



Compliance Component

DEFINITION

<i>Name</i>	Message Oriented Middleware (MOM)
<i>Description</i>	Message-oriented middleware (MOM) is a client/server infrastructure that increases the interoperability, portability and flexibility of an application by allowing the application to be distributed over multiple platforms. It reduces the complexity of developing applications that span multiple operating systems and network protocols by insulating the application developer from the details of the various operating system and network interfaces. Application Programming Interface (APIs) that extend across platforms and networks are typically provided by the MOM[Rao 95].
<i>Rationale</i>	The state of Missouri needs a mechanism to communicate over multiple platforms.
<i>Benefits</i>	<ul style="list-style-type: none"> • MOM insulates developers from connectivity concerns. The application developers write to APIs that handle the complexity of the specific interfaces. • Message-oriented middleware's use of message queues tends to be more flexible because it allows both synchronous and asynchronous communications. • Message-oriented middleware facilitates data transfer for an application across multiple platforms. • MOM is well-suited for object-oriented systems because it furnishes a conceptual mechanism for peer-to-peer communications between objects.

ASSOCIATED ARCHITECTURE LEVELS

<i>Specify the Domain Name</i>	Interoperability
<i>Specify the Discipline Name</i>	Data Exchange
<i>Specify the Technology Area Name</i>	Data Transfer Protocols/Standards
<i>Specify the Product Component Name</i>	IBM Websphere MQ Series

COMPLIANCE COMPONENT TYPE

<i>Document the Compliance Component Type</i>	Guideline
<i>Component Sub-type</i>	

COMPLIANCE DETAIL

It is recommended that all state agencies that are utilizing message oriented middleware should consider the ramifications listed.

MOM is typically implemented as a proprietary product, which means MOM implementations are nominally incompatible with other MOM implementations.

Features to consider when selecting a MOM product are:

- Message Formats
- Message Transfer Method
- Event Registry
- Transaction Logging
- Intelligent Routing

Using a single implementation of a MOM in a system will most likely result in a dependence on the MOM vendor for maintenance support and future enhancements. This could have a highly negative impact on a system's flexibility, maintainability, portability, and interoperability.

MOM is most appropriate for event-driven applications. When an event occurs, the client application hands off to the messaging middleware application the responsibility of notifying a server that some action needs to be taken.

Asynchronous and synchronous mechanisms each have strengths and weaknesses that should be considered when designing any specific application. The asynchronous mechanism of MOM, unlike Remote Procedure Call (RPC), which uses a synchronous, blocking mechanism, does not guard against overloading a network. As such, a negative aspect of MOM is that a client process can continue to transfer data to a server that is not keeping pace.

The message-oriented middleware software (kernel) must run on every platform of a network that is involved in the application data exchange. The impact of this varies and depends on the characteristics of the system in which the MOM will be used:

- Not all MOM implementations support all operating systems and protocols. The flexibility to choose a MOM implementation may be dependent on the chosen application platform or network protocols supported, or vice versa.
- Local resources and CPU cycles must be used to support the MOM kernels on each platform. The performance impact of the middleware implementation must be considered; this could possibly require the user to acquire greater local resources and processing power.
- The administrative and maintenance burden would increase significantly for a network manager with a large distributed system, especially in a mostly heterogeneous system.
- A MOM implementation may cost more if multiple kernels are required for a heterogeneous system, especially when a system is maintaining kernels for old platforms and new platforms simultaneously.

State the Guideline, Standard or Legislation

Document Source Reference #

[Rao 95] Rao, B.R. "Making the Most of Middleware." *Data Communications International* 24, 12 (September 1995): 89-96.

Compliance Sources

<i>Name</i>		<i>Website</i>	
<i>Contact Information</i>			

<i>Name</i>		<i>Website</i>	
<i>Contact Information</i>			
KEYWORDS			
<i>List Keywords</i>			
COMPONENT CLASSIFICATION			
<i>Provide the Classification</i>	<input type="checkbox"/> <i>Emerging</i>	<input checked="" type="checkbox"/> <i>Current</i>	<input type="checkbox"/> <i>Twilight</i> <input type="checkbox"/> <i>Sunset</i>
<i>Sunset Date</i>			
COMPONENT SUB-CLASSIFICATION			
Sub-Classification	Date	Additional Sub-Classification Information	
<input type="checkbox"/> <i>Technology Watch</i>			
<input type="checkbox"/> <i>Variance</i>			
<input type="checkbox"/> <i>Conditional Use</i>			
Rationale for Component Classification			
<i>Document the Rationale for Component Classification</i>			
Migration Strategy			
<i>Document the Migration Strategy</i>			
Impact Position Statement			
<i>Document the Position Statement on Impact</i>			
CURRENT STATUS			
<i>Provide the Current Status</i>	<input type="checkbox"/> <i>In Development</i>	<input type="checkbox"/> <i>Under Review</i>	<input checked="" type="checkbox"/> <i>Approved</i> <input type="checkbox"/> <i>Rejected</i>
AUDIT TRAIL			
<i>Creation Date</i>	12/01/04	<i>Date Approved / Rejected</i>	12/22/04
<i>Reason for Rejection</i>			
<i>Last Date Reviewed</i>		<i>Last Date Updated</i>	
<i>Reason for Update</i>			