

Microsoft Message Queue (MSMQ) in the Defense Information Infrastructure Common Operating Environment (DII COE)

Part I: Introduction and Architecture

Prepared by Program Manager, Global Combat Support System – Army (GCSS-Army)

Abstract: *This paper is the first of four papers that describes Microsoft Message Queue (MSMQ) and its use in the DII COE. MSMQ is a middleware technology that permits developers to add a communication infrastructure for distributed applications and solve problems caused by network communications unavailability and disparate systems. This paper defines MSMQ and explores its architecture. It will then show how MSMQ fits into the DII COE Architecture. Knowledge of the DII COE, the Component Object Model (COM) and Distributed COM (DCOM) is assumed.*

The term “message queue”, in a sense, is an unfortunate term. The first thing that most people think of when they here the term for the first time is a new type of IN box for their messages or a new protocol for messaging. The term is rarely understood in its correct definition or context by persons not familiar with message queuing. This paper will describe Microsoft Message Queue (MSMQ), its architecture and how its architecture fits into the DII COE architecture.

What is Microsoft Message Queue?

According to Microsoft, “MSMQ is a development tool that includes a store and forward protocol model to be used for developing messaging applications.”¹ Unlike many other Microsoft applications, MSMQ is not an install and run application. MSMQ is part of Microsoft’s Distributed interNet Application (DNA) methodology and is designed to be used as part of a developed distributed application. Specifically, MSMQ is designed to help developers add a communications infrastructure to distributed applications. MSMQ also helps solve two problems:

- Unavailability of communications with servers or clients, and

- Communications between disparate systems and components.

The “messages” in message queuing are application dependent data, vice messages in the sense of e-mail systems. Message queuing maintains a similarity to e-mail systems in that it can store data in a server location and forward that data once the appropriate client is available. In this way, message queues solve the problem of communication unavailability between servers and clients in distributed applications.

MSMQ is part of Microsoft’s Distributed interNet Application (DNA) methodology, which use Component Object Model (COM) and Distributed COM (DCOM) for communications. If communication is required with a platform that does not use COM or DCOM, the data in the message can be formatted in a manner that the Microsoft and other platform can understand. By doing so, developers can solve the problem of communications between disparate systems.

MSMQ server components can be installed as part of Windows NT 4.0 Server, Windows NT 4.0 Server, Enterprise Edition, Windows 2000 Server, Windows 2000 Advanced Server and Windows 2000 Datacenter Server. In Windows NT 4.0, MSMQ is supplied either as part of the NT Option Pack and/or as a component of Windows NT 4.0 Server, Enterprise Edition. Only Windows NT 4.0 Server, Enterprise Edition delivers the full product. The NT Option Pack version, which installs on

¹ *INFO: MSMQ Is Not an E-mail Product Competing w/ Exchange Server*, Microsoft Developer Network (MSDN) Library, April 1999

Windows 4.0 Server, does not include routing servers. In Windows 2000, all Server products contain the full product.

MSMQ client components can be installed on Windows NT 4.0 Workstation, Windows 95, Windows 98 or Windows 2000 Professional. The clients are not supported on Windows 3.1 or Windows for Workgroups 3.11.

Message Queue Services

MSMQ provides several services to an application, including connectionless messaging, guaranteed once only delivery, prioritization and routing, message and system security, and disparate system integration.²

- Prioritization and routing services are provided by a rule based system. It is possible to program business rules for MSMQ that will give priority to certain types of messages to be delivered by their urgency. MSMQ also provides a “cost” for each segment of a route to aid in routing a message.
- Message and system security is provided by the security features of the Windows NT operating system as well as using Security Service Provider (SSP) mechanisms on each message. Such mechanisms include certificate based authentication and encryption.
- Disparate system integration is provided by tools in the MSMQ Software Development Kit (SDK).

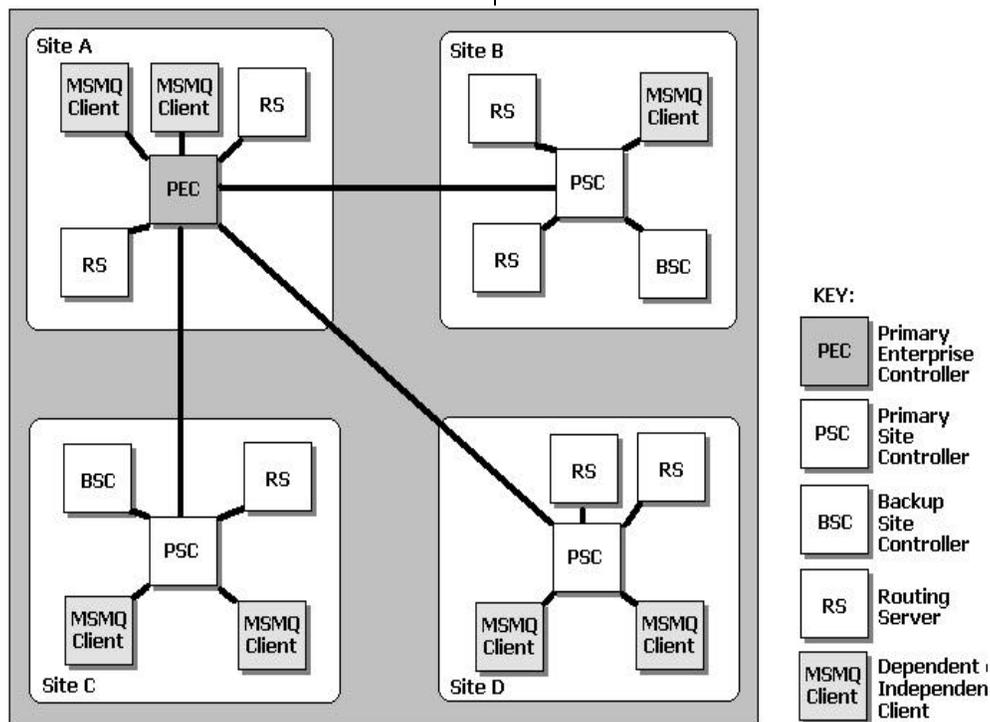


Figure 1

- Connectionless messaging is the store and forward capability of MSMQ.
- Guaranteed once only delivery is accomplished by marking a message as either delivered or undelivered. The system will keep attempting to deliver the message until a predefined error state occurs.

Logical Architecture

The logical architecture of MSMQ includes three basic components: Controllers and servers, clients and messages. Figure 1 is an example of this logical architecture. Controllers and servers store messages, determine transmission paths, control delivery, provide security, administer the system and provides an interface with an application. Clients accept and acknowledge messages, provide security and provide an interface with an

² Homer, Alex and Sussman, David *Professional MTS and MSMQ with VB and ASP* Wrox Press Ltd, 1998

application. Messages package data for transmission. The figure on this page illustrates this architecture.

There are several types of controllers and servers used by MSMQ. Each type can only be installed on a Windows NT 4.0 Server or Windows 2000 Server machine.

- One Primary Enterprise Controller (PEC) is required for each MSMQ network. It is the highest in the hierarchy of MSMQ servers, synchronizes all other servers, and holds the master copy of the Message Queue Information Queue (MSIQ). MSIQ is described in a later section.
- A Primary Site Controller (PSC) is required for each site with an MSMQ network. These controllers hold read only copies of the parts of the enterprise information store that relates to their own site. A site can also have a Backup Site Controller (BSC). A PEC acts as a PSC for the site that it's on.
- Routing Servers (RS) are used to route messages between queues. They do not contain any part of the enterprise information store. PSC and BSC can also perform RS services.

MSMQ clients can be installed on a Windows NT 4.0 Workstation, Windows 2000 Professional, or Windows 9x machine. They require a PEC to be available during installation.

- A Dependent Client uses DCOM to communicate with its site controller and requires the site controller to be available at all times. They have no facilities to store messages or create queues.
- An Independent Client can create

queues and store messages. Independent clients communicate via IP, IPX or RAS. They can also move from one site another.

MSMQ Messages are created by implementing an instance of a **MSMQMessage** object and setting its properties. Once complete, it is sent to a queue, where it is stored and subsequently transmitted. In the case of a dependent client, the message is transmitted immediately, since no queues are available.

Messages are stored in express or recoverable queues. An express queue is implemented in main memory. They provide high performance, but are very volatile. Recoverable queues are implemented on disk. They are slower than express queues, but provide fault tolerance in case of a failure.

Message Queues

Queues can be specified as public or private. Public queues are registered with the PEC and can be located by any MSMQ application. They are persistent and their registration information can be replicated in the enterprise, making them good for long-term use. Private queues are registered on the local computer and typically cannot be located by other applications.

A message queue must be created before messages can be sent. The application can search for and access any public queue. Queues can be assigned permissions, similar to files and directories, to enhance security. The physical location of a queue can also be specified and can reside on any server or client except dependent clients.

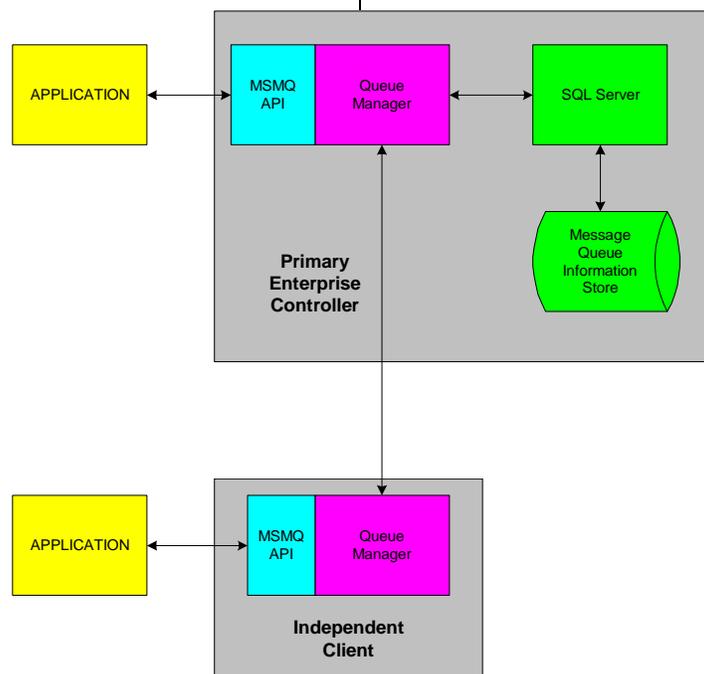


Figure 2

Functional Architecture

The functional architecture of MSMQ is illustrated in Figure 2 using a PEC and an independent client. The center of the architecture is the data store known as the Message Queue Information Store (MQIS).

The MQIS is a database that stores information about the queues and clients in an MSMQ enterprise, such as queue and their locations, site controller locations, computers, etc. The MQIS is kept on the PEC and is replicated as read only to any site controller in the enterprise. The MQIS runs on SQL Server 6.5 in the Windows NT 4.0 Option Pack version. In the Windows NT 4.0 Server, Enterprise Edition, Windows 2000 Server and Windows 2000 Advanced Server versions, the MQIS will not require SQL Server.

A queue manager governs each queue. The queue manager provides all services for the queue, including communications, security, routing, error control and message control. The queue manager works with the MSMQ API to pass data to applications.

The MSMQ API comes in two flavors, depending on the programming language being used. It is available as a set of C++ function calls or as a set of COM or DCOM objects. It's possible to build, send, and receive messages with either one, but the C++ API offers access to a few more services than the COM API. The programming interface for MSMQ will be described in another paper.

The MSMQ API creates a message for transmission, decomposes messages for an application and imposes business rules for MSMQ.

DII COE Architecture³

To use a hardware analogy, the COE is a collection of building blocks that form a software "back plane." Segments "plug" into the COE just as circuit cards plug into a hardware back plane. The

³ This discussion is taken verbatim from : Defense Information Systems Agency (DISA) Joint Interoperability and Engineering Organization (JIEO) *Defense Information Infrastructure (DII) Common Operating Environment (COE) Integration and Runtime Specification (I&RTS) DRAFT Version 4.0*, DISA JIEO, April 1999

blocks containing the operating system and windowing environment are akin to a power supply because they contain the software which "powers" the rest of the system. The segments labeled as COE component segments are equivalent to already-built boards such as Central Processing Unit (CPU) or memory cards. Some of them are required (e.g., CPU) while others are optional (e.g., a specialized communications interface card) depending upon how the system being built will be used. The blocks in the DII COE Architecture diagram labeled as mission application areas are composed of one or more mission-application segments. These segments are equivalent to adding custom circuit cards to the back plane to make the system suitable for one purpose or another.

This hardware analogy can be extended to the Shared Data Environment (SHADE) portion of the COE, but with some significant distinctions. Within this conceptual model, the Database Management System (DBMS) functions as the COE's disk controller and disk drives. The applications' databases can be equated to directories or partitions on the drives accessed through the DBMS "disk controller." Data objects belonging to each database then can be considered as files within those "directories."

The **COE kernel** is the minimal set of software required on every platform regardless of how the platform will be used. The COE kernel components are shown in the DII COE Architecture diagram on the next page and include the Operating System and Windowing Services and a collection of other services that properly belong in the Infrastructure Services Layer. The kernel is intentionally designed to be as small as possible, to be as much COTS as possible, to provide a common starting point for loading segments to build up the system, and to provide an extensible but common runtime environment for segment execution.

Infrastructure Services are largely independent of any particular application. Within the Infrastructure Services layer, Management Services include network, system, and security administration. Communications Services provide facilities for receiving data external to the system and for sending data out of the system. Distributed Computing Services provide the infrastructure

necessary to achieve true distributed processing in a client/server environment. Presentation Services are responsible for direct interaction with the human whether that be through windows, icons, menus, or multimedia. Data Management Services include relational database management as well as file management in a distributed environment. Workflow and Global Data Management Services are oriented towards managing logistics data (e.g., parts inventory, work in process). Note that Data Management Services and Global Data Management Services are part of SHADE.

Unlike Infrastructure Services, **Common Support Applications** tend to be much more specific to a particular mission domain. The Alerts Service is responsible for routing, prioritizing, and managing alert messages throughout the system. The Correlation Service is responsible for maintaining a consistent view of the battle space by correlating information from sensors or other sources that indicate the disposition of platforms of interest. MCG&I Services handle display of National Imagery and Mapping Agency (NIMA) maps or

other products, and imagery received from various sources. Message Processing Services handle parsing and distribution of military-format messages. Office Automation Services handle word processing, spreadsheet, briefing support, electronic mail, World-Wide-Web browsers, and other related functions. (Browsers are in the Common Support Applications layer, but Web Servers fall within the Infrastructure Services layer.) Logistics Analysis contains common functions, such as Pert charts, for analyzing and displaying logistics-related information. Online Help Services provide applications with a uniform technique for displaying context-sensitive help. Finally, Data Access Services are part of SHADE and provide applications with common data access methods, procedures, and tools.

The **Shared Data Environment (SHADE)** is both a strategy for data sharing and the mechanisms to achieve it. SHADE is an integral part of the DII COE, but it must also bridge the gap between COE-based systems and legacy non-COE systems because it must provide mechanisms

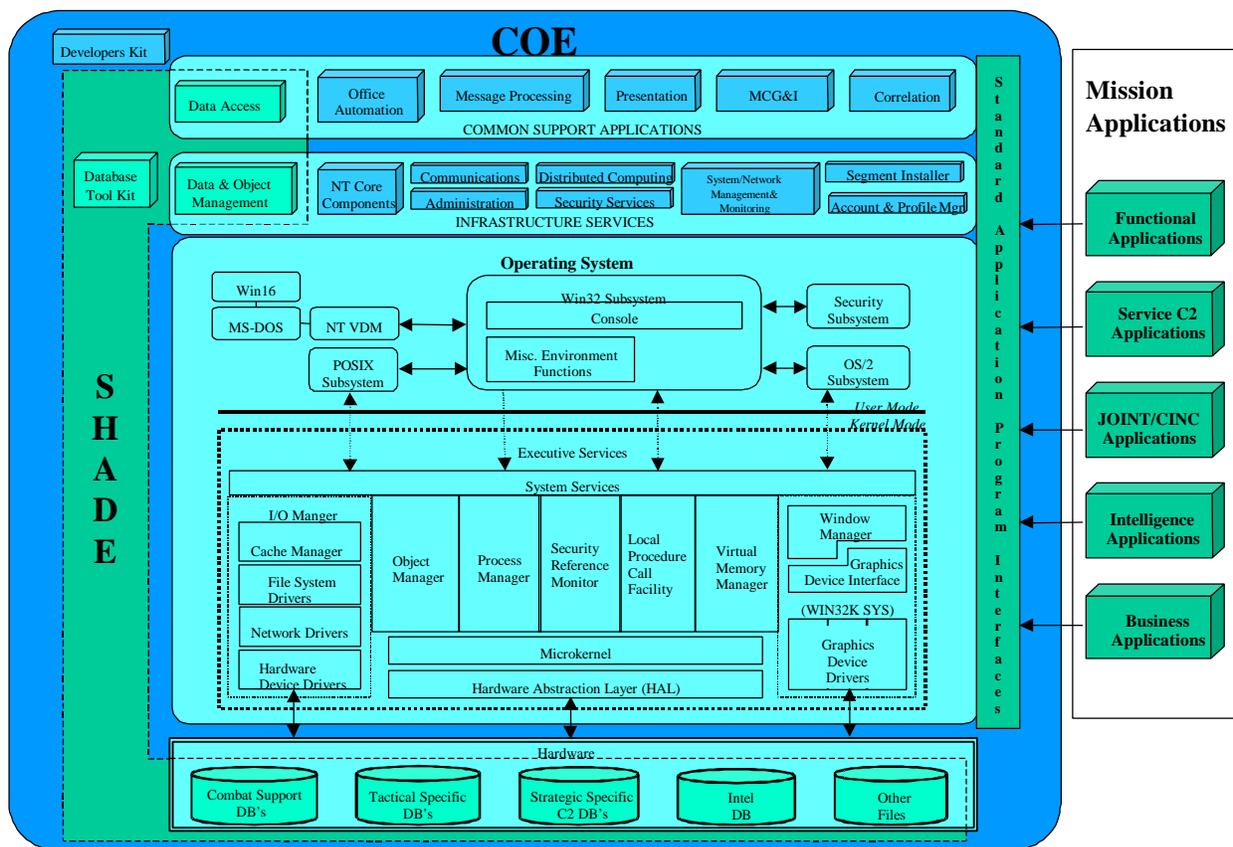


Figure 3

for accessing large databases that are still on legacy mainframes. SHADE provides COE-component segments in both the Infrastructure Services and Common Support Applications layers to accomplish this task. SHADE includes the required data-access architectures, data sharing methodology, reusable software and data components, and guidelines and standards for the development and migration of systems that meet the user's requirements for timely, accurate, and reliable data.

Windows NT COE Architecture

UNIX and NT architectures are similar in many ways, but there are also many fundamental differences. A goal of the COE is to capitalize on the similarity and to negate or minimize the impact of the differences.

Figure 3 is a simplified diagram that illustrates the relationships between the Windows NT version 4 internal components and the COE layers. The labeled boxes in the Infrastructure Services and Common Support Applications layers are not intended to be all encompassing nor are they intended to conflict with the COE architecture described above. The boxes are representative of the services and COE-component segments in the COE and are designed to illustrate the COE platform architecture, as defined above, from the view of an NT COE based platform. The Infrastructure Services, Common Support Applications, and mission-application segments may reside on one platform or may be distributed on separate servers and workstations.

MSMQ in the DII COE

All versions of Windows NT 4.0 and NT Option Pack are presently segmented into the DII COE. All Service Packs up to Service Pack 3 are segmented and Service Pack 4 is expected to be segmented into DII COE Version 4.1, due October 1999. Once Windows 2000 ships, it is expected to be segmented into the DII COE as soon as possible. There is no need to segment MSMQ, since it is an integral part of the Windows NT products.

The main difference between the Infrastructure Services and the Common Support Applications of the DII COE is that the Infrastructure Services deal with data and that the Common Support

Applications deal with information. For the purposes of this discussion, data is the raw unit that the DII COE uses and information is data that has been formatted so that it is useful and intelligible for human consumption.

As the discussion of MSMQ capabilities and architecture shows, MSMQ is intended to present data to applications. This data may be generated by applications or it may be stored in a data store of a database. MSMQ is not intended to present data for human consumption, and by the above definition, does not present information.

MSMQ supports the Infrastructure Services of the DII COE Architecture. It supports different services based upon how it is programmed and the application it's used for. MSMQ can support the Web Services and Distributed Computing services through its COM and DCOM programming interfaces. It can also support Distributed Computing Services through its C++ programming interface. It can use the Transport Layer of the OSI model to provide communications, thereby supporting the Communications Services, but it is difficult to do and not recommended.

MSMQ messages can be formatted to be intelligible to disparate data stores and data engines. Some of the data included in a message can be control commands for data engines. By permitting this level of access and interpretation, MSMQ provides an interface to the SHADE Global Data Management and Data Management Services.

MSMQ uses the security features of Windows NT to provide system security. Security can be provided for individual messages by using certificate based encryption and authentication. These features are supported by the Security Management Services of the Kernel. MSMQ is a user of these services and not a provider.

Conclusion

This discussion intended to be used as an introduction to MSMQ and its use in the DII COE. More detailed discussions of programming interfaces, design considerations, installation and configuration, and security will be presented in other papers of this series. Several references provide the detailed information required to develop MSMQ applications.

Professional MTS and MSMQ with VB and ASP discusses Microsoft Transaction Server and Microsoft Message Queue and their programming interfaces. It can easily be use as a Programmer's Reference Guide to COM, DCOM and C++ objects used in programming MTS and MSMQ.

The Microsoft Developer Network (MSDN) Library is a quarterly set of CD's that contain a host of detailed technical information about Microsoft products. It has been used extensively for preparation of this paper and others in the series.

Bibliography

INFO: MSMQ Is Not an E-mail Product Competing w/Exchange Server, Microsoft Developer Network (MSDN) Library, April 1999.

Chappell, David *Microsoft Message Queue Is a Fast, Efficient Choice for Your Distributed Application*, Microsoft Developer Network (MSDN) Library, April 1999.

Defense Information Systems Agency (DISA) Joint Interoperability and Engineering Organization (JIEO) Defense Information Infrastructure (DII) Common Operating Environment (COE) Integration and Runtime Specification (I&RTS) DRAFT Version 4.0, DISA JIEO, April 1999.

Homer, Alex and Sussman, David *Professional MTS and MSMQ with VB and ASP* Wrox Press Ltd, 1998.

This paper was written by Bob (Fridge) Frees of J. G. Van Dyke and Associates, Inc. (A subsidiary of Wang Federal) under contract to the Program Manager, Global Combat Support System – Army.

Microsoft is a registered trademark of Microsoft, Inc. Microsoft Message Queue, MSMQ, Windows NT, Windows 2000 Professional, Windows 2000 Server, Windows 2000 Advanced Server, Windows NT 4.0, Enterprise Edition, Windows NT Option Pack, Component Object Model, COM, Distributed Component Object Model, DCOM, and Distributed interNet Applications are copyright Microsoft, Inc.