Challenges in Middleware Design in Distributed Systems
A Comparative study: Web Services, Java RMI, CORBA and DCOM

Prof. (Ms) Manisha Shinde
Faculty of Management (Information Technology), Bharati Vidyapeeth Univerisity, Pune,
IMRDA, SANGLI
Mobile No. 9011087009 Landline No. 02346-234500
mjs.imrda@gmail.com , alisha11sept@gmail.com

Abstract: Collaborative and remote applications provide sharing of distributed resources and services on demand. Middleware enables interoperability between applications that run on different operating systems, by supplying services so the application can exchange data in a standards-based way. But such system mostly fails to support the basic requirement robustness, reliability for every real time VM Communication. One of the major use of middleware is to be always available with reliability, robust and that to independent from network services. Having future internet world aspect, these middleware's should be designed by considering challenges of manipulation, connections, object management, network management and should have aspect of reasonable response time only. By comparing, analyzing different infrastructure needs of distributed objects challenges can be studied and middleware design can be improved under systematic, logical and technical approach. This will help the distributed API designers and middleware designers to smoothen out the complexity of heterogeneity.
Keywords: *Middleware, Interoperability, Object Management, Network Management, API, VM.*

1. Introduction:

Distributed middleware such as Web services, RMI, CORBA and DCOM is integrated together independent of network services have helped to convert the applications in web based application for similar or disparate application/platform communication or to function together. The functionality and reliability of such applications communication depends on number of different parameters involved in the processing. Different objects while managing on different platforms, configurations will be considered along with their Data Model, object states, types, communication model, protocol, coupling behavior and garbage collection techniques.

A distributed system is one that looks to its users like an ordinary, centralized, system but runs on multiple independent CPUs. The computers interact with each other in order to achieve a common goal. A computer program that runs in a distributed system is called a **distributed program**, and **distributed programming** is the process of writing such programs.

- Multiple, independent processing units
- Processors communicate via a hardware interconnect
- Processing unit failures are independent
- Manage resource sharing
- State is shared among processors

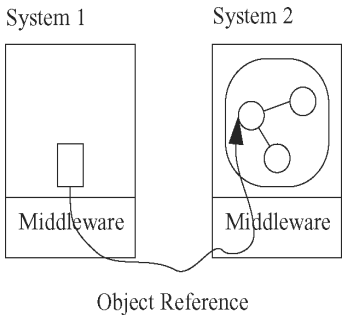
Middleware in the context of distributed applications is software that provides services beyond those provided by the operating system to enable the various components of a distributed system to communicate and manage data. Middleware supports and simplifies complex distributed applications. It includes web servers, application servers, messaging and similar tools that support application development and delivery. Middleware is especially integral to modern information technology based on XML, SOAP, Web services, and service-oriented architecture. It enables interoperability between applications that run on different operating systems, by supplying services so the application can exchange data in a standards-based way.

One of the major use of middleware is to be always available with reliability and that to independent from network services. Middleware can help software developers avoid having to write application programming interfaces (API) for every control program, by serving as an independent programming interface for their applications. For Future Internet network operation through traffic monitoring in multi-domain scenarios, using mediator tools (middleware) is a powerful help since they allow operators, searchers and service providers to supervise Quality of service and analyze eventual failures in telecommunication services

2. Methodology:

The researcher would like to explore, if by applying, comparing and analyzing different functional and technical parameters for distributed applications like DCOM, RMI, CORBA, Web Services as shown in **Table No. 1** to discover, identify, develop and to design the middleware components. Based on certainty of processing, challenges of design can be smoothen by having need based middleware's either standalone, integrated, modular or domain specific.

Objects Reference in Networks



- The network address.
- The port number (transport protocol).
- Object (agent) name or ID.

Distributed Object Systems involves

- Data Object Model
- Developing distributed applications
- Locating remote objects on a network
- Sending messages to those objects.
- Common interface for transactions, security, etc.

Comparative analysis along with business model to implement will generate basic key functionality requirement, which will be focused in middleware design which will smoothen out deferred from function resources either to prune out or to modularise for other needs/requirements. This will ultimately lead in decrease in overhead on resources and can improve response time.

Tabel No. 1 Current distributed infrastructure

FEATURE	DCOM/COM+	Java RMI	CORBA	WEB SERVICES
Data Model	Component Object Model	mathematical data model	Object Model	SOAP Message exchange model
Client Server Coupling	Tight Coupling & Loose Coupling	Tight Coupling	Tight Coupling	Loose Coupling
Protocol	Object Remote Procedure Call	Java Remote Method Protocol	Internet Inter-ORB Protocol	Simple Object Access Protocol
Parameter Passing	Pass by reference /value	Pass by reference /value	Pass by reference / value	Pass by value
Type Checking	Static/Runtime	Static/Runtime	Static/Runtime	Runtime Only
State	Stateless, Stateful	Stateful	Stateful	Stateless, Stateful
Firewall Traversal	Uses HTTP port No. 80	Work in progress	Work in progress	Uses HTTP port No. 80
Service Discovery	Service Control Manager (SCM)	RMI Registry provider	CORBA naming/trading Service	UDDI WSIL
Communication Model	1-way	1-way	1-way, 2-way sync 2-way async	2-way sync (Web Services) 1-way, 2-way sync, 2-way async(Web Process)
Object Implementation	The responsibility of locating an object implementation falls on the Service Control Manager (SCM)	The responsibility of locating an object implementation falls on the Java Virtual Machine (JVM)	The responsibility of locating an object implementation falls on the Object Request Broker (ORB)	The responsibility of locating an object implementation falls on the Object Request Broker (UDDI)
Garbage collection	distributed garbage collection on the wire by pinging	Attempts to perform distributed garbage collection of remote server objects using the mechanisms bundled in the JVM	Does not attempt to perform general-purpose distributed garbage collection**.**	Distributed garbage collection on server.
Specification	COM Specification	Java Remote Interface	IDL	WSDL

Functions that are performed as part of network management accordingly include controlling, planning, allocating, deploying, coordinating, and monitoring the resources of a network, network planning, frequency allocation, predetermined traffic routing to support load balancing, cryptographic key distribution authorization, configuration management, fault management, security management, performance management, bandwidth management, Route analytics and accounting management.

Now along with focusing application and its development strategy in a web based application, the researcher can focus middleware design and function to perform the successful implementation. Developers or analysts can focus on Virtual Machine Communication logic than set of instructions and entities. As shown in above Table No. 1 the researcher can combine the need of processing for particular application or for combined application behavior. This decision can further help in designing algorithms, establishing relationships among different middleware components.

Such Middleware infrastructure will include well predicated combination of set of agents, resources, policies, services and components of communication in distributed Virtual Machines.

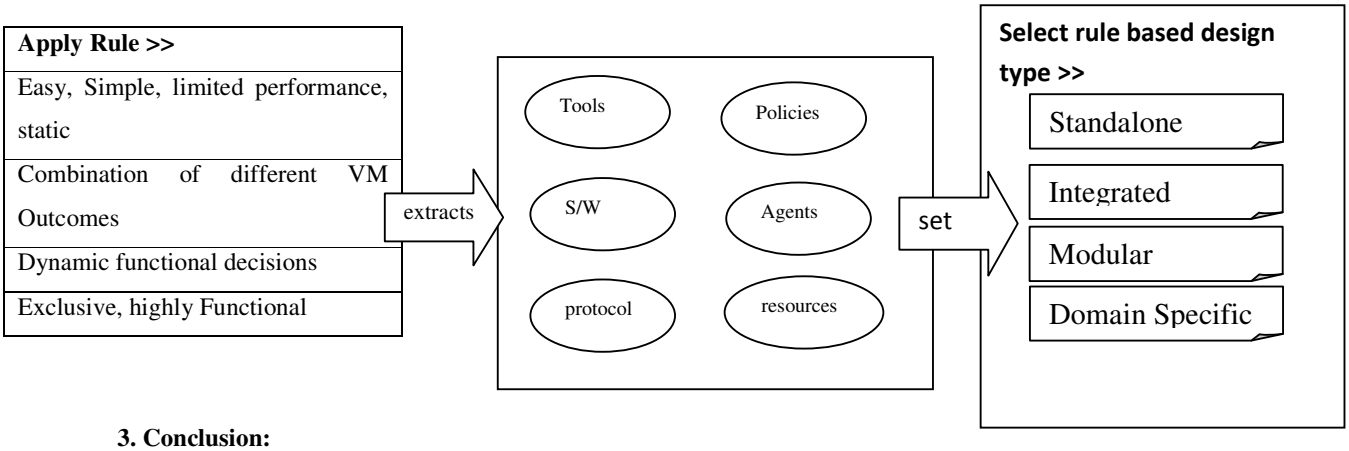
Domain based middleware or Domain layered middleware can be another solution to increase the reliability and robustness of performance. Modular middleware infrastructure also can help to specify services design per module on dedicated base and to artifact it as per demand. As shown in Figure No. 1 by applying different logical rules to extract proper set of components, middleware design strategy and policies can be decided as per demand or need of communication and performance.

Middleware Design Types

1. Standalone : For easy, simple, limited small and/or static performance
2. Integrated : For Combining logical communication of two/ more VMs, Enterprise Wide Middleware
3. Modular : For dynamic functional decisions based on remote requests/ parameters
4. Domain Specific: For Specific, highly exclusive and functional performance oriented

Technically while selecting hardware, software, design policies, if the above logic is applied complexity of design will be smooth as well as it will help System developers to develop more compatible APIs.

Figure No.1. **Categorized Middleware Infrastructure and set of Components.**



3. Conclusion:

The mature, well-defined middleware infrastructure design can help to reduce efforts on software development life cycle. Comparatively robust and reliable communication among distributed systems will be achieved.

Although the basic requirements of distributed communication are being fulfilled based on current built of middleware, there are so many problems and challenges in such communication for large scale systems. In future it will need more precise and proper technical mechanism to built, control(regulate), analyse either software or hardware middleware infrastructure. In future work documentation, database/record managements about middleware can be built in aspect to improve its functional performance.

References:

1. "Java 6 " Black Book, Kogent Solution Inc.
2. "Object Oriented Programming Through JAVA" V. Vijaya Bhaskar, P. Venkata Subba Reddy SCITECH.
3. http://web.info.uvt.ro/~petcu/distrib/TDS1.pdf Distributed Systems –Technology
4. http://my.execpc.com/~gopalan/misc/compare.html
5. http://en.wikipedia.org/wiki/Network_management
6. http://en.wikipedia.org/wiki/Middleware_%28distributed_applications%29#Use_of_middleware