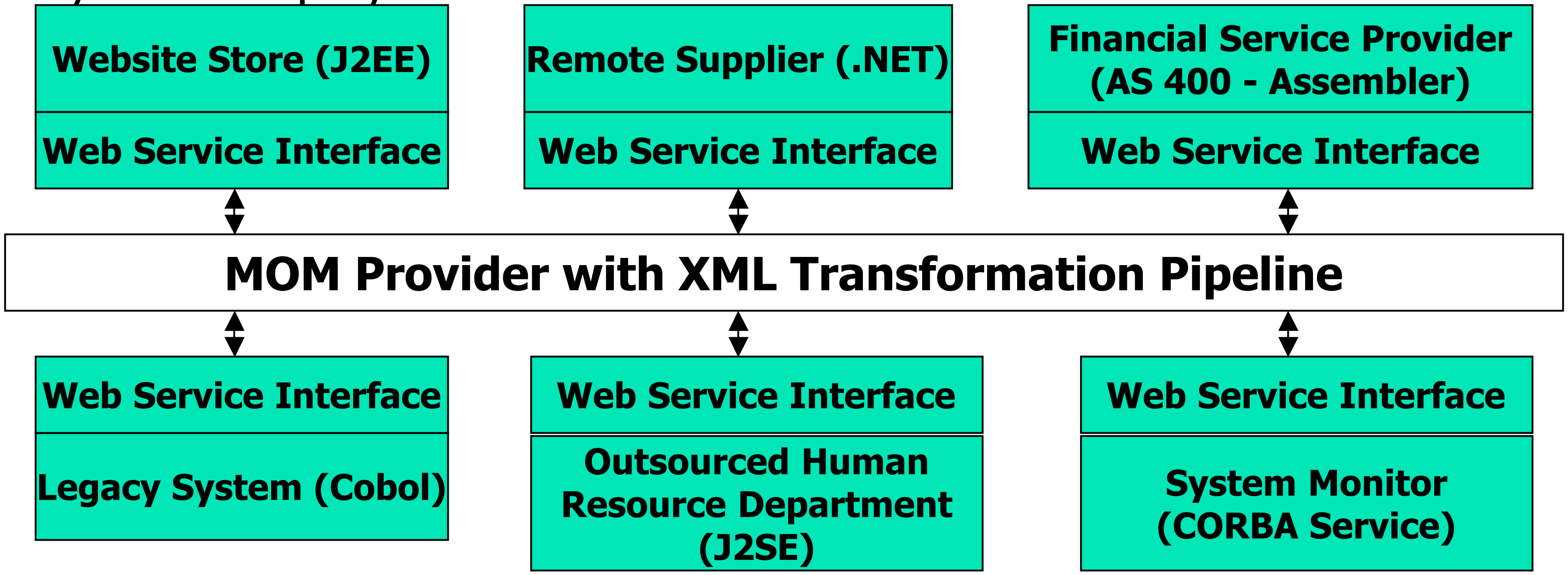
Service-Oriented Architecture (2)

- Reduce application processing to logic black boxes
- Standard XML-based messages can be used to interact with these black boxes
- Message-centric structure: XML-based message can be converted as a certain required format using XML transformation pipeline



Service-Oriented Architecture (3)

- Connecting to trading partners and legacy systems as easy as connecting to an interdepartmental system
- Easily joining and leaving the system for new participants
- Dynamic deployment



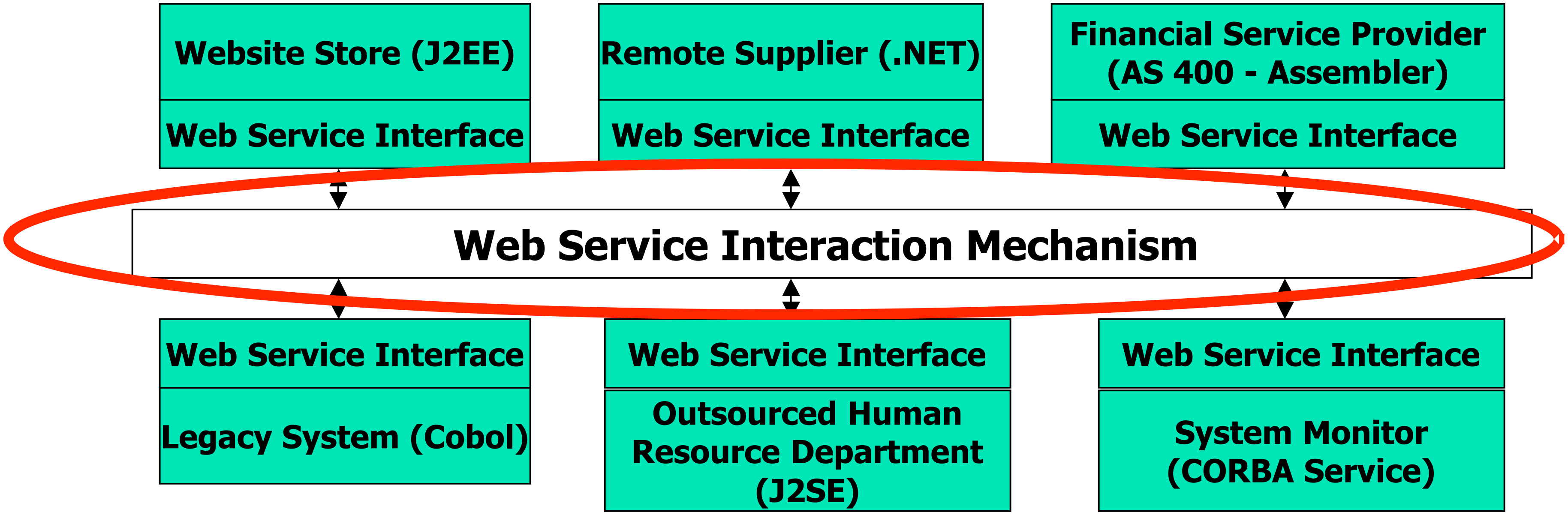
Each of the subsystem is built using a different technology for their implementation

Service-Oriented Architecture (4)

- Transforming a legacy system into an XML service
- Same principle for other application (subsystem) of latest technology (e.g J2EE, Microsoft .NET, etc.)
- Making them joining to SOA
- KEY: interconnecting these services with a MOM-based communication, because MOM promotes loose coupling, flexibility, reliability, scalability and high-performance characteristic (as we discussed before)

XML + Web Service + MOM + SOA = Open System

Web Service Interoperability(1)



Website Store (J2EE)

Web Service Interface

Remote Supplier (.NET)

Web Service Interface

Financial Service Provider (AS 400 - Assembler)

Web Service Interface

Web Service Interaction Mechanism

Web Service Interface

Legacy System (Cobol)

Web Service Interface

Outsourced Human Resource Department (J2SE)

Web Service Interface

System Monitor (CORBA Service)

Web Service Interoperability(2)

- Interoperability is main feature of Web Service due to its open standard (xml, soap)

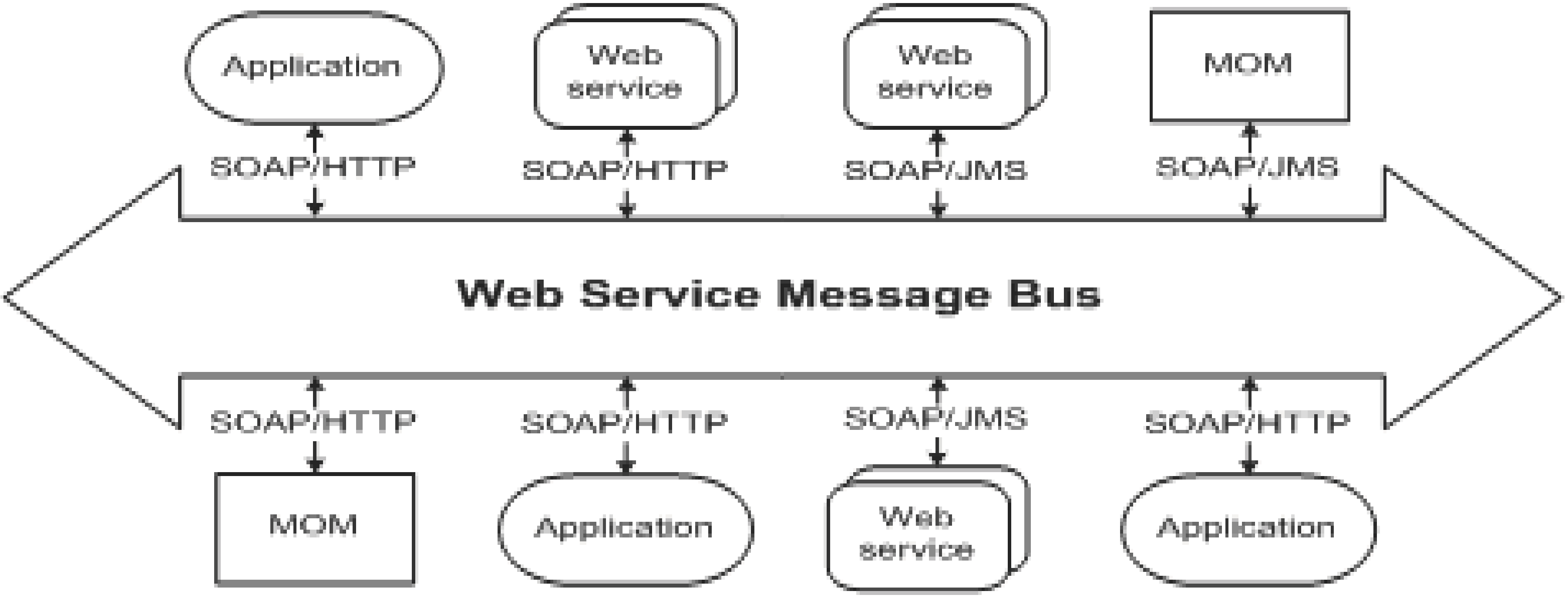
- Potential mismatches that hinders Web Service Interoperability
  - Data mismatch (data structure and semantic)
  - Protocols mismatch (the interaction application might use different protocols)
  - Semantic mismatch (the interaction parties might interpret the same information in different way)

Introducing a mediation layer named Web Service Message Bus (WSMB) to bridge differences in communication protocols, data format and reconciliation of compatibilities in business protocols and process (future)

WSMB – A highly interoperable middleware for Web Service

- An important architectural component for SOA system
- A service intermediary
- Features
  - Multi-protocol services to bridge interfaces and protocol difference
  - Protecting applications from emerging and rapidly evolving Web services standards
  - Protect servers from overload by queuing or redirecting messages using WSMB management console

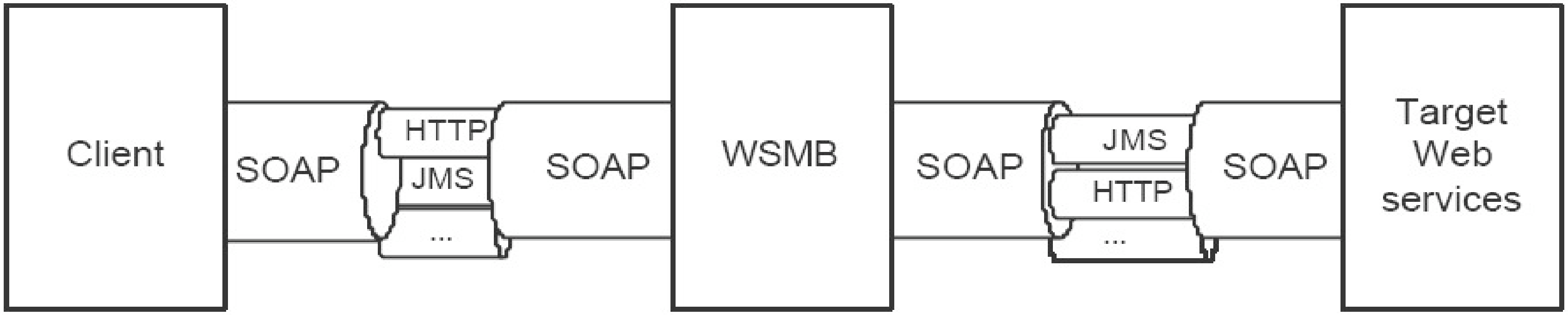
WSMB Architecture



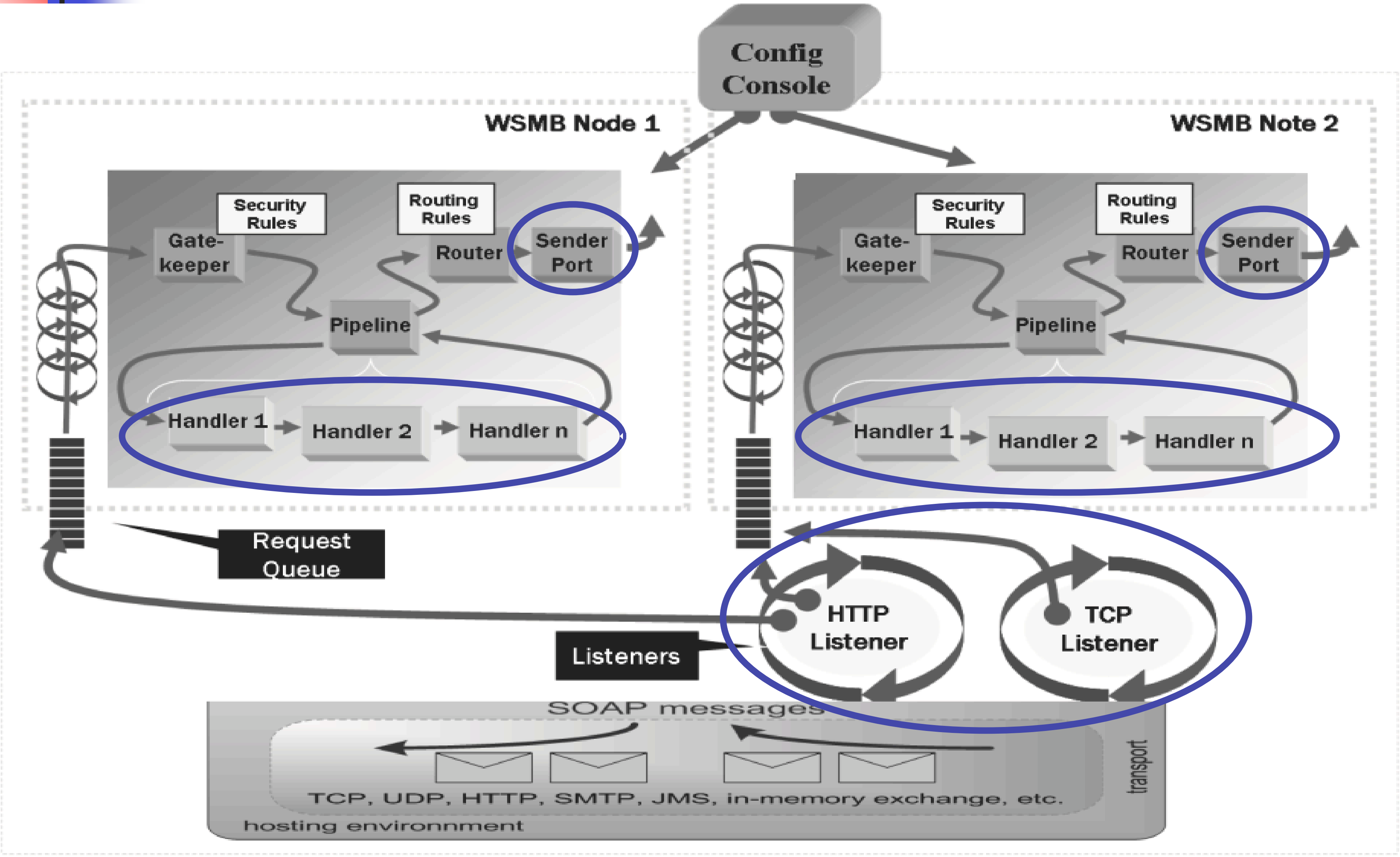
- Implemented by using MOM concept
- Bus architecture, analogous to computer hardware bus system
- Emphasizing on the integration capability

WSMB Communication Layer Interoperability

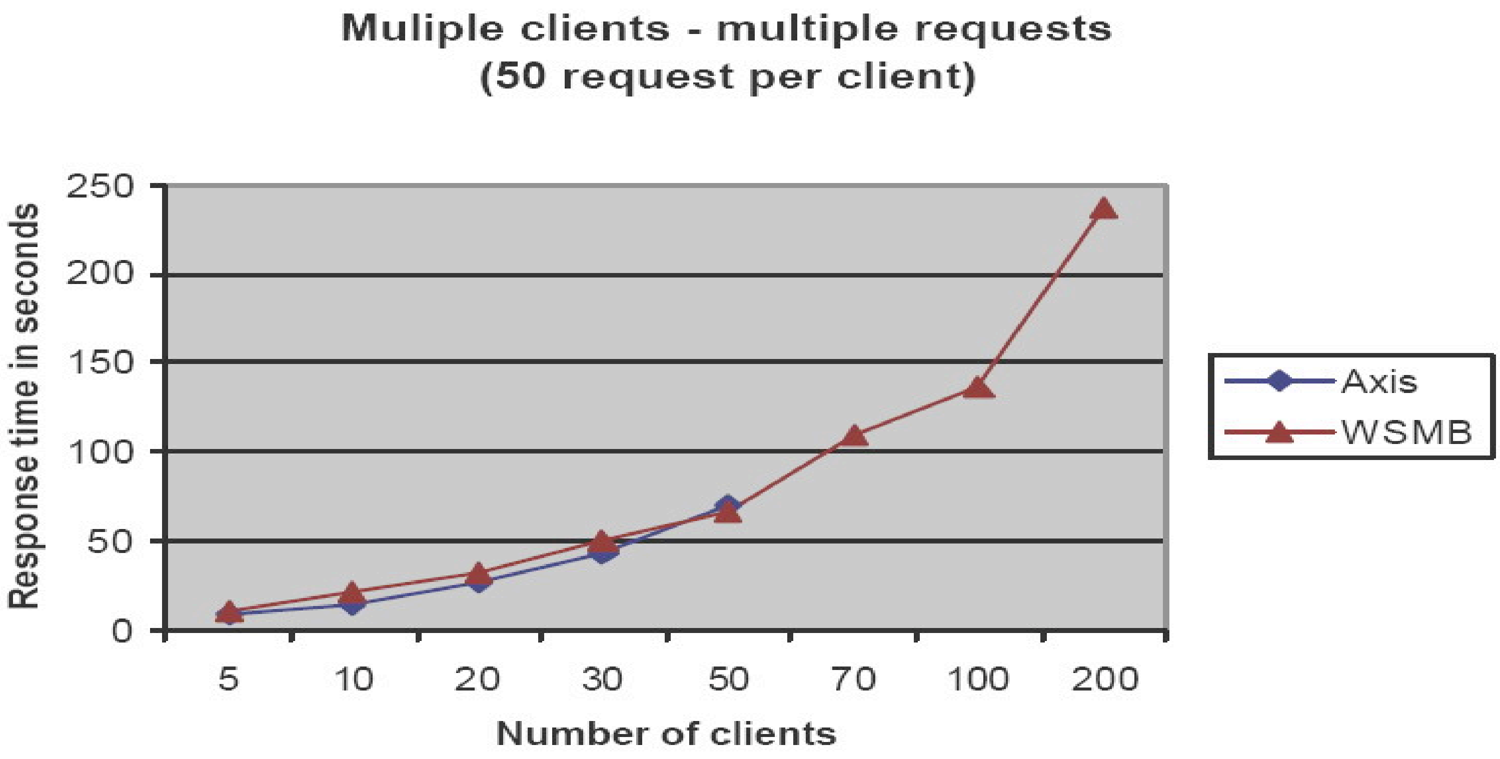
- Providing various channels to access the registered Web Service
- Messages getting queued for processing after being checked
- Dispatching messages to the destination Web Service and the response is passed back to the requester via the same path
- Effectively switching between transports protocols



WSMB Content Layer Interoperability

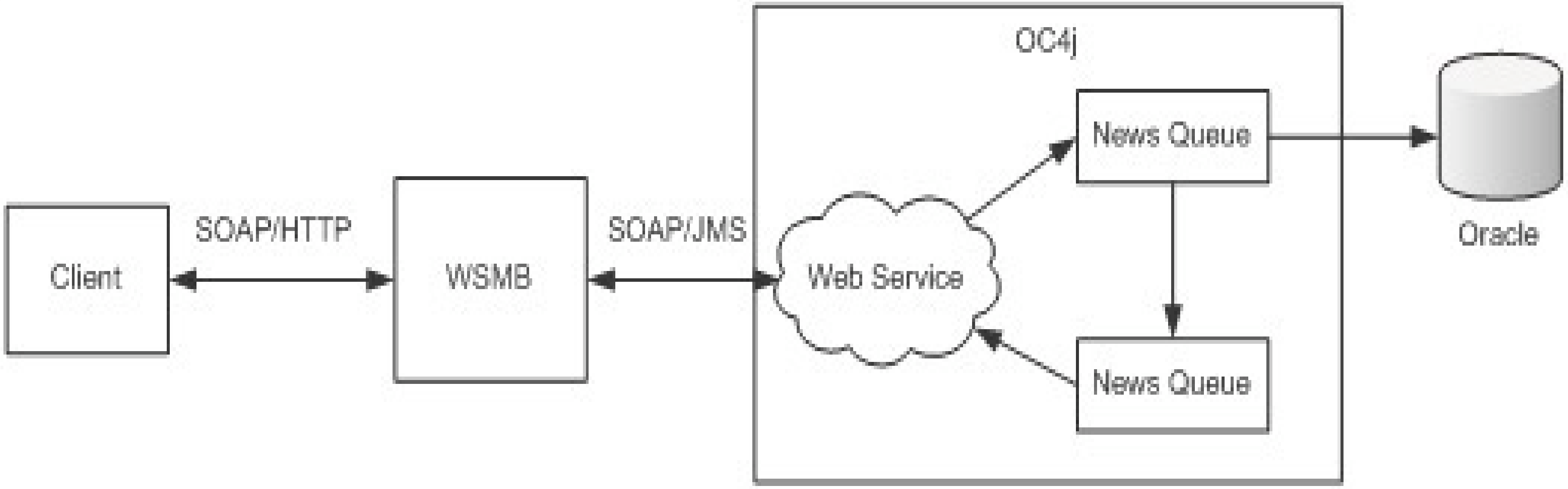


Performance Evaluation



Message through WSMB is more resilient and response time is more predictable

Interoperability Evaluation



Convey SOAP-over-HTTP messages to the target Web Service of SOAP-over-JMS

Conclusion

- MOM is a revolutionary concept in distribution allowing for communications between disparate software entities to be encapsulated into messages.
- MOM-based systems are proficient in coping with traffic bursts while offering a flexible and robust solution for disperse deployments.
- SOA using web service is the state-of-the art approach to support interoperability between distributed systems.
- A highly flexible Service-Oriented Architecture can be created when MOM is used in conjunction with XML messages and Web Services.
- WSMB, a middleware based on MOM concept, can enhance the interoperability of Web Service.

Reference

1. Qusay H. Mahmoud *Middleware for communications,* John Wiley & Sons, Ltd. ISBN: 0-470-86206-8
2. Trung Nguyen Kien, Abdelkarim Erradi and Piyush Maheshwari, *WSMB: a middleware for enhanced Web services interoperability*, School of Computer Science and Engineering, The University of New South Wales, Australia
3. Tanenbaum, A. S. and Steen, M. V. (2002) *Distributed Systems: Principles and Paradigms*, 1st ed., Prentice Hall, New York, USA
4. Banavar, G., Chandra, T., Strom, R. E. and Sturman, D. C. (1999) A Case for Message Oriented Middleware. *Proceedings of the 13th International Symposium on Distributed Computing*, Bratislava, Slovak Republic.