U. S. Railroad Retirement Board

Middleware Domain Architecture

Middleware Domain Architecture
Table of Contents

Middleware Domain Definition

Middleware is software that supports communications between the functional tiers of an application, between two or more different applications, and between application and shared services. The role of middleware is to insulate application developers from having to understand the complexities of the networking and computing environments and to prevent them from having direct access to platform, network, and data layers. Middleware also provides an environment in which to implement business rules (logic) and workflow rules.

The Middleware Domain defines the principles, technologies, standards and guidelines for the use of middleware at the RRB.

Middleware Domain Principles Summary

1. Use middleware to reduce integration complexity.
2. Middleware technologies must be extensible and scalable.
3. Keep current with middleware technologies and their applicability to enterprise architecture.
4. Middleware applications and software will be purchased and implemented in adherence with the agency's security, confidentiality and privacy policies. The middleware security interfaces and procedures will be consistent with agency standards.
5. Use industry standard middleware products.
6. Use middleware for cross-platform communication between application components.
7. Use asynchronous messaging whenever appropriate.
8. Minimize the impact to existing applications when designing middleware solutions.

Middleware Domain Relevant Trends

- There is an increasing demand to make our systems available to the public.
- There is an increasing need for writing reusable code.
- It is becoming more important to take advantage of our investment in existing systems.
- There is a growing realization of the benefits of accessing data as close to the source as possible.
- More tools are available to make it easier to access/transmit data.
- There is a trend to build systems with N-Tier architecture.

Background of Middleware and Related Technologies at the RRB

As the Railroad Retirement Board moves to a more distributed platform infrastructure, middleware will play an increasingly important role in logically presenting a single-system image and in simplifying application integration. As our physical environment of Internet/Intranet servers, application, transaction and database servers becomes more complex, the use of middleware will reduce much of this complexity and the effort involved in delivering applications into this environment.

Middleware products are an evolving technology encompassing a wide range of capabilities from database access to very sophisticated integration engines known as message brokers. Up to now, the Railroad Retirement Board's use of middleware has been limited primarily to screen scraping. The growth of Intra/Internet applications is increasing the need for more sophisticated middleware tools and we are beginning to use and/or explore the use of tools such as database gateways and message-oriented middleware.

By delivering a common, standard solution to cross-platform communication and related services, middleware products offer the following benefits:

- **Adaptability.** A very important aspect of middleware is that by providing a common service through a standard application interface, applications are freed from a specific infrastructure. For example, applications requiring relational data can access that data from a Microsoft SQL database or a DB2 database just by changing the target database management system, not the application. Applications and technology can evolve and adapt to changes with minimal or no impact on each other.
- **Flexibility.** The features and capabilities of applications can be modified without changes to the technical architecture. For the same reasons that the infrastructure can change without impacting the application, the application is free to change without necessarily impacting the infrastructure.
- **Reduced development effort.** The use of middleware products simplifies the development of N-tier applications by providing common services for database access, transaction management and application-to-application communication.
- **Reduced integration effort.** Standardizing and expanding the capabilities offered through middleware products will allow purchased applications and services to be more easily integrated. A growing population of application packages is adopting industry middleware standards allowing these products to inter-operate.

Middleware products fall into several categories depending on the services provided. Each of these categories forms a loosely associated hierarchy based on the level of services required. It is very likely that multiple middleware products will be used at the Railroad Retirement Board to deliver the complement of services needed.

Detailed Middleware Domain Principles

Domain Principle 1

Use middleware to reduce integration complexity.

Rationale:

Reduces time and cost of developing and supporting system enhancements.
Reduces the need to develop custom solutions to make applications interoperable.
Reduces on-going support costs.
Provides easier information access that promotes sharing enterprise-wide resources among

- Employees
- Business partners
- Agency customers.

Reduces risks associated with system implementation and upgrades.
Decreases the number of ways an application system is put together.
Supports and controls our ability to interface with diverse products, applications, and databases.

Implications:

Requires the direct use of data storage systems as data sources.
Will eventually reduce the learning curve for implementing applications.
Inappropriate use of middleware can increase application complexity.
Will require greater collaboration among project teams and team members to identify/develop components accessed by middleware.
Application development teams will require middleware representatives.
Need more knowledge about middleware to determine what middleware we need.
Organizational changes may be needed to implement middleware most effectively.
May suboptimize development time of specific applications.
Need to develop a metadata repository of distributed objects and their interfaces to allow for inter-application communication.

Domain Principle 2

Middleware technologies must be extensible and scalable.

Rationale:

Reduces the cost of integrating new technology and functionality.
Facilitates changes in business to business communication and data exchange.
Supports incremental deployment of existing business applications.
Supports our customers' increasing expectations of more service choices.

Implications:

May need to spend more for software that exceeds immediate needs to be able to meet future business requirements. For example, use of Transaction Processing Monitors may be necessary if numbers of transactions in an application exceed 100's per second.
Requires long-term planning to identify future business needs.

Domain Principle 3

Keep current with middleware technologies and their applicability to enterprise architecture.

Rationale:

Enables the selection of more effective middleware solutions.
Fosters timely reaction to changes in Network, Platform and Data Domains that have an impact on middleware technologies.

Implications:

Need to assign organizational responsibility for middleware research and development.
Requires the development of a process and structure to track or identify changes in middleware technologies.
Need to dedicate resources to keep current with developing middleware choices.
Requires an environment to evaluate the new tools and how they work with the infrastructure.
Requires adequate funding.

Domain Principle 4

Middleware applications and software will be purchased and implemented in adherence with the agency's security, confidentiality and privacy policies. The middleware security interfaces and procedures will be consistent with agency standards.

Rationale:

Help safeguard client information
Protect agency assets

Implications:

Security concerns may require suboptimization of middleware applications.
May increase complexity of integration.
Security concerns may increase the "tuning" needed to install middleware products.

Domain Principle 5

Use industry standard middleware products.

Rationale:

Products with a broad market presence tend to be more responsive to customer requirements.
Product enhancements occur more frequently.
Industry standard products will exist longer and are usually well supported.
User groups for popular products offer peer support.
Third party vendors provide additional support or enhancement options.

Implications:

Customization should be limited to optimize this principle.
Must keep up-to-date on industry standards. (See Principle 3.)
May commit us to one vendor's product.

Domain Principle 6

Use middleware for cross-platform communication between application components.

Rationale:

Enhances the ability to capitalize on and exploit business information.
Supports shorter development times.
Maximizes infrastructure re-use.

Implications:

System development life cycle should include a checkpoint for the effective use of middleware.
Requires determination of the impact of new or revised applications on the middleware interface.
Front-end analysis will increase because of the additional options that middleware will provide.

Domain Principle 7

Use asynchronous messaging whenever appropriate.

Rationale:

Asynchronous communication offers more flexibility than synchronous communication but still allows for synchronization at an appropriate time in the context of the application.
Asynchronous messaging (Message-Oriented Middleware, Object Request Brokers, etc.) allows a subsequent process to decide on an appropriate processing strategy to optimize its throughput and/or respond to different priority requests.
Asynchronous processing increases performance and scalability of the application.
Protects integrity of the data as it is passed from one application to another.

Implications:

Requires establishment of guidelines for when asynchronous or synchronous processing is appropriate.
Existing programs may dictate which type of processing to use.
Will require training for developers to learn how to program asynchronous applications.
May make program development more complex.

Domain Principle 8

Minimize the impact to existing applications when designing middleware solutions.

Rationale:

Using middleware to minimize the impact on legacy systems leverages the large investment we have in our existing systems.

Implications:

Need to consider the cost of changing existing systems to be compatible with middleware applications.

Sometimes replacing the existing legacy application will be the best solution.

Could lead to the development of parallel systems.

Domain Participants

Domain Team Leader: Melissa Mickelberry

Line of Business Representatives: Cathy Leyser, Martha Barringer

Domain Participants: Robert Schempp, Steven Roberts, Annie Williams, Barbara Gleason, Howard Klopp, John Lach

APG Representative: Sally Mui

Appendix 1: Domain Glossary

Term	Definition
Asynchronous	A form of communication by which applications can operate independently, so that they do not have to be running or available simultaneously. A process sends a request and may or may not wait for a response.
Middleware	The sweet nougaty center of the infrastructure. The good stuff in the middle. Think candy bars or jelly donuts or tootsie roll pops.
Message Oriented Middleware (MOM)	A middleware product that relays data from one application to another in the form of messages. This allows for efficient use of resources and is especially useful where asynchronous processing is possible or required. IBM's MQSeries and Microsoft's Message Queue Server (MSMQ) are examples of MOM.
Object Request Broker (ORB)	An architecture and programming model that allows and enables applications to access common components and network services. There are three main ORB architecture models: ®Common Object Request Broker Architecture (CORBA) ®Enterprise JavaBeans (EJB) ®Component Object Model (COM/COM+) ORBs allow applications to request a service without knowing anything about what servers are attached to the network. The ORBs receive the requests, forward them to the appropriate servers, and then hand the results back to the client.
Synchronous	A form of communication that requires applications to run concurrently. A process issues a call and waits until it receives a response.
Transaction Processing Monitor (TP Monitor)	A program that monitors a transaction as it passes from one stage in a process to another. The TP Monitor's purpose is to ensure that the transaction processes completely or, if an error occurs, to take appropriate actions. TP Monitors are especially important in three-tier architectures that employ load balancing because a transaction may be forwarded to any of several servers. In fact, many TP Monitors handle all of the load balancing operations, forwarding transactions to different servers based on their availability. Microsoft's Transaction Server (MTS) is an example of a TP Monitor.

Appendix 2: Conceptual to Domain Principle Matrix

	Relationship Between RRB's Middleware Domain Principles And Conceptual Architecture Principles																							
Domain Principle	***Conceptual Architecture Principles***																							
	CA 1	CA 2	CA 3	CA 4	CA 5	CA 6	CA 7	CA 8	CA 9	CA 10	CA 11	CA 12	CA 13	CA 14	CA 15	CA 16	CA 17	CA 18	CA 19	CA 20	CA 21	CA 22	CA 23	CA 24	CA 25
D-1					X			X		X		X													
D-2						X																			
D-3													X												
D-4																X	X					X			
D-5					X	X														X		X			
D-6									X			X		X											
D-7						X	X		X						X								X		
D-8							X		X					X				X			X				

Conceptual Architecture Guiding Principles:

1. Use guidelines consistent with the Federal Enterprise Architecture. 2. Support a single Enterprise Wide Technical Architecture (EWTA). 3. IT projects are to be consistent with the Enterprise Architecture. 4. IT projects are to be consistent with the Enterprise Architecture. 5. Reduce integration complexity. 6. Technical architecture must be extensible and scalable. 7. Manage information and data as enterprise-wide assets. 8. Validate information as close to its source as possible. 9. Enhance the ability to capitalize on and exploit business information. 10. Support multiple data types. 11. Make an informed buy versus lease versus build decision before proceeding with any new development project. 12. Require shorter development cycle times. 13. Keep current with emerging technologies and their applicability to enterprise architecture. 14. Maximize infrastructure asset reuse. 15. Sustain reliable connectivity. 16. IT systems will be implemented in adherence with the agency's security, confidentiality and privacy policies. 17. The agency will use a consistent set of security interfaces and procedures. 18. Reduce total cost of operation (TCO). 19. Extend E-Mail to Become a Corporate Information Exchange Vehicle. 20. Adopt Open Systems Standards. 21. Reduce duplicate information systems. 22. Reduce duplicate information systems. 23. Maximize and exploit Internet and Intranet technologies and approaches. 24. Integrate Enterprise Architecture into the investment management process. 25. Customer perception is a measure of the quality of the automation processes.