AFRL-VS-PS-TR-1998-1071

AFRL-VS-PS-
TR-1998-1071

CONTROLLER AND COMMUNICATIONS MIDDLEWARE SURVEY AND EVALUATION

Dr. Roland Kranz
John Forrest Harrell

GenCorp Aerojet
P. O. Box 296
1100 West Hollyvale Street
Azusa, CA 91702

August 1998

Final Report

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION IS UNLIMITED.

AIR FORCE RESEARCH LABORATORY
Space Vehicles Directorate
3550 Aberdeen Ave SE
AIR FORCE MATERIEL COMMAND
KIRTLAND AIR FORCE BASE, NM 87117-5776

AFRL-VS-PS-TR-1998-1071

This report has been approved for publication.

GEORGE S. SCHNEIDERMAN
Project Manager

FOR THE COMMANDER

KEITH SHROCK, D-III
Acting Chief, Space Sensing and Vehicle Control Branch

CHRISTINE ANDERSON, SES, USAF
Director, Space Vehicles Directorate (VS)

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	27 August 1998	Final Report - 8 May 1997 through 4 June 1998

4. TITLE AND SUBTITLE
Controller and Communications Middleware Survey and Evaluation

5. FUNDING NUMBERS
F29601-97-C-0051
PE: 63401F
PR: 2181
TA: TC
WU: 02

6. AUTHOR(S)
Dr. Roland Kranz
John Forrest Harrell

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Gencorp Aerojet
P. O. Box 296
1100 West Hollyvale Street
Azusa, CA 91702-0296

8. PERFORMING ORGANIZATION REPORT NUMBER
Report 11149

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Research Laboratory
Space Vehicles Directorate
AFRL - VSS-Satellite Control
3550 Aberdeen Ave SE
Kirtland Air Force Base, NM 87117-5776

10. SPONSORING / MONITORING AGENCY REPORT NUMBER
AFRL-VS-PS-TR-1998-1071

11. SUPPLEMENTARY NOTES
None

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 Words)***
There is a need for a standard core architecture to reduce the costs of developing and maintaining ground control systems for different satellites. This report documents a survey and evaluation of next generation computing technology trends relevant to the need. In addition to the advent of the Internet, there are two important on-going evolutions impacting the area – the migration to distributed client-server computing and the paradigm shift to object-oriented (OO) programming. New application software is being developed based on a distributed object model and clients are tied to servers by software called "middleware." Available commercial-off-the-shelf (COTS) controller and communication middleware are evaluated for their applicability to satellite ground stations and for distributed processing, distributed database access, and distributed systems management. Available products and vendors are listed. Low-cost satellite ground station application software with the flexibility to adapt as requirements change can be built using COTS-available middleware. Although most middleware frees the application developer from the need for network programming, it nevertheless requires an experienced implementation staff for the selection of an integrated set of middleware products.

14. SUBJECT TERMS
Software, middleware, CORBA, survey, evaluation, ORB, satellite ground station, distributed applications

15. NUMBER OF PAGES
146

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	Unlimited

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18
298-102

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

➡ **Middleware Vendors & Products:** ➡ **See Section Six** ➡

LIST OF FIGURES

LIST OF TABLES

EXECUTIVE OVERVIEW

The primary goal of new satellite control - ground segment - technology development is fundamental: To improve operational performance. A close second aim is to reduce acquisition and O&M costs. These objectives can be met by the use of client/server (C/S) distributed computing and object-oriented (OO) programming. A C/S OO architecture provides scalability, extensibility, and portability.

A menu of added, and cost effective, capabilities is available through the use of COTS software. Today, COTS software is designed to simplify the adaptation to changing requirements. Software technology, based on open standards, provides consistent graphical user interfaces (GUIs), Object-Oriented Database Management Systems (ODBMS), and automated reasoning techniques (Expert Systems). "Middleware" is used to isolate the application software from the C/S multiple-level processing environment. These techniques enhance performance, provide the flexibility to adapt to new missions, and decrease over-all cost.

The challenge in proposing a computing architecture is that the design must be flexible enough to accept not only requirements changes but also the rapid advances in technology. New COTS software products, from idea-to-market, are being developed on a 6-month schedule. A major contribution to speed the development of complex distributed applications has been provided by the availability of COTS standards-based middleware. Middleware has relieved the developer from the need to keep track of, and to program for, the target-for-deployment system configuration.

"Middleware" is an inclusive term used to describe all of the software that is used to tie clients to servers in a networked computing environment. Middleware relieves the developer from having to write the complex inter-process communications code.

This report summarizes survey results on the availability and utility of COTS middleware products for developing the next-generation, common satellite ground station (GS). The survey is not exhaustive, but is considered "representative." The intent was to identify and evaluate the product-sector leading/high-market-share products. Vendor product names and offerings are constantly changing in response to competition. Thus the survey establishes a Vendor's "product capability," instead of concentrating on an exact product, or the "product version" presently available, or on a product that "will be available."

Due to the numerous mnemonics found in any discussion of middleware, and with the paradigm shift to object-oriented technology, some background material is included to provide direction to the product survey and evaluation. Established standards and technology trends are discussed, and provide the basis for the product recommendations.

The main competing middleware standards are the *Object Management Group's (OMG)* Common Object Request Broker Architecture (CORBA), an "open" standard, and *Microsoft's* Distributed Component Object Model (DCOM). The choice of middleware for a "common" GS must be open standards based. Thus CORBA compliance is recommended, since this has industry-wide acceptance.

Truly "off-the-shelf" or "shrink-wrapped" middleware products are not, as yet, available (1997). Most middleware is bundled as a part of an application development tool package. For example, full implementation of CORBA *Object Request Broker (ORB)* systems is not expected until the year 2000.

Most middleware today (1997) has been developed to support message-based systems (e.g. *IBM's MQSeries*, which is the market dominant Message-Oriented Middleware (MOM) for the enterprise, and is used to tie all *IBM* platforms together.) CORBA Messaging, the merging of messaging systems and CORBA applications, is in the proposal stage (1998). While CORBA is expected to own a majority of the enterprise, *Microsoft's COM* still holds a majority on the desktop. COM/CORBA interoperability is available, but it remains in the bridge stage.

Thus, Vendor product interoperability, Vendor support, and Vendor consolidations, are important considerations in selecting a middleware technology. Further, the GS software staff must have application development and system implementation capability. These are over-riding factors in the choice of the COTS-available middleware products.

Chapter 6 of the report lists COTS-available middleware Vendors and products. The products are divided by middleware category, and technology. A section also lists Vendors who provide distributed application, as well as OO application, and *Java*, development tools. Further sections cover remote data access via-the-Web products, and the relevant industry consortiums and standards organizations.

For quick reference, an alphabetical list of the surveyed COTS-available middleware Vendors and products is provided in Appendix B.

Representative controller and communications middleware for the next-generation, common satellite ground station, is provided by *IBM's Component Broker* product family. Distributed data access middleware is represented by *Oracle Server 8*. And, distributed systems management middleware capability is exemplified by *HP's OpenView*, an open systems management framework, with, for example, the addition of *BMC's PATROL* family of application management products.

It is concluded that legacy environments will continue to exist, and will need to be encapsulated by the OO paradigm. Relational databases will be extended to handle objects. The Internet will be a part of all applications. C/S distributed computing solutions based on middleware will reduce the cost and complexity of application development, management, access, and use.

1. SCOPE

1.1 Identification

This study is a survey and evaluation of commercially available "off-the-shelf" (**COTS**) middleware products for utilization in architecting the next-generation, common satellite ground station (**GS**). The study is sponsored by the Satellite Control and Simulation Division of the United States Air Force Research Laboratory's Space Vehicles Technology Directorate (**AFRL/VSSS**).

1.2 Objective Of This Middleware Survey and Evaluation

A middleware Vendor and product search is to be performed to find, evaluate, and propose a set of COTS middleware services that can be applied to automate the GS applications environment. This architecture uses the **distributed object model** (see Section 2.3.2) as the method for isolating clients and services from environment-specific communications and data access mechanisms. A list of middleware vendors and products is provided in Appendix B, with recommendations for implementation.

1.3 Study Overview

The objective of this study is to examine the availability and the use of COTS middleware communication products (i.e. COTS middleware software) for linking GS analyst workstation applications to real-time derived or archived satellite telemetry data, as indicated by Figure 1-1.

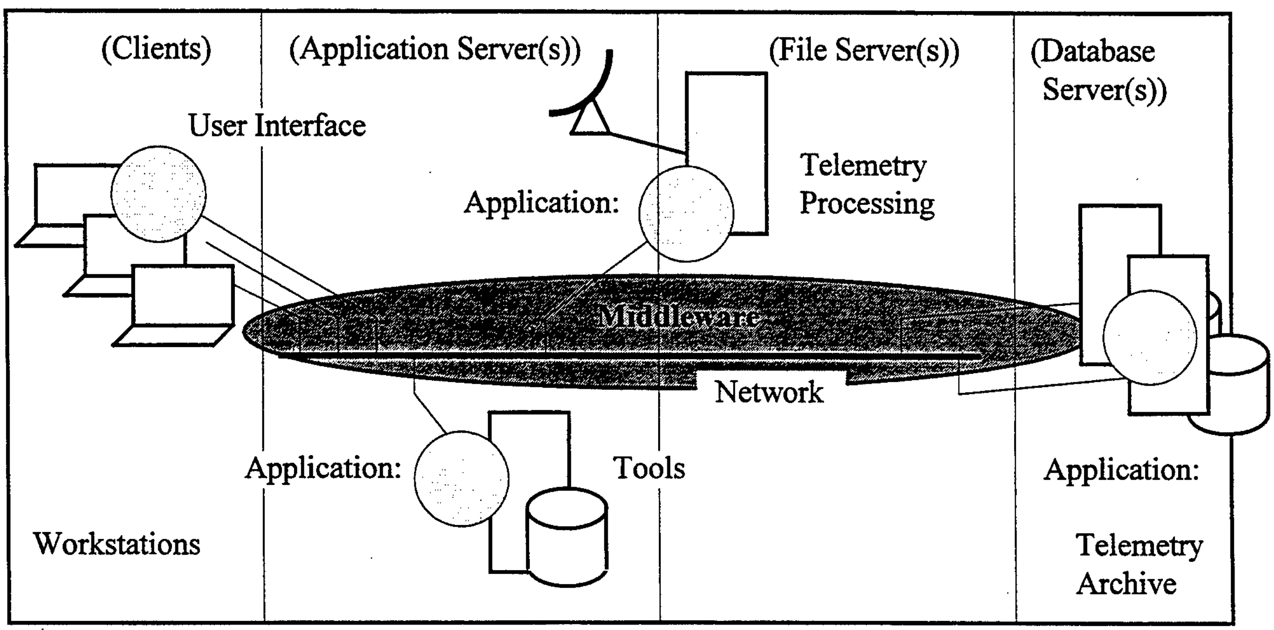


Figure 1-1. Use of Middleware

The appeal of middleware software is that "it is a tool that simplifies tasks." Specifically, it hides the system-specific communication interfaces, and the network location of distributed services from a service-requesting client (the application).

1.3.1 The Client/Server (C/S) Computing Environment

The C/S computing environment is a special case of distributed cooperative processing. A "client" system requests services to be performed by a "server" system. In C/S, the term "client" usually refers to an application program, and the term "server" refers to a program that provides data. Applications, in this environment, are called "distributed," "networked," or "client/server." C/S models can be distinguished by the service they provide. See Section 5.2 for further discussion of the C/S models, components and styles.

The benefits of a C/S architecture are:

- Lower hardware and software costs.
- Easier to use, standard user interface.
- Flexibility of an open systems environment.
- Flexibility to support changes, in technology, and in requirements.

1.3.2 What Is "Middleware"?

Middleware is software, and refers to the various methods of establishing communication between a client (application program) and a (remote) server (resource), (such as a database).

Middleware is a term applied to a set of **run-time software services** that architecturally resides above the communication protocols. This "infrastructure" software is used to isolate application software from the need to know C/S implementation-specific communication and data access mechanisms.

Middleware isolates application logic, which is distributed across platforms, from the detailed inter-process communication and network protocols.

Middleware provides communication (physical and logical connection) **between** clients **(applications) and** many/different servers (the **source data**).

Middleware can be further defined as a "**set of** Application Programming Interface (**API**) invoked **software services, formats and protocols that** interact with the operating system (OS) and network services, and protocols and **provides an economic infrastructure to allow the location of an application transparently across a network.**" [8]

> **Middleware is software that is used to obtain a service or data in a distributed processing environment.**

Or, from a user's perspective,

> **Middleware is the term for the software that connects two computers together.**

> **Middleware generally starts with an API.**

For example, basic data access middleware consists of both, a client-side API, which initiates the data request, and a server-side multithreaded "catcher," which handles the data request. The API provides the call, and the catcher translates the call into server-specific commands.

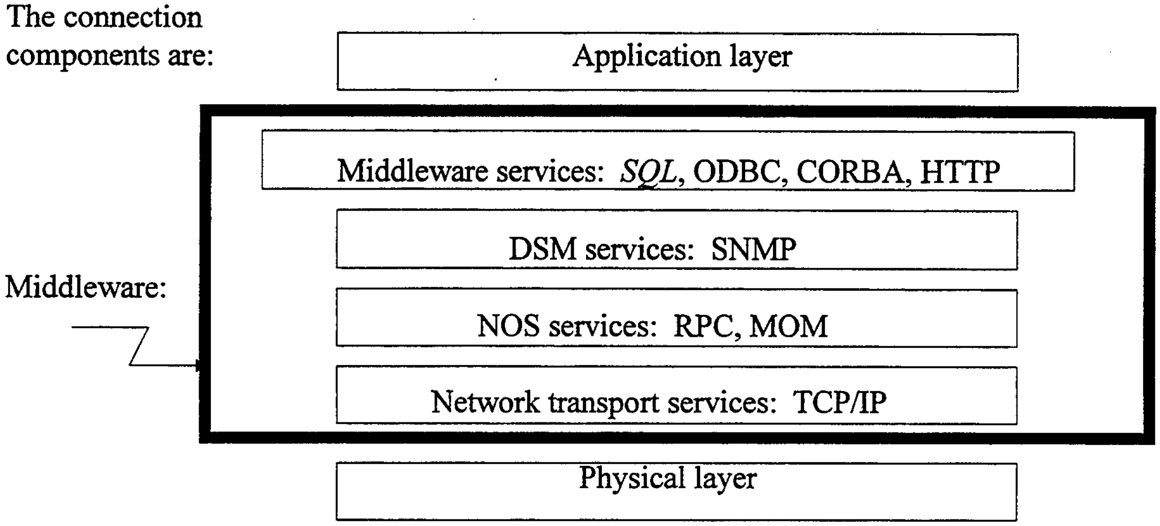
Software other than applications, operating systems and databases is defined as "communication software." Middleware is communication software that facilitates C/S interaction. It architecturally resides above the low-level communication protocols.

> Middleware <u>does not include</u> the client software that provides <u>the user's interface</u> or <u>the application logic</u>, nor the server <u>software that provides the service</u>.

Middleware may be grouped by intended use as the "traditional middleware" used within the corporate network (or in the GS environment) and "Internet middleware." The World Wide Web (or **Web**), the world-wide hypermedia system deployed over the Internet, can be considered as a very distributed C/S application, and as middleware. The emphasis of this study is to take a look at the traditional middleware. However, as most Vendors have Web-enabled their products, or have added products that enable communication between Internet clients and other resources, the impact of the Internet is also assessed. (The techniques used to implement the Web may replace other forms of middleware in the traditional Client - middleware - Server connection (see Section 3.2.))

1.3.3 The Client - Middleware - Server Connection

The connection components are:



1.3.4 In Comparison, What Is "Groupware" ?

Groupware is software, and refers to the various methods of establishing communication between people using networked computer technology. Another term for groupware is collaborative software. A prime example is e-mail.

Groupware deals with multimedia document management (automation of image handling), workflow automation (routing of work, as for example, shop order routing in a manufacturing environment, document routing for loan processing, or co-authoring a paper), and other areas (such as PC-based voice-mail) which allow people, and businesses, to communicate. Examples of groupware are *IBM's Notes/Domino*, the comprehensive all-in-one groupware market leader which was responsible for *cc:Mail*. Also, *Microsoft's Exchange* and *Novell's GroupWise XTD*. *Exchange* includes *Microsoft Mail*, and *GroupWise XTD* includes *Novell's* mail product *GroupWise*.

Groupware-specific middleware comprises the e-mail server-to-server backbone infrastructure that provides, not only for client access to other mail networks, but also allows application-to-application exchange of files, documents, workflow events and images. The competing mail backbone APIs/standards are discussed in 4.6.

The groupware Vendors are embracing the Web standards, but at present (1997) do not support distributed objects. This technology is not covered here. See 1.4.1, and 4.6.

1.3.5 What Is "Data Warehousing"?

A **data warehouse** is an information systems architectural construct consisting of an **intermediate server** that copies data from a number of multiple database servers. Client systems interact with the intermediate server, via a LAN. The data is presented to the user as "read-only," and is consolidated and summarized to support the management decision process. This provides a consistent view of the enterprise, and safe-guards the original data. This also allows location of the data closer to the user for faster, less-expensive, data access.

Database-centered C/S applications can be divided into two (2) classes: The above discussed **data warehouse server** , which provides the "framework" for automating the corporate decision-support process, and Online Transaction Processing (**OLTP**) systems.

The analysis and consolidation of the warehouse data is performed by a Decision Support System (**DSS**), which focuses on providing a consolidated view of the enterprise. Due to the possibility that extensive data correlation, plotting, etc., may be required, the use of DSS tools is not considered time-critical. The DSS process therefore is termed as "NOT mission critical."

Data warehouses focus on "getting data from the database" (i.e. users query, and retrieve data). The data is considered to be "<u>informational data</u>," i.e. data that is organized around subjects, such as vendors, and products. The data use is "strategic," to gain a strategic advantage over competition. Special **data mining** tools have been developed to sift through the data to look for unsuspected patterns or significant factors. Other advanced information processing techniques are also applied, such as Online Analytical Processing (**OLAP**), which deals with answering complex (...broken down by ..., or compared to ...) queries. OLAP, and standard (*SQL*) queries, return data that satisfies a query. Data mining tools return data based on discovery, rather than on a question.

The informational data, for decision support, is extracted from production databases. The warehouse server data is usually not fully up to date, and needs periodic replication. Copying the data to the warehouse server is the function of **replication services**, which are considered middleware services.

OLTP systems focus on "transactions," which consist of both, "getting the data stored" (i.e. adding data), and on "getting data back." OLTP deals with "<u>production</u>," or "<u>operational data</u>," i.e. data that reflects current values. Operational data stores focus on the state of the business in real-time. Operational data focuses on transactional functions, such as bank withdrawals and deposits. Transaction processing is discussed further in the next Section, 1.3.6.

The data warehouse is a 3-level (see 5.2.2) C/S architecture. The data warehouse server is most likely a PC running inexpensive copies of, for example, *Oracle*, or *Microsoft's Access*, depending on the amount of data and the number of users.

Data warehouse solutions, or frameworks, are for example, provided by *IBM*, with it's *Information Warehouse, Sybase*, with *Warehouse Works*, and *Oracle*, with *Warehouse Technology Initiative*. This technology is not covered here. See 1.4.1, and 4.7.

1.3.6 *What Is "Transaction Processing"?*

Transactions consists of a sequence of predefined actions. A computing system that performs transactions is called an OLTP system. The database usually contains operational data (i.e. the data reflects current, updated values), and is stored on a **production database server.** Transaction Processing **(TP)** tools, or **TP Monitors**, control access to the databases in this environment, and are considered middleware services. TP Monitors manage the transaction process.

Operational data requires database integrity, security, and high availability (<10 sec. response time). For faster response, the client typically invokes remote procedures on the production database server. The OLTP application thus resides on both the client and the server.

This can be defined as a 3-level (see 5.2.2) C/S architecture - if the TP Monitors manage the application process independently from the front-end GUI and the back-end databases. Examples of TP Monitors are *IBM's CICS* family, and *IBM/Transarc's Encina.*

OLTP tools are tied heavily into a business's/organization's functioning, and are considered as "mission critical." TP Monitors typically are used in managing C/S OLTP applications with thousands of nodes. This technology is not covered here. See 1.4.1, and 4.8.

1.3.7 *Architecting the "Next Generation, Common" Satellite Ground Station (GS)*

A satellite ground station is defined by its mission: Strategic, Tactical, Combination, Manned, Weather, Commercial, Communications, Earth Resources, Surveillance, etc. The GS may be further classed as Fixed Data Processing, Mobile Data Processing, Data Relay, etc. Looking for commonality, the functions that need to be performed by a *USAF* GS may be stated in generic terms as "mission" or "TT&C." The discussion in this study centers on the requirements for TT&C, and more specifically, C&S. The GS considered here is thus defined by the requirements for satellite Commanding / GS Control (C) and Health & Status (H&S), or, simply Status (S). The GS can then be designated as a "satellite ground control system." The middleware we are looking for is then designated as "controller and communications middleware," or, simply **"communications middleware."**

1.4 *Study Areas*

Since COTS products are the main consideration, C/S development tools are not evaluated. Several middleware categories also come under the heading of "separate concepts," and thus are not evaluated. Wherever possible, however, references found during the study period, to all C/S relevant vendors, standards, and tools, are also documented here.

1.4.1 Not Included

- It is assumed that the GS architecture does not include 3-tier hardware computing, i.e. connection to (legacy) mainframe systems. The computing architecture is client/server, NOT enterprise server. However, at the same time, a possible requirement for hardware scaling from 2-tier to 3-tier hardware is kept in mind (see Section 5.2).

- Middleware is generally considered to "sit on top of" the network communication protocols, TCP/IP, IPX/SPX, SNA LU6.2, DECnet, etc., i.e. on top of the ISO transport layer. This communication system functionality has generally been structured to be invisible to the application developer. This C/S interaction level is well defined, and therefore not discussed in this study. The focus is on the "client-obtain-a-service from a server" middleware.

- Groupware deals with capturing unstructured data in a container called a "document," and provides a solution for handling multimedia in a C/S environment. It provides for automating document routing, and for such human interface activities as electronic conferencing and scheduling. These "collaboration in getting the work done" automation applications are not in the primary need category in the GS environment.

  The groupware environment is the large office or enterprise. There is a large market, with many vendors. Some components, such as e-mail, are widely used. Groupware has its own standards as well. Thus, since groupware is universally used enterprise software, and available to be added at any time, the use of groupware products, such as *IBM's Notes/Domino*, are not considered here.
- Data warehouses/DSS systems are generally business-specific and are used to analyze data and to create reports. They are mainly for handling high-volume information queries. In the GS environment, database access is not considered "high volume", and this type of support is accomplished by data analysis tools, e.g. for trending analysis, and by Expert Systems. Thus DSS tools are not considered here, but may be added to the GS tool repertoire in the future.

- Most COTS OLTP systems reside on mainframes, and are used by banks, airlines, stockbrokers, etc. The user typically interacts with a production/transaction server on the 2nd level, which pulls data from 3rd-level databases. TP Monitor, or **TP management** software is used to ensure transaction integrity. In the GS - non-enterprise - environment, the user has local and immediate control of the data. Since the database is local, and with a limited number of users (of the order of 10 or less) OLTP tools, such as TP monitors, are not needed.

  As added incentive for not considering TP Monitors in this study, is the expectation that they will eventually graduate into TP Monitors for components, and will thus be designated as *ORBs*, and can then be re-considered.

1.4.2 Included
Satellite ground station software can be divided into components based on function, for example: Mission, C&S, Display, Support, Telemetry (TLM), Training, Simulation software, etc. The division may also be by processing stream functions, for example, for a down link: Front-End Processing (satellite-specific TLM processing), Back-End Processing (Mission processing), and Display and Support Processing (Display and Control & Status). Further sub-division provides software components for: Analysis, Expert System (ES) analyst/operator decision support, Database access, Communications, Data Logging, etc. In order to address the GS software requirements in generic terms, no attempt is made here to classify the GS software by function.

The GS is designated as a "satellite telemetry processing and monitoring system." This means we propose that the **GS "core" functionality requirements can be represented by a function-based distributed application C/S model**, that incorporates clients, application and data servers, AND middleware.

- C/S middleware development tool vendors/products are included for reference only.

1.5 Document Overview
In order to evaluate the available COTS middleware products, we must understand what we are looking for. This report thus consists of two main parts: (1) A definition of the study objectives, and (2) the product survey and evaluation.

Style used

Conclusions applicable to architecting the next-generation, common GS:

Middleware:

Vendors, industry consortia and standards organizations are shown in *Times New Roman Italic* font. When the subject is software and not a generic designation, it is also shown in *Times New Roman Italic* font. For example, the Object Request Broker *ORB.*

1.5.1 Primary Purpose of Report
This report provides a list of COTS middleware products which may used in architecting the next generation, common satellite ground station.

1.5.2 Section Contents

The report follows a define ➡ survey ➡ evaluate ➡ recommend products sequence:

1. SCOPE
   Background and introduction to C/S and middleware.

2. SUMMARY OF MIDDLEWARE REQUIREMENTS
   GS operational and middleware **requirements**.

3. MIDDLEWARE SURVEY AND EVALUATION
   Survey of available COTS products,
   impact of the Internet, and
   evaluation for suitability for use in a GS environment.

4. MIDDLEWARE TECHNIQUES AND STANDARDS

5. ARCHITECTING THE NEXT GENERATION, COMMON SATELLITE GROUND STATION
   Architect the GS.

6. **MIDDLEWARE VENDORS & PRODUCTS**

7. MIDDLEWARE FOR NETWORKED SYSTEMS MANAGEMENT

8. CONCLUSIONS, O&M, and
   PRODUCT RECOMMENDATIONS
   Future trends, effect on O&M.

A. NOTES Glossary, Acronyms

B. Alphabetical List: Middleware Vendors

REFERENCES

Follow-On:
Proof-of-Concept.

2. SUMMARY OF MIDDLEWARE REQUIREMENTS

2.1 Satellite Ground Station (GS) Operational Requirements

The basic form of a satellite ground station is the same, regardless of the system in which it is used. Reference the 1.2.5 and 1.4.2 discussion. The generic GS hardware architecture contains antenna(s), telemetry processing, data processing, (intelligent, i.e. PC or Workstation) displays, and terrestrial communications. Our concern is with **communication between the displays and the data processing elements**: Clients and Servers, which are connected by a Local Area Network (LAN). Reference Figure 1-1.

The generic software architecture consists of applications, database software, and communications software. The down link software performs telemetry data capture, processing, display, analysis, storage and reporting. Up link software adds two-way telemetry data link. Our concern is with providing **the applications access to information**: Clients and Servers.

2.1.1 What Do We Mean By "The Next Generation, Common" GS?

The GS technology vision is to reduce the cost to acquire, maintain and modify a GS for different satellites. This would be accomplished by developing a standard distributed open systems GS core architecture that can be reused for different satellites. The core/common architecture decreases costs through use of standards-based COTS software, that is flexible to adapt to changing user requirements, and standard PC hardware. All components would be "open," i.e. Vendor independent. Operation and maintenance costs (**O&M**) would be reduced by the use of affordable COTS products that are highly re-configurable, by creating autonomous processing systems, and by adding automated reasoning techniques.

Specifically, the GS would use dynamically reconfigurable and reprogrammable hardware elements, industry standard expansion bus interfaces, and parallel architectures. Software would use object-oriented (**OO**) software development and database techniques, graphical user interfaces (**GUI**), Expert Systems (**ES**) to aid in operator decision making, and open systems communications standards and technologies, such as Common Object Request Broker Architecture (**CORBA**), Object Linking and Embedding (**OLE**), and the World Wide Web (**WWW**), with systems management conforming to standard protocols, such as the Simple Network Management Protocol (**SNMP**). Data access and distribution would be over standard Local Area Networks **(LAN)** such as **Ethernet**, using standard protocol suites such as **TCP/IP.**

2.1.2 Robustness Requirements

Access to data generally must be 7x24, with backup and recovery, and allowing application and system on-line upgrades.

2.1.2.1 Fault-Tolerance

Fault tolerance is generally built on redundancy, such as hot and warm standby, both hardware and software processes.

2.1.2.2 *Scalability*
Infrastructure software (i.e. "middleware") must support data access using a variety of platforms, operating systems, and networks, as well as via the Internet and various intranets.

2.2 *Impetus Given To the Study By the Air Force Research Laboratory's Multimission Advanced Ground Intelligent Control (MAGIC) System*
The ***MAGIC*** system is a satellite ground station architecture testbed. The objective was to develop new technologies for the next generation of satellite ground stations, via a multi-year, multi-phase program. For example, see [10]. The vision was to develop a GS core architecture, using open distributed COTS system components. The aim was to reduce acquisition and O&M costs, while also providing the flexibility to adapt to new missions, and allowing for easy introduction of future technology to enhance operational capability.

The ***MAGIC*** technology vision of a distributed open architecture of "small components that communicate through message passing" provides the impetus for considering C/S computing.

The ***MAGIC*** technology vision calls for use of object-oriented (OO) software development. This provides the impetus for the use of an **object-oriented middleware software** layer to facilitate the interaction between distributed objects in a networked, or distributed computing environment.

2.2.1 *MAGIC System Environment*
A standard telemetry capture, analysis and reporting environment is envisioned. A <u>pre-pass</u> set up loads the front end, TLM processor, with the telemetry stream format and calibration information for a particular satellite, and connects and initializes a workstation. During the pass, the satellite can be monitored in <u>real-time</u>, processing the telemetry data to display satellite status. Anomaly identification and resolution, supported by an expert system (ES) is made available. All information received from the satellite is archived in a relational database for later retrieval for analysis. <u>Post-pass</u> analysis and plotting tools are available.

2.2.2 *MAGIC System Applications*
The ***MAGIC*** application software consists of seven (7) major components that communicate by message passing: The TLM Front End, Master Control Program (MCP), Display, Databases (Telemetry and Mnemonic), Expert System (ES), and Analysis Tools, as shown by Figure 2-1.

The ***MAGIC*** workstations and servers run the *Windows NT* OS. The use of *Windows*, vs. *UNIX*, choice was used to "keep the costs down."

The middleware that allows connection to the databases is Open Database Connectivity (**ODBC**), which is actually a C-language API specification. This API is a component of *Microsoft's* Windows Open Services Architecture (**WOSA**). The telemetry and mnemonic (satellite unique information) databases are controlled by *Microsoft's SQL* RDBMS. The Workstation database access and analysis functions are controlled by *Microsoft's Access.*

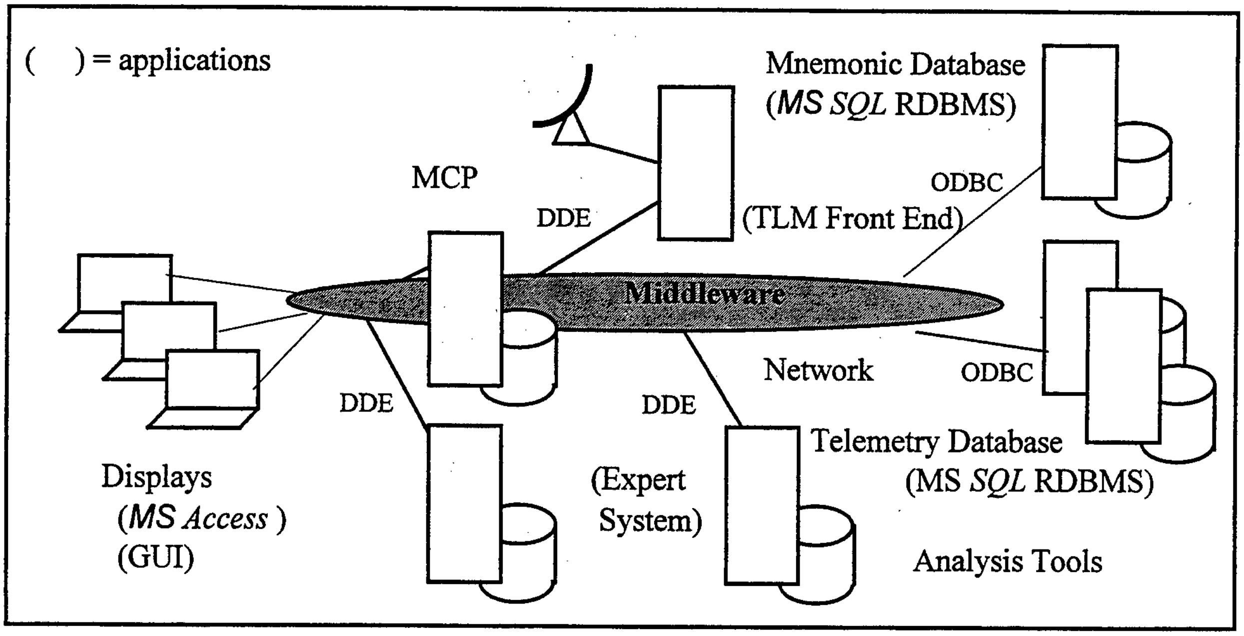


Figure 2-1. The *MAGIC* Software Architecture

The client applications must be ODBC-compliant, i.e. must use ODBC drivers. ODBC is also available for *UNIX.*

Application interprocess communication (IPC) is by means of the Network Dynamic Data Exchange (**DDE**) message-passing protocol. This is the middleware which allows the software components to communicate with each other. The applications use shared memory to exchange data. Network DDE is provided as a part of *Microsoft's Windows NT OS.*

2.3 GS Controller and Communications Middleware Requirements

The GS middleware requirements are derived from the generic, or "common" GS systems development paradigm discussed in 2.1.1. All software components must:

- Be Vendor independent.
- Communicate through standard protocols.
- Can operate on many different hardware platforms.
- Provides individual functionality. (Can thus be easily changed. Designated as "Little COTS," as opposed to "Big COTS," which integrates many functions. [10])

Questions to be asked are:

- Is there a core GS operational reason for distributing applications?
- Are there many operators or users with different needs or different missions?
- Is access required to different applications?
- Are there a variety of databases?
- Is point-to-point (two-level connectivity) possible, for increased performance and decreased administrative overhead? Or, is access required to mainframe (legacy) databases, via a gateway, in a three-level architecture?
- Can thin clients be used? With an application server?

2.3.1 Transition of Legacy Applications

The connection to legacy applications can be provided at the second C/S level, e.g. by: (1) "Screen scraper" software, which captures the contents of mainframe terminal screens, or by (2) middleware that either (a) puts the response (from a mainframe) data into HTML, and presents it to a browser, or (b) "wraps" the data for presentation to an *ORB*:

2.3.1.1 Data Access

Most satellite ground stations have a large investment in ground data processing software and in archive databases. This legacy investment can most easily be integrated with an object-oriented system by encapsulating the data. COTS (proprietary, however) object/relational products which support object extensions based on *SQL3* are available which read the "schema" of a database and generate class definitions to represent the data as objects.

For data stored in a RDBMS, each row, of the tabular row and column format of the relational table, represents an "object." Each column represents the "instance" variables of the object. (See 4.2.1.1.3 for a discussion of *SQL*.)

Examples: *IBM's Common DB2 Server (DB2 2.1)*, which runs on *UNIX*, *OS/2*, and *NT* platforms, *Oracle's Oracle 8* database, and *Illustra/Informix's Universal Server.*

2.3.1.2 Wrapping

Legacy applications may be expected to be re-packaged as object resources by **wrapper** code. For example, see[A].

Both yesterday's legacy systems and tomorrow's open systems must be accommodated. Programs on one machine must talk to programs on other machines. A "wrapper" encapsulates code and data into a single entity, and allows treating existing data and file formats as distributed objects. Interoperability is provided by middleware services that hide the complexities of different operating systems and make interconnections by any LAN and WAN protocol. A major wrapping consideration is to provide for a separation of the legacy functionality to match the C/S distributed object model, to allow multithreading.

An example of middleware for open access to legacy systems is provided by *NetWeave's NetWeave Server*, which resides on host/mainframe, workstation and *SQL* database platforms and provides legacy message and database services.

2.3.2 The Distributed Object Model

An "**object**" is an abstraction which combines both the data structure and the procedures that are implemented on the data in a single entity. Objects are reusable and extensible, and encapsulate data and the procedures that can be used to manipulate the data. See, for example, [a] for a discussion of the paradigm shift from traditional procedural programming to object-oriented development.

Object-oriented programming (OOP) code is modular. Programming chores are separated into components. Since objects have no set size and are not linked to specific operating systems or protocols, objects may be packaged as individual pieces of code. In a networked computing environment the objects may be hosted on more than one platform, and can be thought of as independent software **components.**

A **distributed object model** defines software components that are not bound to a particular platform, computer language, or implementation.

Further, objects can request services of other objects. If the request is to be independent of any concern for where the two objects are, or how they accomplish their respective tasks, the objects must communicate with each other via an *Object Request Broker (**ORB**)*. The *ORB* allows objects to communicate irrespective of the specific platforms and technique used to implement the objects. **The *ORB* thus functions as the middleware for objects.**

The *ORB* concept was developed by the *Object Management Group (OMG)* consortium, by adding OO features to the standard RPC defined by the *Open Systems Foundation (OSF)*. The *ORB* routes service requests and responses. See Section 5 for further discussion of the *ORB*.

Standards-based objects can be used to link disparate systems.

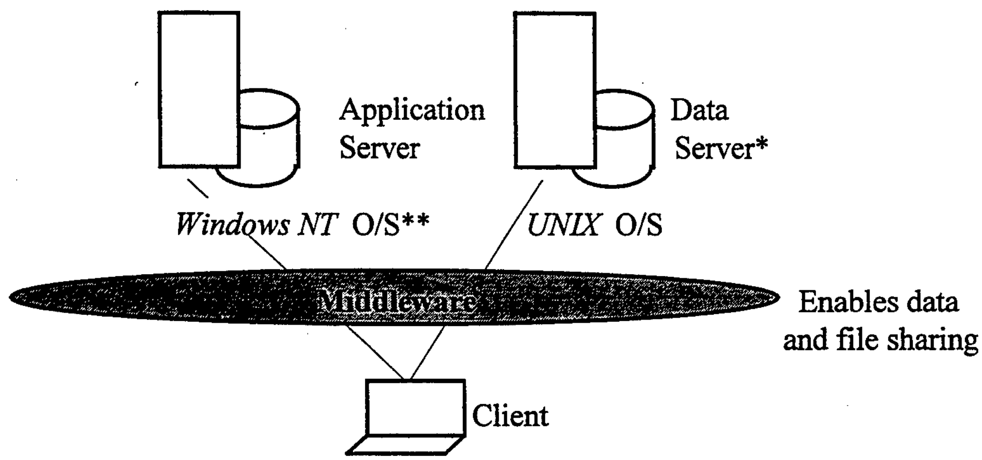
At least three (3) distributed object model standards have been proposed. These "*ORB* standards" describe how individual applications inter-operate and locate object resources. See Section 4 for further discussion of the standards.

OOT, merged with distributed computing, gives **"distributed objects"** [16]. That is, objects can be viewed in terms of the operations with which they can be manipulated. If we say that a set of run-time software services uses the "distributed object model" as the method for isolating clients and services from the environment-specific communications and data access mechanisms, we are talking about **"distributed object middleware."** This is the middleware we require to operate the next generation, common GS.

The common OOP languages are *C++* and *Sun Microsystem's Java.*

We also talk about **CORBA objects** and **Java objects**. The use of *ORBs* as a middleware framework is growing, but the middleware market (1997) is still dominated by OLTP monitors.

3.4 GS Mixed UNIX/Windows Computing Environment



* C/S databases were initially developed for *UNIX: Oracle, Sybase, DB2, Informix.*

** *Microsoft's BackOffice* server "suite" includes: The *SQL Server* database management system developed by *Sybase* and ported by *Microsoft* to *NT.*

Figure 2-2. ***UNIX*** **and** ***NT,*** **The Best of Both Worlds**

In the PC LAN arena, *Microsoft's Windows NT* Server is expected to pick up market share from *Sun's Solaris*, *Hewlett-Packard Co.'s HP-UX*, *IBM's AIX* and *The Santa Cruz Operation's (SCO) UnixWare*, *UNIX* network operating systems. However, this market encroachment is expected to be from an increased adoption of NT for the low-end, mass-market *Wintel** Work Group Server platforms. *NT's* main growth has been at the departmental and workgroup level, against the competing operating systems: *Novell Inc.'s IntranetWare* and *IBM's OS/2 Warp Server*. *Microsoft*, so far, has committed to supporting *NT* on *Intel*-class PCs and *Digital Equipment's* Alpha platforms.

Mission-critical and enterprise-level applications have traditionally run on *UNIX* platforms, and may be expected to continue doing so for some time. The attraction of *UNIX* has been its performance tunability for specific applications and platforms, reliability, and scalability. Scalability, via symmetrical multi-processing (**SMP**), is expected to continue to be a *UNIX* price performance advantage. And, *Silicon Graphics, Inc's. (SGI) IRIX*, for example, may be expected to continue in popularity in the high-end graphics Workstation market. Table 2-1 compares the advantages of *UNIX* and *NT*.

(Note: *Microsoft* is expected to enter the clustering arena with two-node fail-over, in 1997. [1])

Table 2-1. ***UNIX* and *NT***

The advantages are: *NT*	*UNIX*
- Price ($)	- Scalability, via clustering
- Easier end-user training and administration	- Application support (large installed base)
- Lower cost of ownership ($)	- Application tunability (for best-performance)
	- Platform support (*UNIX* runs on *Intel* and *DEC* Alpha, as well as on *MIPS,* PA-RISC, PowerPC and other platforms)**

Enhancements that may be expected are:	
- Increased 3rd party application support	- Enhanced ease of use
	- Enhanced ease of management

What determines if a federal agency buys *Wintel* or *UNIX*? Unless there is a strategic shift to new information technology, the choice is based on (1) the installed base, and (2) the existing investment. What this means is we can expect continued *NT* and *UNIX* coexistence in the GS operations environment for some time. This has been acknowledged by all *UNIX* Vendors, by providing software to allow data transfer across *NT* and *UNIX* machines on the network.

Middleware strategy thus calls for software that runs across both, *NT* and *UNIX*, as noted in Figure 2-2.

IBM strategy is to supply **middleware** that runs across all platforms: *NT, OS/2 Warp Server, UNIX,* and mainframes, and is based on the Distributed Computing Environment (DCE) and Distributed File System (DFS) standards. [11]

(Note: *Microsoft* has announced, May 6, 1998, the development of a *Windows NT Services for UNIX Add-On Pack*, to ease integration of the *Windows NT Server 4.0* into existing *UNIX* environments.)

---

* The "Wintel" platform combines *Windows NT* and *Intel Corp's* PC processors.

** CISC architecture platforms: *Intel's* Pentium. RISC architecture platforms: *Digital* Alpha, *SGI's MIPS, IBM's* RS/6000, Apple, *IBM,* and *Motorola's* PowerPC, *Sun's* SPARC, *HP's* PA-RISC.

2.5 Security Requirements

Most Network Operating Systems (**NOS**), as well as *UNIX,* provide "**C2**" level security. This is a government standard which requires application and user authentication. Unless government security classification data is involved, C2 level security is deemed adequate for GS operations. The C2 requirements are: A client authenticated user ID, server resource protection by access control lists (**ACL**), and audit (user activity) trails. Authentication is provided by the DCE-adopted and augmented *Kerberos* session-key protocol. A NOS will also including the capability for "single-log-on" security, allowing a user to access any server resource.

Government classified information control is specified by DoD security regulations.

Data encryption can be provided by private keys (*Kerberos*), or public/private keys, e.g. based on the *RSA* public-keys for electronic signatures, or the Data Encryption Standard (**DES**) private-keys for data.

Communications encryption can be provided if needed, by cryptographic systems.

Backup copies of data and program files, passwords, and physical security of the GS computer center and specific workstations with badge readers, retina recognition, or combination locks is common. Computer use ethics statements rise awareness for data sensitivity and the need to protect data. User management, such as separation of duties, is widely enforced. Scanning for computer virus invasion counter-measure programs are routinely used. Detection of security breaches methods, such as hiding special instructions and computer use logging, are also to be expected.

Transaction control, database concurrency (same-time update) control, data integrity control, etc. are generally incorporated within the various software products and DBMSs.

Access to a GS's data via the Internet introduces, due to the easy accessibility, new security threats. Internet firewalls, and groupware **S/MIME** (Secure/Multipurpose Internet Mail Extensions) and *Netscape Navigator 4.0* **SSL** (Secure Sockets Layer) protocols, for example, are needed to transmit information over the Internet. *VeriSign's OnSite,* for example, provides e-business **PKI** (Public-Key Infrastructure) services to operate a certification authority.

2.6 "Open" System Requirements

The two main goals of the "open" system are **portability** and **interoperability**. Both are cost effectiveness issues, and are discussed throughout this report.

2.7 Bandwidth Requirements

MAGIC's message passing with DDE (Dynamic Message Passing, a *Window* inter-process communications mechanism) may incur a queuing overhead, which affects system response times. Should DDE be replaced with another communications technique?

With message-oriented middleware (**MOM**), the location of the queue affects performance. Locating the queue in memory (i.e. DDE uses shared memory for data exchange between applications) does speed access, but locating it on disk provides for recovery if a system goes down.

Some considerations:

- Performance overhead is incurred as the layers of software involved in the C/S process increase.
- Early binding, which uses RPC's to call stored database server procedures, rather than calling them with *SQL* (late binding), speeds up the process over many calls.
- Performance can also be improved by (1) making middleware API calls, and the call result collection, at native code speed (i.e. the application should consist of natively-compiled executables), and (b) by multithreading front-end applications (by allowing queries, to a database for example, to execute in the background).
- A more obvious performance improvement is gained, for BLOB data, by using larger packets.

ORBs ride on top of RPCs and MOM, and thus have the same performance impacts.

2.8 Government Trends in the Use of Middleware

The operative approach today is to establish a "framework" that will allow utilization of the various technologies available in the commercial environment - as opposed to "specifying the design."

In the **groupware** arena, the *Defense Information Systems Agency (DISA)* is in the process of certifying *Microsoft Windows NT* for use by the *DoD's* Defense Message System (DMS), but allows local sites to use, for example, either *Microsoft's Exchange* or *Lotus Notes/Domino* (which also runs on *NT).* [2] *NT* is expected to become the "standard network operating system for e-mail throughout DoD." [3] But, until later ("imminent") releases of *NT* allow scaling-up to support 1000s of users, a large user base still requires *UNIX.*

NASA's scientific and engineering workstations, at Goddard Space Flight Center, Greenbelt, Md., use *Compaq's* dual Pentium Pros running *Sun's Solaris.* The *Animal and Plant Health Inspection Service (APHIS)*, in Ft. Collins, Co., plans to use *IBM's* **middleware** to connect *Windows* and *UNIX* environments. [4]

An example of COTS product integration is provided by the *Air Force's Center for Research Support*, which uses *Talarian's SmartSockets* messaging **middleware** [19].

Examples of ***ORB*** use are: *Iona's Orbix* by *ARPA*, and the *Los Alamos National laboratory* for a tele-medicine application.

Microsoft has announced, April 21, 1998, the signing of a **CRADA** (Cooperative Research and Development Agreement), signed April 14 at Hanscom AFB in Bedford, Mass., with the *U.S. Air Force Electronic Systems Center (ESC)* to begin converting *UNIX*-based Command and Control applications to *Windows NT*. *Softway Systems, Inc.* will supply it's *OpenNT* **middleware** for the project.

3. MIDDLEWARE SURVEY AND EVALUATION

3.1 Middleware Survey

This middleware survey was undertaken by (1) evaluating the functions that we want to provide, (2) examining the available COTS middleware, and then (3) evaluating specific middleware in depth, with the goal of providing a recommendation for use in the GS environment.

The over-all goal is to reduce complexity, not add to it. With a plethora of middleware on the market, two guidelines were followed to keep it simple: If possible, (1) limit the number of vendors/middleware products that have to work together, and (2) hide the lower-level middleware techniques, such as remote procedure calls (RPCs) and messaging and queuing, by using higher-level middleware services products that incorporate them.

3.1.1 Available COTS Middleware, By Functional Category [8][9][12][others]

Middleware may be grouped into three (3) functional categories (also designated as technologies, techniques, or types). In English language terms, the middleware categories are as shown by Figure 3-1 (a).

1. **Distributed processing middleware,**
   (also designated as program-to-program communication, application-to-application communication, network, or message-oriented middleware).

   Interaction between networked applications and operating systems is facilitated by sending requests and data in the form of Vendor, platform, operating system and networking protocol independent messages.

2. **Distributed data access middleware,**
   (also designated as application-to-database connectivity, distributed database access, or DBMS middleware).

   Provides a common, high-level programming interface, such as *structured query language (SQL)*, that packages information requests and replies in a uniform manner to allow interfacing multiple applications with multiple databases. The middleware translates requests into database-specific commands. Included are APIs (for example, *Microsoft's* ODBC and *IBM's* DRDA) and *SQL* gateways.

   Database middleware constitutes more than half of the (1997) middleware market.

3. **Distributed systems management (DSM) middleware,**
   (also designated as C/S systems management middleware).

   Provides continuous monitoring of distributed applications and systems functions for optimum quality of service.

Figure 3-1 (a). Available COTS Middleware, By Functional Category

In middleware language terms, middleware may be grouped into techniques, as indicated by Figure 3-1 (b). These techniques are discussed further in Section 4.

Category	Techniques
1.	1. **RPC** (Remote Procedure Call). Allows sending calls to another machine, across a distributed computing environment, synchronously. 2. **MOM** (Message Oriented Middleware). Allows sending messages (procedures, data, controls) between applications, asynchronously. 3. ***ORB*** *(Object Request Broker)* Allows objects to communicate with one another, across a distributed computing environment.
2.	4. **Remote data access** Usually provided by gateways that allow clients to access data from different database servers. 5. **TP Monitors** (Transaction-Processing monitors) Ensure transaction integrity: Ensure that every transaction goes through, it does not go through more than once, and that you are notified if a transaction fails.
3.	6. **Middleware run-time monitors**

Figure 3-1 (b). Available COTS Middleware, By Technique Category

Middleware grouping may also be in terms of the middleware market. In this case there are seven (7) categories, as shown by Figure 3-1 (c). Both major distributed object middleware (**DOM**) standards, CORBA and DCOM, are built on RPCs. Middleware products can also be placed in different market categories (for example, the *Bea/Novell Tuxedo* TP Monitor environment, may be considered to be PSM, MOM, or RPC middleware. Therefore, in this survey the COTS middleware products are grouped by functional categories - of Figure 3-1 (a).

3.1.2 Available Middleware Category Services

The middleware services that are provided by the middleware categories are cumulative, that is, they are added, in enabling distributed applications, as the scale and criticality of the deployed computing environment expands, from the Workgroup to the Enterprise, to external to the Enterprise.

Market: 1. Remote Procedure Call (RPC) 2. Message Oriented Middleware (MOM) 3. Distributed Object Middleware (DOM) 4. Remote Data Access (RDA) 5. Transaction Processing Middleware (TPM) 6. Publish/Subscribe Middleware (PSM) 7. World-Wide Web (WWW) Middleware

Figure 3-1 (c). Available COTS Middleware, By Market Category

The initial requirement is for a common set of low-level middleware service, such as DDE. This expands to the addition of middleware services for the different application classes, such as for access to database systems.

3.1.2.1 Category 1: Distributed Processing Middleware

Portability for passing information across a network is provided by *sockets, streams, named pipes, RPCs* and *ORBs*. Competing *sockets* are *Berkeley UNIX, System V UNIX* and *Winsock.* Competing *RPC* techniques are *ONC RPC* and *DCE RPC.* Competing *ORBs* are CORBA and DCOM.

3.1.2.2 Category 2: Distributed Data Access Middleware

In 1997, the majority of data is found in hierarchical storage. In terms of popularity, however, relational storage schemes/*SQL* databases have been developed and are in wide use. Distributed data access middleware performs database access across the network. Middleware services are provided by the *Structured Query Language (SQL)* Relational Database (**RDBM**) access solutions, including Vendor-proprietary *SQL* format and handshake **FAPs** (Format and Protocol), database gateways to other Vendor databases, *Microsoft's* Open Database Connectivity (**ODBC**) specification, and *IBM's* Distributed Relational Database Architecture (**DRDA**) strategy.

Examples of *SQL* databases are *IBM's DB2* family, *Oracle's Oracle 7*, *Sybase's Sybase 11*, and *Ingres'* and *Informix's* databases. In terms of market share, these are termed "the Big 5" by the *Gartner Group. Microsoft* is picking up a share of the *SQL* database market with *SQL Server* at the high end, and *Access* for the desktop. Object/ component databases, or relational databases with object extensions, to allow storage of complex data types, are beginning to be mentioned.

Object extensions on top of the *SQL* databases are being provided by *IBM* with "Common Server" *DB2 2.1* family, and by *Oracle* with *Oracle 8* (see 4.2.1.1.3 for a more detailed discussion). The object/relational databases allow storing of non-traditional data, or **complex objects** such as digital images (pictures), entire documents, CAD drawings, faxes, fingerprints, HTML files, spreadsheets, movies, sound clips, e-mail messages, etc., in a relational database. The *Oracle 8* approach supports a 3-level C/S architecture model (see 5.2.2).

Data access gateway software translates database access statements into the database access language of a target database, such as the proprietary *SQL* dialects used by the different database Vendors. The gateway approach follows the 2-level C/S architecture model (see 5.2.1), and allows access to other Vendor databases.

Complex data type management, such as "**BLOB**" (Binary Large Object) manipulation – rotation of multimedia data, or downloading (e.g. over the Web) of large text documents, medical X-rays, or engineering drawings - may require specialized query/query optimizer tools, and Object Databases (**ODBMS**s). Examples of ODBMs are *Objectivity's Objectivity/DB* and *Versant's* ODBMS.

Enterprise database-centered C/S applications fall into two (2) categories: Decision Support Systems (DSS) and On-line Transaction Processing (OLTP). [12]

3.1.2.3 Category 3: Distributed Systems Management (DSM) Middleware
C/S distributed systems management is handled by TP Monitors, and when allowing objects, by middleware **run-time monitors**, or **object monitors**. The dominant network management protocol today is the Simple Network Management Protocol (**SNMP**). The foundation for enterprise C/S systems management is the *Tivoli Management Environment (TME).* The *TME* architecture is based on the use of a CORBA-compliant *ORB.* DSM solutions must integrate with both SNMP and TME.

Most SNMP platforms are *UNIX OS* based, and serve as the basis for distributed systems management. Examples of open DSM platforms are *IBM's SystemView and Tivoli/TME, HP's AdminCenter,* and *Sun's Solstice Enterprise Manager. Microsoft's* DSM platform is the *Windows NT OS.*

3.1.3 What GS Middleware Is Needed, By Functional Category & Specific Service?
No one product may provide the total solution. A "plug-and-play" approach must thus be followed. For example, *HP's* open *OpenView* platform can be used as a core which allows for the addition of 3rd party multivendor system management applications.

3.1.3.1 Category 1: Distributed Processing GS Middleware Needed?
Does MOM know best? Yes, if guaranteed, once-only message delivery is required. However, in the GS arena, a CORBA *ORB* offers the complete range of connectivity.

3.1.3.2 Category 2: Distributed Data Access GS Middleware Needed?
If using *SQL* database servers, the *SQL* language may be proprietary but is expected to port to all the platforms supported by the database vendor.

Support is needed for multiple database types through a single middleware layer. This can be provided by an ODBC driver package, which packages the application request for data, transports it across the network to a specific server for processing, and after processing returns the requested data to the application.

3.1.3.3 Category 3: Distributed Systems Management (DSM) GS Middleware Needed?

Integrated systems and network management across all GS platforms is required. The solution must provide availability, integrity, and performance features, and control of all the C/S applications, via an **object monitor**, from a central operator console. A GUI-based client workstation uses DSM middleware (SNMP, or RPCs, MOM, or an *ORB)* to obtain management information from agents residing on the different platforms.

The client side (open DSM platform) provides a visual representation, from a dynamically discovered topology of agents, a view of the managed objects. A historical and trend database is maintained is by MIBs.

The systems management functions may be provided by a 3rd party, on top of the management platform facilities.

CORBA provides object monitor services via its *ORB* implementation, and Transaction Services.

The GS DSM framework must thus be (1) open (i.e. it must use industry standards for its main interfaces), and (2) built on top of a distributed object bus such as CORBA.

3.2 Impact of the Internet

The Internet has provided another category of middleware: **"Internet"** or **"Web-enabled" middleware.**

The Internet is today's largest, most extensive, C/S environment. The impact of the Internet on GS design is twofold: (1) The GS internal C/S implementation may be modeled on and/or use Internet and Web technology, and (2) remote users, may communicate with the GS via the Web, requiring an Internet firewall, and possibly secure communication links.

The group of technologies that provide for optimization of application resources in the Web *browser* is referred to as **"client-side technology."**

The primary Web technologies are the graphical Web *browser* - the Internet client software used to access Web information; HyperText Markup Language (**HTML**) tags, used to embed hyperlinks in, and to describe Web documents; the HyperText Transfer Protocol (**HTTP**), an RPC-like protocol for accessing Web documents and other resources such as image files; and the Common Gateway Interface (**CGI**) server protocol, which provides the "shell" for running executables on the server.

Included are also *Java* and *Java Beans*, ActiveX and *Visual Basic*, and *Java Script.*

Web technology is used to maintain a "common - Web page - look and feel":

- The page layout (e.g. a common layout of the home page, program information page, general information page, etc.) makes it easier for users to find the information they are looking for. For example, a task bar is used at the top of each page, with clickable links to the HOME page, for data/document SEARCH, and for on-line HELP.
- Information is accessed through clickable hypertext.
- Access is password protected.

Web technology:

- Allows linking to heterogeneous clients and servers.
- Allows applications to span both the Enterprise and the Internet.
- Finds the database where information is stored automatically, launching the appropriate application for the task to be done.
- Formats the data for viewing on the desktop.

CGI is the established HTML (Web) server protocol. CORBA IIOP servers (with a client-side CORBA *ORB*) are expected to coexist on the Internet with HTTP/CGI servers for the near future (1997), rather than acting as a replacement. [12]

The Web protocols operate above the Internet transport protocol – and de facto inter-networking protocol –TCP/IP, and HTTP software runs on virtually all major computer platforms. Traditional middleware shields the application from the network. Internet middleware shields the application from the Web protocols, i.e. HTTP, HTML, data access and state management. Examples are *Active Software's ActiveWeb*, *Bluestone's Saphire/Web*, and *Wafarer's QuickServer SDK*, which used a C/S/A (A = agent) architecture to communicate with the Internet.

A GS connected to the Internet must look at Internet security protocols, such as *Netscape Navigator browser's* Secure Sockets Layer (**SSL**) and Secure HTTP (**S-HTTP**). (Security is discussed further in Section 5.7.)

3.2.1 Web Browsers

Middleware enables heterogeneous clients and servers to communicate. This is what a Web ***browser*** does. The *browser* has also become the interface that is used for local files (i.e. "the intranet") as well as remote (on the network) files. The question then is: Can a Web *browser* replace other forms of middleware? The *browser* may provide a "friendlier" user interface, but other considerations, paramount being availability and data security, must be addressed.

Examples of Web *browsers* are:

- *Sun's HotJava* Web *browser*: Has "dynamic extensibility," which allows automatic plug-in of software modules when needed. Available for systems running *Solaris*, *Windows 95* and *NT*, and *MacOS* operating systems.

  Reference: http://java.sun.com/ for a free of charge individual, non-commercial download from the Internet.

- *Netscape's Navigator 4.0 browser*: A suite of Internet and intranet client applications. The *Professional Edition* includes user access to databases located on mainframes, and has *Netscape's* SSL 3.0-based security.

- *Microsoft's Internet Explorer 4.0 browser*: Has dynamic HTML support, which allows dynamic changes/updates of an element on the Web page, from any script on the page, without connecting to the Web server.

3.2.2 *"Webify" The Database Server?*

Web technology can be used to "front-end" a DBMS. A "**Webified**" DBMS allows the dynamic creation of Web pages - by using middleware to extract the database records. *Browsers*, instead of client-based applications, can then be used to initiate Web server application processing, which in turn accesses the DBMS. Also, a thin client could be used to run the *browser* and the net software needed to communicate with the Web server (see 5.2.3).

For example, IBM, *Oracle* and *Sybase* have extended their database products by using *Java* stored routines as methods for access to new abstract data types. *Netscape* and *Microsoft* use APIs to connect a Web server to the DBMS server: *Netscape's Server API*, and *Microsoft's Internet Server API*.

The question is: Should we tie the back-end applications to the front-end applications via the Web? It may be too soon for integration of mission-critical applications with the Web. However, secured data access across the Internet is a definite possibility (1997).

3.2.3 *Internet- Specific Middleware:*

3.2.3.1 *Security Protocols*

To-date (1997) "the" Internet security standard, and which is supported by *Microsoft* and *Netscape*, has been Secure Sockets Layer 3.0 (SSL). This standard has been the primary method for encryption of Web *browser* data. This is a session-layer protocol , and used mainly for TCP/IP streaming data. SSL is being replaced (1998) by Transport Layer Security (**TLS**), promoted by the *Internet Engineering Task Force (IETF)*. TLS mandates support for triple-**DES** (Data Encryption Standard) encryption, and also works with other transport protocols such as *Novell's Netware* SPX (of the IPX/SPX stack) and *Apple's* AppleTalk.

Two security protocols are currently used on the Web: *EIT's* Secure HTTP **(S-HTTP)** and *Netscape's* Secure Sockets Layer (SSL) [12].

The main requirement for a GS is to have as "security policy" in place. For example, encryption of SNMP network management tools should be considered. "Layering" is recommended, i.e. **firewall, trust, AND verify**.

3.2.3.2 Firewall Requirements

A "**firewall**" is a computer that is located between your private network and the Internet, and acts as a gatekeeper by filtering traffic. It can impose access controls and audit network traffic. There are generally two types: Network-level filters and application-level proxies. [25] Host/subnet **routers** work transparently at the network (IP) level, screening packets using filtering rules. A **proxy server** runs secure and trusted programs called "proxies" that filter e-mail, HTTP, etc. based on context, authorization and authentication rules.

If a firewall is configured to pass only HTTP traffic, IIOP messages can be an *ORB* in HTTP packets.

Examples of firewall software include *Sun Microsystem's SunScreen SPF-100*, a router-based firewall which uses the Simple Key Management for Internet Protocol (**SKIP**), and *IBM's eNetwork* LDAP cross-platform directory server, for storing user and security information. [24]

3.3 Alternatives to Middleware

Middleware allows heterogeneous client-server communication. This is what happens when we access data over the Internet. The "Web" can thus be considered as "middleware."

3.3.1 Web Page

Establishment of a server Web page, which can then be accessed using *browsers*, eliminates the problem of heterogeneous client systems. Database middleware would still be needed to connect the database server (level 3) to the Web server (level 2) (as expressed in 3.2.2).

3.4 Middleware Evaluation

In addition to the Users - or Operators and Analysts, for the GS case, the Support staff - or Information Systems (**IS**) personnel, may be divided into **application developers**, **database administrators** and **network administrators**. In a C/S environment, a fourth group may need to be added, the **infrastructure developers**. With the clear goal of keeping the need for development to a minimum (i.e. the middleware products must be truly COTS), the middleware product evaluation criteria - in order to minimize the need for a infrastructure development group - must focus on application transparency (i.e. on transparent access to data and services). The problem is to keep the evaluation criteria from becoming "how closely the middleware API matches the application developer's API familiarity"!

3.4.1 Middleware Survey Rationale

For purposes of this report, the assumption is that "it can be established what COTS middleware products are available" can be based on:

- Those Vendors having a Web home page, and if not,
- assuming that, due to a lack of time and resources, any other products are not important enough to pursue.

3.4.1.1 How the Product Search Was Conducted

The COTS available middleware product search sequence included the following steps:

1. Define the application requirements.
2. Evaluate data provided by previous middleware survey report(s).
3. Group the available middleware by functional category.
4. Group the available middleware by specific service.
5. Match the service provided with the GS requirements.
6. Evaluate Vendors:
   - Vendor reputation?
   - Market share?
   - Probability of product survival/ dominance? Market momentum?
   - Meets standards?
7. Evaluate Vendor's products.

The COTS available middleware product search looked at: Representative Middleware Vendors, Products, Platform(s), Database Support, Capability Highlights, and Price structure.

3.4.1.2 What Products Were Investigated, and Why?

The COTS available middleware product investigation included the following considerations:

- Attempt to assess the strengths and weaknesses of the COTS available products in the marketplace.
- Look for Vendors that provide an integrated solution of middleware development, runtime, and management environments, where possible.
- Look for COTS truly "off-the-shelf" or "shrink-wrapped" solutions, i.e. look for solutions requiring no, or little development by the User's IS Support staff.

3.4.1.3 Available Middleware Software Packages, Selection Criteria

Guidelines for deploying middleware are (see for example [14]):

1. Keep it simple: Reduce complexity, do not add to it. Does it cut application development time? Is it transparent to the user?
2. Focus on needs, not technology. Does it integrate legacy applications in an easy manner? Does it run on a variety of platforms, including yours?
3. Understand your technology bias.

4. Decide on what your middleware must do before deploying. Does it improve the performance of your application?
5. Allow for new technologies: Choose Vendors with clear strategy for integrating new technologies. Can it grow with your needs?
6. Choose platforms with the most flexibility: Avoid middleware Vendor lock-in. Is it standards compliant?

The selection of the middleware was thus based on:

1. GS application requirements.
2. The existing GS systems environment.
3. COTS product availability.
3. Cost:
   - Unit cost?
   - Run-time fees?
   - Annual support/licensing fees?
   - Training costs?
4. Development requirements:
   - Development required by the user?
   - Language? C-like or Basic-like?
   - Development platforms supported?
5. Deployment platforms supported?
6. Databases supported?
7. Implementation expertise, provided and needed:
   a. The software supplier must have middleware technology application expertise.
   b. The software supplier must have a middleware support infrastructure.
   c. Expertise needed by user? Programming, development, power users, end-users?
8. Can the solution be easily implemented, within cost/schedule constraints?
9. Can the solution be successfully tested?
10. Expected future GS needs.

3.4.2 Applicable For GS Middleware Implementation?

A single all-encompassing, common API does no cover all the GS requirements discussed in Section 2. Therefore, a middleware "solution" must consist of a set of APIs and functions. implemented using a 3-level C/S architecture and using OOT:

3.4.2.1 *Category 1: Distributed Processing GS Middleware Implementation?*

- Distributed objects.
- CORBA compliance.
- Both, *UNIX* and *Microsoft's* Windows NT *OSs*.
- Secure Internet and Extranet interface.

3.4.2.2 *Category 2: Distributed Data Access GS Middleware Implementation?*

- Multiple relational servers, with object extensions.
- CORBA compliance.
- ODBC compliance.
- Secure Internet and Extranet data access.

3.4.2.3 *Category 3: Distributed Systems Management (DSM) GS Middleware Implementation?*

- Open platform with 3rd party management applications.
- CORBA compliance.
- SMTS compliance, for legacy systems.

4. MIDDLEWARE TECHNIQUES AND STANDARDS

4.1 Middleware Techniques [8]

In the Enterprise arena, middleware uses either the well-established message-based approach (for example, *IBM's MQSeries*) or the new object-based approach using *ORBs* (for example, *ORBs* that are compatible with the CORBA/IIOP standard). *IBM*, as a further example, also bridges these two middleware approaches with its *Component Broker ORB.*

4.1.1 Lower-Level Middleware Techniques

Referring to Figure 1-1 (b), the higher-level middleware services products, described in 4.1.2, incorporate (make use of) the lower-level middleware techniques described in the following paragraphs, and compared in Table 4-1:

4.1.1.1 RPC

Remote Procedure Calls (**RPCs**) allow you to send calls, as opposed to data, to another computer.

A client (application program) issues a request in the form of a RPC to execute a procedure on a remote server system. This is a synchronous communication technique, in that the application waits for a response before proceeding. It can be thought of as a subroutine call, where the client (application program) requests a procedure to be performed (e.g. opening a file) on the remote(file) server. The RPC usually includes a specification for exchanging the arguments and results between the client and server, i.e. parameters can be established for the remote procedure.

Examples are: *OSF's* Distributed Computing Environment, **DCE/RPC**, *Sun's* Open Network Computing, **ONC/RPC** and Network File System (**NFS**) file access, and *NetWise's RPC Tool.*

4.1.1.2 MOM

Message Oriented Middleware (**MOM**) routes messages (data, control information, or both) between applications. Messages can be sent either in a "conversational" synchronized delivery mode, or the messages may be queued, which provides asynchronous delivery. Message queuing means that the sending application's message is posted to a queue for delivery to the receiving application, and the sending application does not need to wait for a response before continuing processing.

MOM technology is usually built on RPC facilities.

Examples are: *IBM's MQSeries*, *DEC's MessageQ* (the product line has been acquired by *BEA Systems, Inc.*), and *Covia Technologies' Communication Integrator.*

4.1.1.3 ORB

A mechanism for locating objects across language (*C, C++* and *Ada*), and location (network) boundaries, *Object Request Broker* (***ORB)*** technology is usually built on either RPC or MOM facilities. An RPC calls a function. With an *ORB*, the call is for a function (defined as "**method**") within a specific object. The *ORB* software intercepts the client call, finds an object that can implement the call request, passes parameters to the object, invokes the method, and returns the results. The *ORB* thus "brokers" inter-object calls.

Objects communicate across a network using "**messages**." The messaging can be RPC or MOM based, or direct via a network transport.

The two competing *ORB* standards are *OMG's* CORBA (which provides operating system independence), and *Microsoft's* COM (or DCOM, for distributed objects). *ORB* quality depends on the product implementation. *ORB* examples are: *DEC's ObjectBroker* (sold to *BEA*), *IBM's SOM*, and *Iona Technologies' Orbix.*

Table 4-1. A Comparison of Middleware Techniques

Category 1: Distributed Processing Middleware			
	RPC	**MOM**	***ORB***
Strengths :	For synchronous communications/services. For more homogeneous application integration. Faster, vs. MOM. Included in NOSs *OSF's* DCE RPC is the standard, and is supported by *Microsoft's Windows.* synchronization is not	For asynchronous communications/services. For more heterogeneous application integration. A way to tie legacy systems to C/S, since no constraints are imposed on the structure of an application. Fault-tolerant, in the form of message queues. Essential if C/S transparency. possible/desirable.	Layers object-oriented features on RPCs. Provides more-sophisticated services. CORBA's IDL provides a well-defined interface, providing portability and interoperability for objects. Application language independent. Provides local/remote Allows self-describing of object services. Security is built-in. Allows for dynamic discovery of objects. By extension, is the "new NOS."
Weaknesses:	RPCs from different Vendors have different APIs, and don't integrate, leading to Vendor lock-in. RPC coordinating code (stubs) must be available for both client and server applications. Server must keep up with clients.	Messaging products from different Vendors have different APIs . Slower, vs. RPC. Not included in NOSs.	Method resolution performance may not scale. May not have MOM functionality.

Table 4-1. A Comparison of Middleware Techniques (Continued)

Category 2: Distributed Data Access Middleware		
	Remote data access	**TP Monitors**
Strengths:	<u>*SQL* APIs</u>: The industry standard for access to relational databases. <u>*SQL* gateways</u>: Provide heterogeneous RDBMS connectivity.	Provide cross-platform transaction management (control) for a distributed OLTP environment: <u>Process management</u>: Perform shared load balancing by managing priorities of server data requests. <u>Transaction management</u>: Guarantee transaction integrity.
Weaknesses:	<u>*SQL* APIs</u>: Differences in *SQL* syntax and semantics among RDBMS. Application developers must incorporate Vendor-specific APIs. Some Vendors support *ESQL*, some *CLI*. FAP/stack support may vary. <u>*SQL* gateways :</u> Too slow for OLTP applications. Not a seamless data access method.	All applications must adhere to the TP Monitor protocol.
Category 3: Distributed Systems Management (DSM) Middleware		
	Middleware run-time monitors (object monitors)	
Strengths :	Robust runtime environment provided by CORBA *ORB* Object Services.	
Weaknesses:	Very small installed base (1997).	

4.1.2 Higher-Level Middleware Techniques

The higher-level middleware service products hide some of the complexities associated with the lower-level middleware techniques described in 4.1.1. These consist of more familiar tools such as DCE services, NFS services, and *SQL* access of relational data bases, products such as *IBM* acquired *Transarc's* TP Monitor *Encina,* and standards such as *Microsoft's* ODBC and *Borland's* IDAPI.

The ***MAGIC*** system (see 2.2) used **"little COTS,"** i.e. functional pieces such as a *GUI* and an *Expert System*, in its design, as opposed to **"big COTS,"** which is defined as "packages that provide many integrated functions." This is NOT to be confused with using higher-level middleware techniques, which can still be looked on as "components."

4.1.3 Other - C/S - Techniques

Application development today, in addition to the traditional LAN-centric RPC and message-based methods (e.g. OO-Interprocess Communication (**IPC**)), needs to examine the use of application development tools that are used for producing Web content. Two technologies available are *Java Applets* and *ActiveX Controls.*

4.1.3.1 Java Applets

Java Applets are software components, written in *Java,* the component development language developed by *JavaSoft*, an operating company of *Sun Microsystems. Applets*, at the present time (1997) are used mainly for enhancing HTML content on Web pages. *Applets* can be down-loaded into *Java*-compatible *browsers*, allowing, for example, the incorporation of dynamic visual effects within a *HTML* page. See for example [11].

Java is supported by the most popular *browsers*: *Netscape Navigator* and *Microsoft Internet Explorer*. The browser executes the *applets,* using a *Java* interpreter.

Java and *Java applets* are portable, and supported on the major client and server *OSs*. *Java* licensees (e.g. *Netscape, Microsoft, IBM, HP,* Apple) are embedding *Java* into their *OSs*. They also provide platform-dependent adapter software between the *Java Virtual Machine* (and the *Java Class* Libraries) implementation of *Java* and their respective *browsers.*

Java provides various levels of protection against computer viruses for its *applets*. When the *applets* are downloaded they are run through a *Java* "verifier" before the code is passed to the-*Java* interpreter (or a just-in-time compiler, or a full compiler).

JavaSoft has also released a *Java Beans* API, which defines a set of interfaces for creating reusable components. The components can be hooked together using visual development tools, such as *Borland's JBuilder, IBM's VisualAge, SunSoft's Java Workshop*, or *Symantec's Visual Cafe*. [13] A *Java Beans Bridge* is also provided, to connect to *ActiveX Components* – allowing OLE applications to use the *Java Beans* components.

In the GS environment, we can use, or develop, CORBA-augmented *Java* applications by the use of CORBA/*Java ORBs* (e.g. *JavaSoft's Joe). JavaSoft* is also expected to provide a *Java* Interface Definition Language (**IDL**) for the *Java* Developer's Kit (1997), which would let CORBA (non-*Java)* applications use *Java* objects.

4.1.3.2 ActiveX Controls

ActiveX Controls are software components, written in *Microsoft's Visual C++. ActiveX Controls* can be down-loaded, and typically run inside *browsers.*

ActiveX is supported by the most popular *browsers*: *Netscape Navigator* (presently, 1997, by plug-ins) and, of course, *Microsoft Internet Explorer*. Unlike *applets, Controls* are downloaded as binaries.

ActiveX is *Microsoft's Windows OS* -based. (*Microsoft's* efforts to port ActiveX to other platforms has not met with acceptance at this time, 1997.)

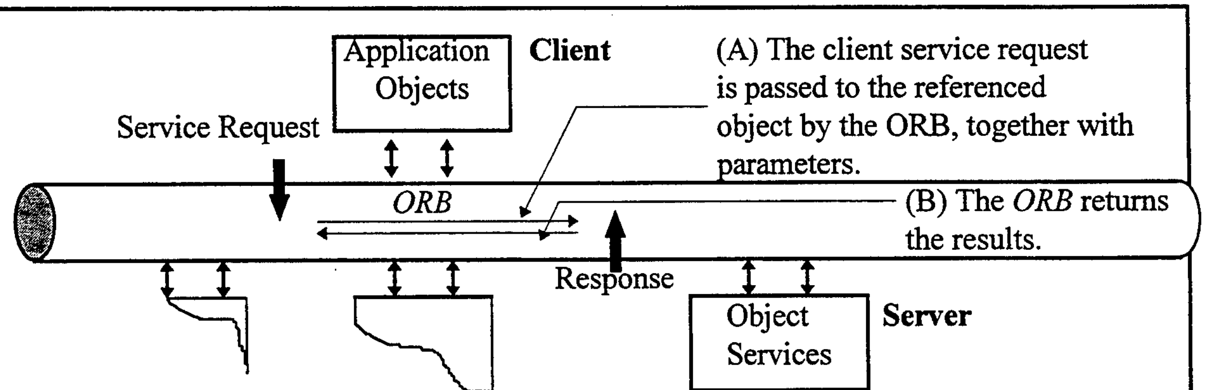
ActiveX programs have one advantage over *Java*, they can access files and perform other desktop functions. However, since *ActiveX Controls* have access to system resources, this present a security problem. Authentication technology is required to restrict downloading of *Controls*. (Keys for certifying software are managed by *VeriSign Inc*., of Mountain View, CA, a spin-off of encryption developer *RSA Data Security*. [13]) Physical encryption, however, is still the only method to obtain a higher level of security.

Java may be the choice for the public side of the enterprise (or GS) firewall. *ActiveX* may be the choice inside the firewall.

Each *Control* is a self-registering DCOM object.

4.2 (Emerging) Middleware Standards

To achieve a Client - Middleware - Server architecture - which has a lifetime beyond any specific product - requires that the architecture is based on standards. The Object Request Broker (***ORB***) object communication concept provides a standard that specifies the procedures required to locate an object, locally or remotely, invoke it, and communicate with it. The *ORB* is defined in terms of its interfaces, and an *ORB* may be client, server, or Operating System based. Reference Figure 4-1.



The *ORB* is responsible for all the mechanisms required to find an object implementation over the network, prepare it to receive the request, transmit the request, and return the outcome of the request to the client.

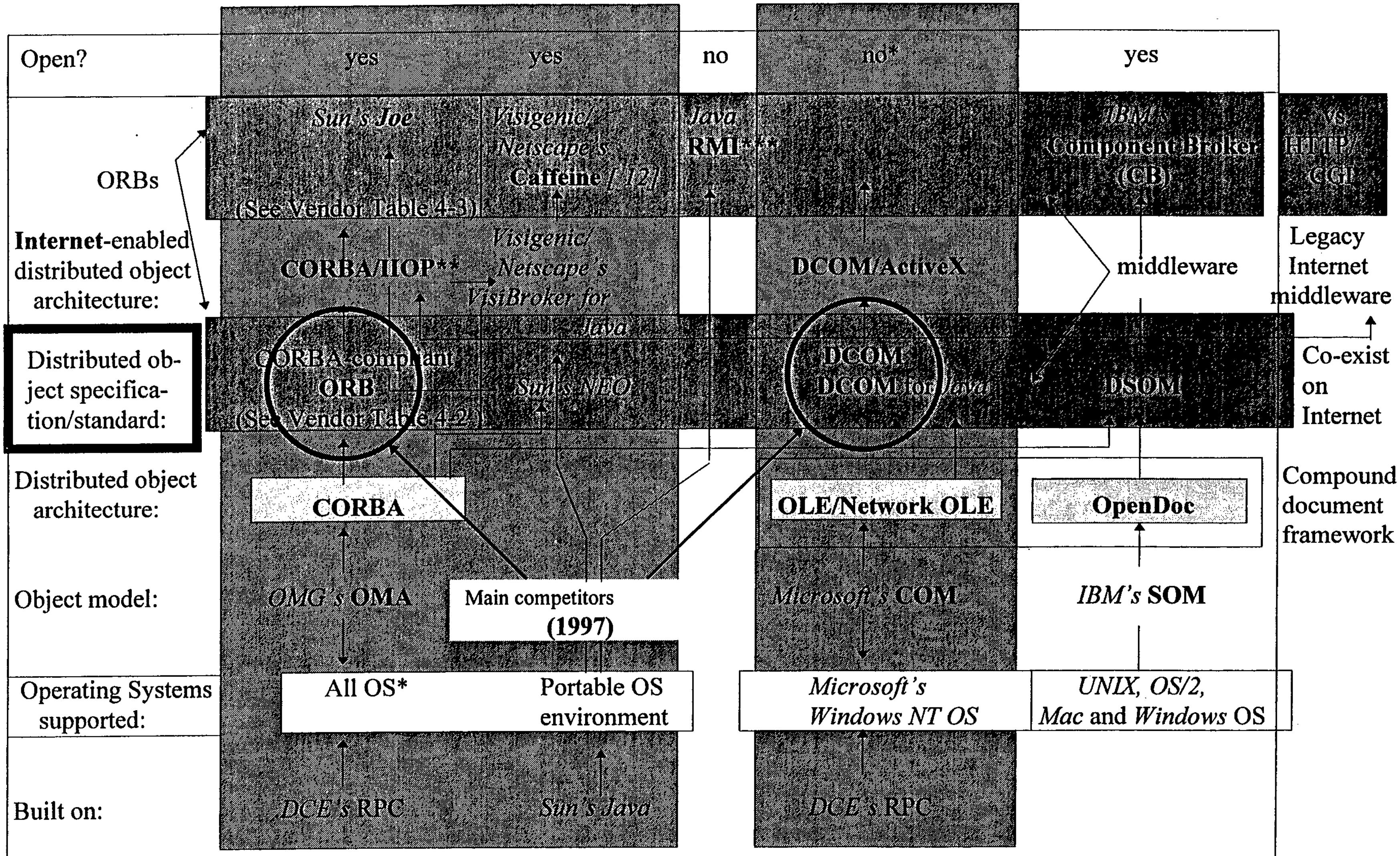
Figure 4-1. The Object Request Broker (*ORB*)

The three (3) competing object-oriented application development *ORB* standards (see 3.3.2) are based on (1) the *Object Management Group's (OMG)* Common Object Request Broker Architecture (**CORBA**), which is supported by more than 100 Vendors and end-user companies, including *IBM Software Solutions*, *Netscape Communications*, *Oracle*, *Sun Microsystems*, and *ORB* makers *Visigenic* and *Iona Technologies*, on (2) *Microsoft's* Object Linking and Embedding (**OLE**) technology, the CORBA direct competition, and which has been established as a "de facto" standard, and on (3) *Component Integration Laboratories (CIL)* **OpenDoc**, which in turn, represents direct competition to OLE as a compound document specification. [11]

The competing technologies on the Web are **CORBA/Java** and **DCOM/ActiveX**.

The competing object models and standards for writing distributed applications are shown by Figure 4-2, and for distributed database access by Figure 4-3.

Some COTS-available CORBA *ORBs* are listed in Tables 4-2 and 4-3.

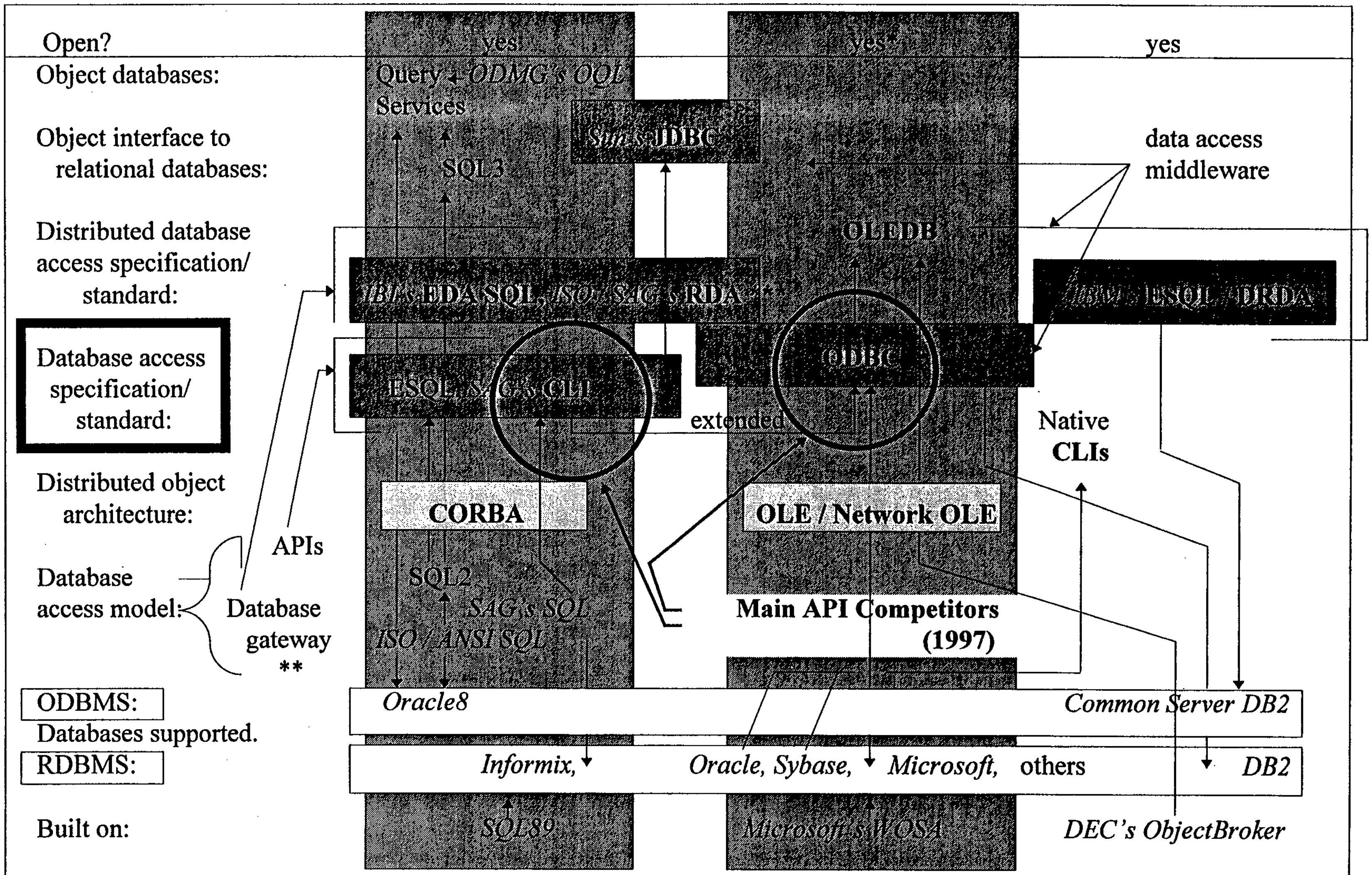


* *Microsoft* is working on making its technology open (1997).

** CORBA 2.0

*** *JavaSoft* is working on CORBA compatibility (1997)

Figure 4-2. The Competing Object Models



* Since DBMS-specific drivers are needed, may not be portable. ** Interoperability standards for common / universal data access.

*** Minimal market share.

Figure 4-3. The Competing Database Access Models

Table 4-2. Some CORBA 2.0-Compliant *ORBs*

Vendor	Product	Description	More Information
DEC	*ObjectBroker ORB*	Runs on *OSF/1, AIX, HP-UX* and *Sun OS.*	Sold to *BEA*. See Table 7-1
Expersoft	*CORBAplus*	For *C++*.	
HP	*ORB Plus*	Founded on DCE RPC.	
IBM	*Component Broker (CB)*	*C++* or *Java ORB.*	
Iona	*Orbix*		
Oracle	*Web Request Broker*	Framework. Uses IIOP.	
Visigenic/ Netscape	*VisiBroker*	For *C++* or *Java.*	

Table 4-3. Some CORBA/*Java ORBs*

Vendor	Product	Description	More Information
Sun	*Java ORB, Joe*	Supports client-side *Java* objects. Downloaded along with an *applet* to (or residing in) a client. Used to invoke server objects using the IIOP protocol.	[12] Called an *ORBlet*. See Table 7-1.
Iona	*C++ ORB, OrbixWeb*	Supports client-side *Java* objects. Runs on *UNIX, OS/2, NT*, etc.	
Visigenic/ Netscape	*VisiBroker for Java*	Supports both client and server side *Java* objects.	
Expersoft	*CORBAplus, Java Edition*	Also CORBAplus, ActiveX Bridge.	

4.2.1 Vendor Neutral (Open) Middleware Standards

"Open systems" standards provide insurance of being able to change/mix Vendors as the market and technology advances dictate:

4.2.1.1 Standards Organizations/Vendor Consortium Middleware Standards

Object-specific middleware is referred to as "CORBA" or "OLE." The terms are applied to any software built under the *OMG* OMA or *Microsoft* COM paradigms, respectively:

4.2.1.1.1 CORBA

OMG's Common Object Request Broker (CORBA) architecture defines a standard way for distributing objects across multiple platforms and operating systems. CORBA provides a specification for development of an *Object Request Broker* (*ORB*). The *ORB* is a messaging facility used to establish communication between distributed objects.

CORBA describes the architectual structure of the *ORB* and its components. It provides an application language-neutral *Interface Definition Language (IDL)* that is used to define object interfaces. *IDL* includes the capability to implement both static (stub-compiled) and dynamic (immediate passing of a request) interfaces to the inter-application request handling software "bus" called the *ORB*. *IDL* APIs can be invoked from *C*, *C++*, *Ada*, *Smalltalk*, *COBOL* and *Java*. *IDL* grammar is a subset of *C++*.

The two key CORBA architecture elements are the *ORB* and *CORBA services*. The *ORB* allows objects to transparently make service requests to, and receive responses from, other, locally or remotely located objects. *CORBA services* extend the *ORB* capabilities. A main object service is the *Persistence Service*, which defines a standard interface for components stored on different servers, such as Object Database **(ODBMS)**, Relational Database **(RDBMS)**, or file servers. CORBA object services thus provide an approach for creating built-to-order middleware.

CORBA provides run-time metadata, in a *CORBA Interface Repository*, that describes the functions a server provides and their parameters. Clients use the metadata to discover the available services at run time. A CORBA system is thus stated as being "self-describing."

CORBA does not follow the document-centric metaphor of OLE and OpenDoc.

CORBA 1.1 specified the IDL, language bindings, and APIs for interfacing to the *ORB*. The actual implementation of an *ORB* was left to the *ORB* Vendor. See Figure 4-4.

CORBA 2.2 specifies interoperability across Vendor *ORBs*, by use of the Internet Inter-*ORB* Protocol **(IIOP)**. IIOP is TCP/IP, with CORBA-defined message exchange added. A CORBA-compliant *ORB* must thus implement IIOP, or provide a bridge to it. IIOP supports an interface to OLE. See previous Tables 4-2 and 4-3 for CORBA-compliant *ORB* Vendors.

CORBA/IIOP is today the distributed object standard for both *Java* and the Internet.

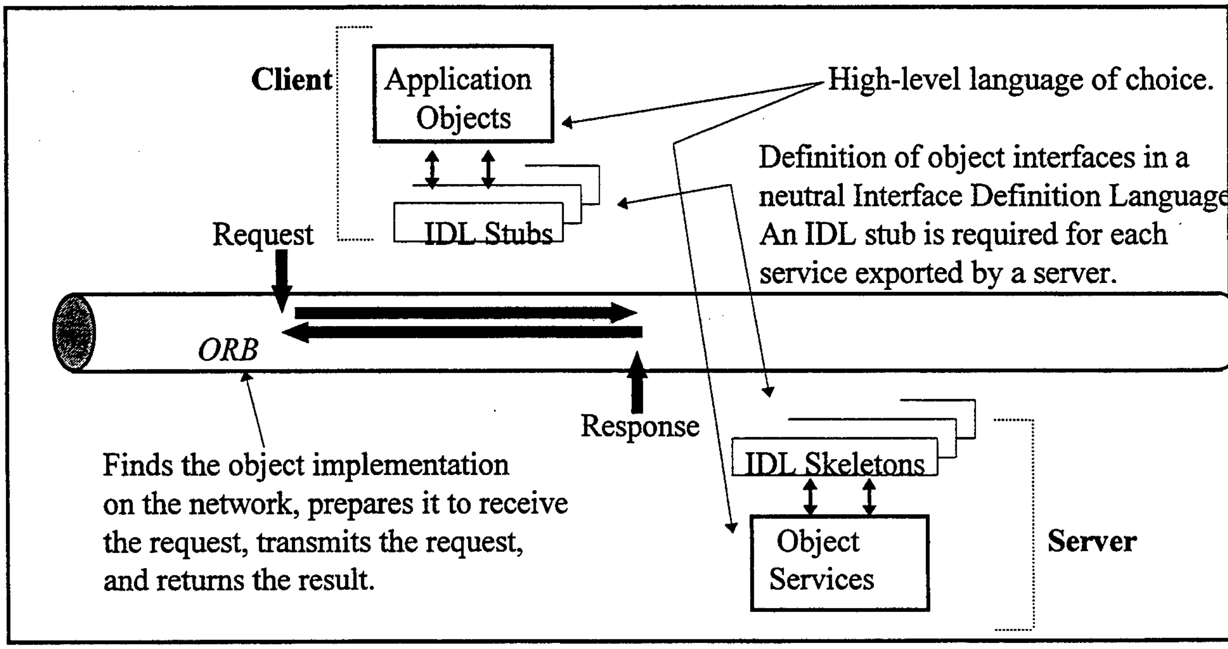


Figure 4-4. CORBA IDL Bindings

4.2.1.1.2 OpenDoc

Component Integration Laboratories' (CIL) OpenDoc specification lets users create complex documents (video clips, e-mail messages, word processor files, etc.), and was conceived as an "open" alternative to OLE. It provides a **framework** (provides a model and defines the rules) for building components that can be integrated into compound documents.

A part of the OpenDoc architecture is the *Open Scripting Language (OSL)* API set, which supports application-independent scripting. This allows the coexistence of scripting languages such as *IBM's REXX, Lotus' LotusScript* and *Microsoft's Visual Basic for Applications (VBA).*

OpenDoc is based on *IBM's* System Object Model (**SOM**) framework for building and packaging language-independent run-time class libraries. (*IBM's* Distributed SOM (**DSOM**) adds CORBA compliance via a set of IDL-defined class libraries to SOM, to extend the method invocation mechanism to allow construction of distributed object applications.)

OpenDoc, (as does OLE with ActiveX, and also *Java Beans*), provides a compound document framework. The components can be packaged as DLLs.

OpenDoc extends the CORBA middleware architecture to the desktop. It uses CORBA as it's object bus. It allows creation of a C/S system based on components. Desktop applications (e.g. front ends) can be populated with active components found on the desktop, OR on servers on the network.

4.2.1.1.3 SQL

SQL (Structured Query Language) is the only means of providing access to data in a relational database, or **RDBMS** (Relational Database Management System). *SQL* may be "embedded" (***ESQL***), (i.e. embedding *SQL* statements within the programming language), as defined by the *SQL-92* standard, or "callable" at run time (***CLIs***).

SAG** (X/Open SQL Access Group)* has defined a common (Vendor-independent) API set for *SQL* databases, called the *X/Open CLI*. This standard was extended by *Microsoft's Open Database Connectivity **(ODBC) standard (see 4.2.1.2.1).

If a client is interested in "the relationship between server objects," an object extension on top of the *SQL* database would be required. However, due to the more complex processing, the response may be slow. A "query optimizer" (indexing, to represent the relationship among several database objects, or cost estimation to schedule query operations), or a true ODBMS (Object Database Management System) are the answer. The ODBMS stores the data and the links between the data in the same format, without being "flattened" to a tabular structure of the RDBMS.

The *Object Database Management Group **(ODMG)*** industry consortium's ODMG-93 specification provides an ODBMS standard, and also an *Object Query Language **(OQL)***, which is based on *SQL3*. However, since the major database Vendors are extending their relational database investment to handle objects, the ODBMS is not yet COTS wide-spread (1997).

Since each Vendor's database has a unique set of extensions to *SQL*, the database Vendors first support their native CLI API sets, and then, as an option, provide support for standard *CLIs*, such as the ODBC API.

Examples of native CLIs are: *Oracle's Call-Level Interface **(OCI)***, *IBM's ESQL/DRDA*, *Sybase's Sybase Open Client*, and *Microsoft's SQL Server* (which is ODBC).

See 3.1.2.2 for ODBMS Vendors.

4.2.1.2 De Facto Middleware Standards

If a standard sees prevalent use, it is considered a "de facto" standard:

4.2.1.2.1 ODBC

Microsoft's Open Data Base Connectivity (ODBC) is the *Windows* API standard for *SQL*. It is an extension of the *SAG CLI*. It is an ubiquitous data access standard, for connecting to DBMSs, since it allows access to multiple databases using a common API set. Most database Vendors support the ODBC API calls, in addition to their preferred native *SQL* APIs, as well as ODBC drivers for their respective DBMS servers. It has "pass-through" functionality.

However, the specification is controlled by *Microsoft*. Also, as ODBC drivers may be developed by a 3rd provider, quality may be an issue. It is a C implementation of the *SQL CL*, and provides a procedural API interface to relational databases.

4.2.1.2.2 Joe

Joe is *JavaSoft's* CORBA IIOP *ORB* written in *Java*. It can be downloaded or bundled with the *Java* runtime environment, to run *Java* applications under CORBA. It is thus also designated as a CORBA/*Java ORB.*_Other examples of CORBA/*Java ORBs* are *Iona's OrbixWeb* and *Visigenic's VisiBroker for Java.*

4.2.1.2.3 JDBC

JavaSoft's Java Database Connectivity (**JDBC**) provides for an ODBC-like *SQL* database access interface. As is ODBC, it is based on the *SAG CLI.* It consists of a set of *Java* classes. It is a part of *JavaSoft's* JDK.

It is a <u>native *Java* implementation of the *SQL CL*</u>, and provides an object interface to relational databases.

For example, *Visigenic's VisiChannel for JDBC*, with embedded *VisiBroker ORB*, conforms to the ODBC standard, and provides for cross-platform database access. It can access any ODBC data source, using the IIOP protocol and the appropriate ODBC database drivers supplied with *VisiChannel*. See Figure 4-5.

4.2.2 Proprietary/Vendor-Unique Middleware Standards:

4.2.2.1 DCOM (or Network OLE)

Microsoft's Object Linking and Embedding (**OLE**) architecture may be considered as a de facto standard for building OO applications. It provides the *Microsoft Windows* for the desktop, and *Microsoft Windows NT* for servers, operating systems with OO capabilities.

The OLE standard is based on *Microsoft's* Component Object Model (**COM**) architecture. COM defines how individual applications interoperate under the *Windows OS* (i.e. it defines how to connect software components). It is based on the *DEC* RPC. It was originated as a **compound document** standard, and thus provides a compound document framework.

OLE provides a method for linking or embedding one kind of object into another, typically called the "container." (A **container** is a component that can embed other components.) For example, an *Excel* spreadsheet (a component) can be linked or embedded into a *Word* document (the container). OLE also provides the means for one application to control another, by the use of scripts. For example, *Excel* figures can be consolidated into a new chart before displaying the data in a *Word*
document.

Distributed COM, or DCOM, is COM with RPCs. DCOM lets COM objects, such as OLE and *ActiveX Controls*, interact over a network.

Digital Equipment Corporation (DEC) and *Candle Corporation* have been working with *Microsoft* to add interoperability between *DEC's* ORB, *ObjectBroker*, and OLE. [11] This would support OLE applications to run on UNIX, since *ObjectBroker* runs on *AIX, HP-UX* and *SunOS*.

Vendors building OO *Windows* applications can use *Microsoft's* OLE Software Development Kits.

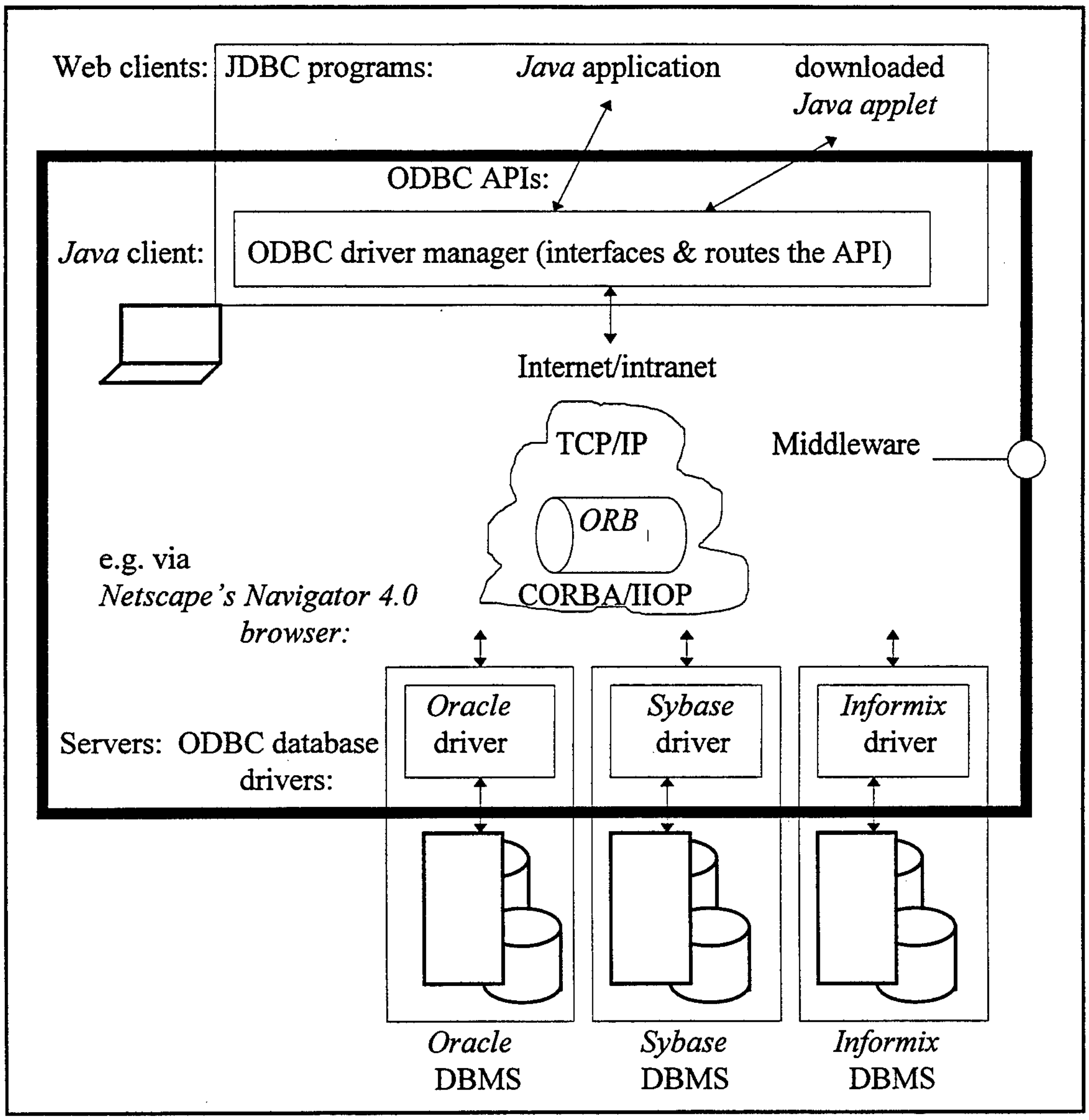


Figure 4-5. Using ODBC Middleware For Database Access Over the Internet

4.2.2.2 DRDA

IBM's Distributed Relational Database Architecture (**DRDA**) is *IBM's* **FAP** (Formats and Protocols, for interoperability). It is used in *IBM's DB2* family of databases, and is supported by all the major database Vendors.

DRDA defines the protocols for C/S interactions, with the goal of providing an interoperability standard for heterogeneous database environments.

4.2.2.3 HTTP/CGI

HTTP/CGI is the Web's end-to-end C/S protocol. **HTTP** (Hypertext Transfer Protocol) and **CGI** (Common Gateway Interface) are the Web's legacy communication techniques.

HTTP is the Web's *"RPC"* on top of TCP/IP. It is used to request and retrieve URL-named resources. CGI is a real-time back-end program, residing on the Web server. It executes the Web server received HTTP request and passes the results (builds a dynamic Web page) in HTML format back to the Web server for return to the client.

4.2.2.4 RMI

Remote Method Invocation (**RMI**), a part of *JavaSoft's* JDK, is a *Java* interprocess protocol that makes CORBA transparent to *Java* programmers. It supports remote method invocation on objects across *Java Virtual Machines.* An RMI invocation passes a local object's state by value (i.e. by copy, instead of reference), as a parameter inside a message. It passes remote objects by reference, like CORBA.

RMI provides new interfaces and classes for remote objects.

Due to being proprietary and not being able to invoke objects written in another language, either *Joe* (*SunSoft's* CORBA IIOP *ORB*) or *Caffeine* (Visigenic/Netscape's, with an RMI-like programming environment, CORBA IIOP *ORB*) are expected to replace RMI in the *ORB* competition. [11]

The middleware communication standards are compared in Table 4-4.

Table 4-4. A Comparison of Middleware Communication Standards

Object-Oriented Standards			
	CORBA/IIOP	**DCOM**	**OpenDoc**
Strengths:	There are basically 2 standards for distributed objects: CORBA and DCOM. Both separate the object interface from the implementation. They are OO switching mechanisms vs. OpenDoC, which is a document-centric programming model.		
	The industry standard *ORB*, an open *software bus.* desktop. Components can inter-operate.	The de facto standard *ORB.* Extends CORBA to the	The compound document standard.
Weaknesses:	Small installed base (1997).	Specification is controlled by *Microsoft.* Only runs on *Microsoft Windows OS* platforms.	
Internet-Savvy Standards			
		CORBA/Java	**DCOM/ActiveX**
Strengths:	Portable. Proven inter-operability . accessible by a *Java* program.	DCOM IDL defined interface definitions are OpenDoc containers.	*Java* provides the *Beans* that will fill the
Weaknesses:	Small installed base (1997).	Does not run on *UNIX OS* platforms.	

4.3 Other - C/S - Standards

The leading candidate for a de facto standard for open and distributed computing has been *Open Software Foundation's (OSF)* (now known as *The Open* Group) Distributed Computing Environment (**DCE**). DCE provides an open Network Operating System (**NOS**) architecture for developing applications in a heterogeneous C/S environment. The main goal of DCE has been to link geographically distributed (enterprise) C/S systems.

In comparison, CORBA (and OLE) create object interfaces on top of a NOS. CORBA is a standard that provides an "architecture" for an *ORB*. DCE is a standard that provides a "product," consisting of APIs and services:

4.3.1 DCE

Among the "products" provided by DCE, and which is incorporated into most operating systems, is DCE's Distributed File System (**DFS**). DFS establishes a single-system-image for a distributed file system infrastructure. DFS specifies a method for multiple users (clients) of a (file server) file system to access and modify the same data via a single network ID and network password, for "single-sign-on" capability. Thus, DFS also implements access security and protection.

An important aspect of DCE is its name and directory services, which, by using a set of attributes to describe a (network service, computing platform, etc.) directory entry, **provides information location independence**. The directory service API is based on the *X/Open* foundation's X/Open Directory Services (**XDS**) API specification, which provides resource location transparency.

Other DCE's suite of functions includes the DCE RPC, threads, and time services. DCE comes bundled with all major server platform operating systems (e.g. *IBM's AIX* and *Sun's ONC+)*, and its functions are also incorporated in *Microsoft's Windows.*

DCE DFS, developed by *Transarc Corp.* a subsidiary of *IBM*, provides interoperability with the current de facto standard for accessing remote files - *Sun's* Network File System (**NFS**), by the use of gateways that allow NFS clients to access DCE DFS servers. Thus a migration path exists, if one is needed, from NFS to DCE DFS.

DCE has been extended to include the distribution of information via the Web (see *Gradient Technologies Inc. WebCrusader*). However, integration of DCE features into application development tools for the GUI environment such as *Microsoft's Visual Basic* and *Powersoft Corporation's Power Builder*, or in *Sun's Java* component development language, or in *Microsoft's ActiveX Controls* components, is still to be accomplished (1997).

4.4 Internet-Specific Middleware Standards (See 4.2.2.3)

4.5 Distributed Systems Management (DSM) Middleware Standards

The competing system management models and standards <u>for managing distributed objects</u> are shown by Figure 4-6.

4.5.1 DSM Vendor Neutral (Open) Middleware Standards

Open frameworks for systems management are required

4.5.1.1 Standards Organizations/Vendor Consortium Middleware Standards:

4.5.1.1.1 SNMP, SNMPv2, etc.

The *Internet Engineering Task Force's (IETF)* Simple Network Management Protocol (SNMP), and its derivatives, is the established standard for managing TCP/IP networks. SNMP proposed the concept of a central management station, which communicates with a number of managed devices throughout a network, the *agents*. It is a reporting mechanism that queries compatible *Management Information Base (MIB)* databases that store information about a resource that needs to be managed, defined as a "managed object." It actually consists of three elements: An asynchronous request/response protocol (User Datagram Protocol/ Internet Protocol - **UDP/IP**), plus the Structure of Management Information (**SMI**), which defines the data types, notations and naming conventions used to specify the managed objects, and the *MIB*. The *MIB* consists of hierarchical databases distributed across managed stations.

SNMPv2 extends SNMP, and attempts to correct some of the perceived deficiencies of SNMP. For example, a SNMP node, under SNMPv2, can be both a "managing node" and a "managed object." This provides for a "manager-of-managers" concept for distributed systems management.

4.5.1.1.2 RMON, RMON-2

The Remote Monitor (**RMON**) standard extends SMTP by defining a number of new "managed objects," for example, to report on ENET or Token-Ring LAN statistics.

RMON-2 provides standardization to RMON's *MIBs*. It also adds an *OSF* model layer 3, the network layer, *MIB* definition - for an end-to-end view of the network.

4.5.1.1.3 DMI, DMI 2.0

The *Desktop Management Task Force's (DMTF)* Desktop Management Interface (**DMI**) is an open standard for managing all the components on a PC, Mac, or workstation. The managed components include hardware, the applications, and the operating system. *DMTF* members include *IBM*, *Microsoft*, and *Apple*. A DMI interface allows the desktop resources to be managed by SNMP, CMIP, or CORBA-based management applications. A *Managed Interface File (MIF)* file, similar to a *MIB*, stores descriptions of the managed devices.

DMI 2.0 defines a standard method for transmitting management information across a network using *ORBs*.

4.5.1.1.4 XMP, XOM

X/Open has two standards for distributed systems management APIs: X/Open Management API (**XMP**), which defines a set of *C* API calls for managing system-to-managed system communication, and X/Open Object Manager (**XOM**), which defines the handling of the data structures defining the managed objects. XMP and XOM work with SNMP and CMIP.

4.5.1.1.5 DME

The Distributed Management Environment (**DME**), is an *OSF*-proposed standard as a "total solution" for network and systems management. DME has an object orientation and provides for object wrapping of a management resource. It is built on CORBA services, and the use of a

CORBA-compliant *ORB* called a *Management Request Broker (MRB)*. The CORBA IDL definitions have been enhanced to allow encapsulation of SNMP or CMIP functions. The DME CORBA framework has found its way into the marketplace via *IBM/Tivoli's TME* (*Tivoli Management Environment*), available on *UNIX OS platforms* (see 4.5.1.2.1).

4.5.1.2 DSM De Facto Middleware Standards

Some distributed computing system management standards are used worldwide, and are supported by many 3rd-party software Vendors:

4.5.1.2.1 CORBA - TME

A popular enterprise C/S system management solution is provided by the CORBA-based *Tivoli System's Tivoli* and its system management platform *Tivoli Management Environment (TME 10)*, which is built on the *ORB*-based *X/Open* Systems Management Reference Model.

4.5.2 Other - C/S - Systems Management Standards

Market share dictates acceptability of a standard:

4.5.2.1 CMIP

OSI's Common Management Interface Protocol (**CMIP**), for manager/agent, and also manager/manager communications, has not caught on. It was deemed as "too complex to implement" and has been broadly replaced by SNMP.

The middleware for systems management standards are compared in Table 4-5.

Table 4-5. A Comparison of Middleware for Systems Management Standards

Distributed systems management standards			
	SNMP	**RMI**	**TME**
Strengths:	The industry standard for managing TCP/IP networks.	Defines new managed objects.	The industry "de facto" standard for systems management.
Weaknesses:	Good for simple network management, but needs to be replaced by RMI (or CMIP) for more complex systems diagnostics and higher *OSI* model layer data collection. Could be replaced by CORBA, as exemplified by TME.		
Object-oriented distributed systems management standard			
	CORBA		**TME**
Strengths :	CORBA has defined a Systems Management Facility for managing distributed objects.		The industry "de facto" standard for object management.
Weaknesses:	The solutions are enterprise-level.		

4.6 *Groupware-Specific Middleware Techniques (For Reference)*

The e-mail APIs are the cross-platform *Lotus's* Vendor Independent Messaging (**VIM**) standard, and *Microsoft's* Messaging API (**MAPI**) standard. Both address e-mail on the PC Desktop. Both are supported by most Vendors. These APIs allow ("mail-enabled") applications to access the mail messaging infrastructure (transport services, directories, and stores). MAPI, due to the dominance of the desktop by *Microsoft,* is generally the most popular today (1997). MAPI messages can include data attachments and OLE objects.

Mail applications are categorized as "front-end," and the mail infrastructure as the "back-end." They are connected using a C/S model. E-mail is thus the "middleware." (Groupware, however, does not have the distributed object bus infrastructure of CORBA.)

Several standards have been defined for interconnecting e-mail mailboxes. The "mail backbone" server-to-server standards are the "de facto" standard developed for the Internet, for TCP/IP networks, the Simple Mail Transport Protocol (**SMTP**), and the international standard X.400. Both are supported by most Vendors, although SMTP is the most popular. **MIME** (Multipurpose Internet Mail Extension) extends the SMTP standard to handle non-character mail attachments.

Another groupware standard deals with Electronic Data Interchange (**EDI**). This has been implemented by the government, for example, for exchange of technical product documentation with contractors.

4.7 *Data Warehousing-Specific Middleware Techniques (For Reference)*

The data warehouse is a database. Generally, clients invoke remote procedures that reside on the warehouse server, which then execute as transactions on the server's database. Support is provided by the traditional RPC, MOM and TP Monitor standards. Relational/OLAP tools are used to retrieve data from, and to add data to the data warehouse. A form of middleware used here is called "copy management/data replication middleware" and is used to copy databases or parts of databases to the data warehouse. Various database specific techniques are used to "refresh," "update" and "cleanup" the data.

Other than using low-level metadata format standards, data warehouse technology is relatively new, and being database centric (as opposed to TP Monitors, which tie together clients and servers), there has not been a need to develop data warehouse-specific standards (1997).

4.8 *Transaction Processing-Specific Middleware Techniques (For Reference)*

Transaction Processing Monitors (**TP Monitors**) guarantee delivery of each transaction: Every transaction goes through, it does not go through more than once, and your computer informs you if a transaction fails for any reason. TP Monitors are usually used in a synchronous mode, i.e. the computer waits for each transaction to be completed before it begins another.

A main standard that specifies how a TP Monitor interfaces to a resource manager (e.g. a DBMS) is provided by ISO's OSI: The communication uses the *X/Open* transaction interface protocol

(**XA**) together with the underlying OSI TP standard, to define the "two-phase commit" protocol to synchronize transactions on different nodes/platforms.

X/Open's Distributed Transaction Processing (**DTP**) Reference Model specifies the FAPs that provide for multiple applications and resources. The DTP FAPs works together with the underlying OSI FAP, and support multiple transport protocols.

4.9 Other - Space Data Systems - Standards

Vendor		Product	Description	Web page/Information Comments

4.9.1 CCSDS

The *Consultative Committee for Space Data Systems* ***(CCSDS)***, an international organization of space agencies, is chartered to develop standard data handling techniques to support space research. Its recommendations are forwarded to *ISO* for adoption as *ISO* standards. The CCSDS protocols have been adopted as the standard for space data packet transmission across serial links. Users include *NASA,* the U.S. military, and commercial space missions. For example:

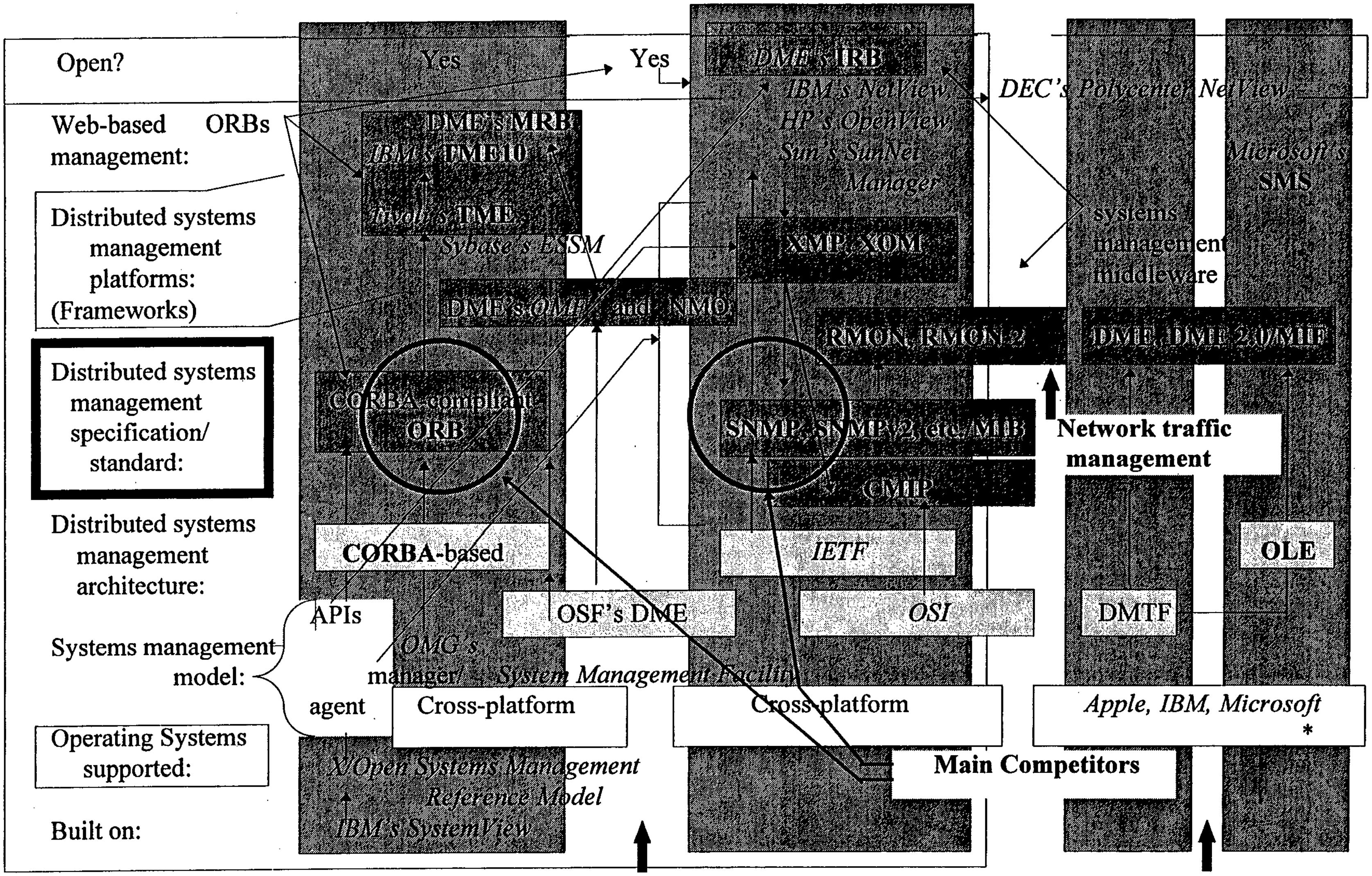
TSI Telsys		Telemetry Ground Station Communication Systems	Based on CCSDS protocols	http://www.tsi-telsys.com 7100 Columbia Gateway Drive Columbia, MD 21046 (410) 872-3900

4.9.2 *STK*

Analytical Graphics, Inc.		*Satellite Tool Kit (STK 4.0)*	Satellite system analysis software. (888) ASK-4STK	http://www.stk.com 660 American Ave. King of Prussia, PA 19406

4.9.3 *SCL*

Interface & Control Systems, Inc. (ICS)		*System Control Language (SCL)*	For developing real-time monitor and control applications. Includes Real-Time Engine. The *Naval Research Laboratory (NRL)* is standardizing on *SCL* as a COTS TT&C software system.	http://www.sclrules.com 8945 Guilford Rd. Suite 120 Columbia, MD 21046 (301) 596-2888



* Also PC adapter Vendors.

Figure 4-6. The Competing Systems Management Models

5. ARCHITECTING THE NEXT GENERATION, COMMON SATELLITE GROUND STATION

5.1 The Critical "Applications"

While the **peer-to-peer computing model**, where all participating systems are equals, and can request and provide services to and from each other, or the **client/network/server model**, where any client can establish a session with any server, over a network, without pre-arrangement, are desired goals, today's (1997) model is standard **client/server**. That is, the clients and servers are specialized. This C/S model, the "second generation" of C/S computing, has dedicated servers for applications, data, systems management, etc. Thus, most middleware products available today (1997) require, and assume, a well-defined C/S environment.

The typical computing application can be divided into the following components: Presentation processing, (for example, a GUI), business processing and data manipulation (the application programs), and database management system processing (performed by a DBMS).

In terms of 3 functional levels, "the computing components" can be thought of as: The user interface, the business logic, and the shared data.

The typical GS application includes a data-gathering (hardware-interface) function, the operator-interface/data-display function, and a recording (data archive) function. The computing may be distributed, with the functions separated, and each running in a separate computer, connected by a network. The data-gathering function almost invariably implies the use of a computer which includes specialized hardware and software. The data display function may include the use of custom analysis tools and display screens. Data display manipulation may occur locally, or the data may be passed to standard programs located on their own platform for processing.

In terms of 3 functional levels, "the GS computing components" can be thought of as: The data display/ operator interface, the custom application function, and the archive facility. Any two of the 3 functions, if they are distributed between a client and a server, and must work together, are considered to be two parts of the same "task" or the same "application." An application programmer must design the functions to work together over a network, using the "technology platform" hooks provided by the communication protocols (such as *IBM's* request-reply LU 6.2 protocol, or a procedure-oriented protocol such as RPCs), and by the operating system. Thus the GS software structure consists of two major divisions: The **applications** and the **infrastructure**. The "infrastructure" comprises the design of the network and the attributes of the physical nodes (the hardware), and has, as its primary responsibility, the assurance of data flow reliability.

"**Infrastructure software**" comprises the software components that allow the two parts of the "application" to communicate. The infrastructure program is also divided into two parts, bound to the two halves of the application, respectively.

The "applications" then view the "infrastructure" as the means of moving data and messages between the two separated halves of the application. The infrastructure provides the networking and communication between the two halves of the application, and isolates the application from the detailed workings of the C/S environment. If the infrastructure is standards-based, the autonomy of the application software is maintained as a completely separate entity.

The "infrastructure software" is called "**middleware.**" See Figure 5-1.

The functional levels, or the two parts of the "application," communicate with each other by passing messages to one another via the infrastructure. The application calls a function supplied by the infrastructure software. The infrastructure packages the call in a network packet, provides the appropriate addressing, and transmits it to the infrastructure program in another computer. There, the infrastructure calls the message-processing function in its half of the application, passing the message. This "**message-passing**" communicates data and control information (data packets, streams of data, *SQL* strings, images, and so on) between the distributed functions. For example, the application-infrastructure message may say: "Find my other half and connect to it....," and "Tell my display half to display this message...," etc.

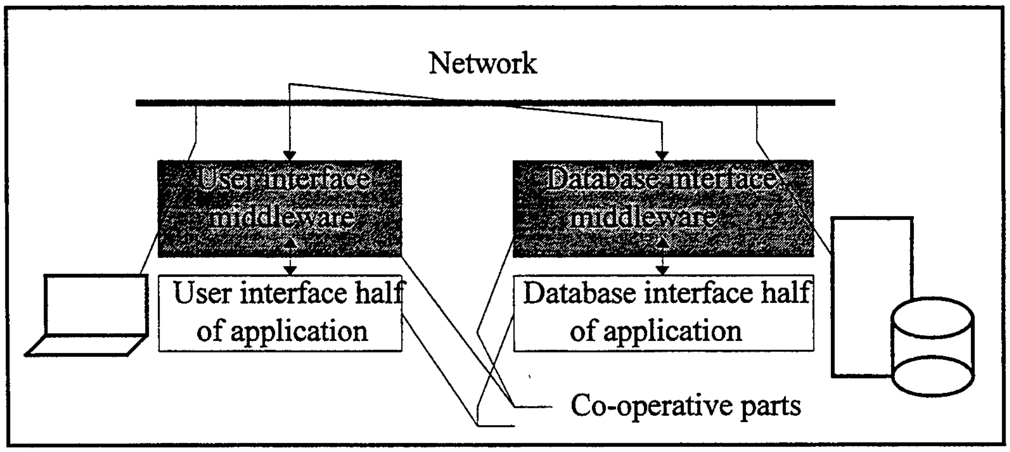


Figure 5-1. Relationship Between the Application and Middleware

5.2 *The Client/Server (C/S) Computing Environment*

The C/S services are distributed across multiple systems, in a cooperative computing environment. This environment can be described in terms of 2 or 3 levels (or tiers) of platforms (the hardware), or in terms of the distribution of the applications (the software). A 2-level C/S hardware partitioning is shown in Figure 5-2 (a), for reference. The 3-level C/S hardware partitioning (also called "multilevel"), is the prevalent C/S model today, and is shown in Figure 5-2 (b).

Since the C/S services are actually performed by the applications software, the application split between the client and server can provide the best differentiation. The [**3-level functional C/S model**], as discussed previously in 5.1, is shown by Figures 5-3 (a), (b), and (c).

5.2.1 2-level Architecture

A 2-level C/S system links a PC client directly to a server. This architecture is based on the client-file server and client-database server models. The application logic generally resides either either in the client, in the server, or both.

5.2.2 3-level Architecture

The **[3-level platform-based C/S model]** is characterized by inter-operation dependence on the platforms used. For example, clients are designed to interact with server DBMSs of a given type. Abstraction is lacking - and is gained by using the **[3-level functional C/S model]**, which allows implementation of applications that "**scale**" and are "**open**."

"Open" applications are independent of database management systems, communications software, operating systems, and windows managers. This "independence" is achieved by the use of standards-based middleware. For example, data has location transparency, and the client (user) sees a single system image. Objects can be used to integrate complex data types.

The 3-level C/S, "second generation," (function-based) system more closely represents the distributed cooperative processing environment. The application logic generally is separated from the user interface and the data, and resides in a middle layer. Examples of the 3 functional levels are provided by distributed objects, the data warehouse, and the Web.

The clients (level 1) perform presentation services (e.g. they provide the GUIs); the application servers (level 2) off-load common (work group) functionality applications from the clients; and the data servers (level 3) focus on providing data-related services. This eliminates the "fat client" (see 5.2.3) and improves scalability and manageability, and allows heterogeneity. "Scalability" allows multiplexing for concurrent users, and the addition of more servers for more users. "Management" of re-configuration and updating is made easier by code isolation. And, support of mixed processing environments ("heterogeneity") is possible, by allowing the use of designated, and specialized, application servers.

As the complication increases, when going from a 2-level to a 3-level architecture, additional functionality is required to make the distributed applications work together. This assurance of cooperation is provided by a middleware architecture component. The 3-level model is expanded to a multilevel model, which has clients, application servers, database servers, <u>and</u> middleware. This **[multilevel functional C/S model]** addresses the distribution and placement of the application components, and becomes a **[client-middleware-server model]**.

Middleware provides:

- Local/remote transparency.
- De-coupling of applications from specific database formats.
- Connection of disparate systems without the need to write application code.
- An application plug-and-play environment.
- A standard interface for communication, event handling, and other services within the network.
- Ability to incorporate data from various sources.
- A self-describing system (CORBA).

Middleware also provides (continued):

- A bridge for the gap between C/S and mainframe environments (e.g. a mainframe can be used for legacy data storage)
- Message routing.
- Security: User authentication, data encryption.
- Transaction recovery.
- Directory services.
- Integrity: Reliability in communications.

Middleware thus provides for quick application development cycles and a consistent application environment. A key principle of 3-level design is to encapsulate the "business logic" (i.e. the application) on the middle level. The business logic can thus be easily changed as requirements change.

The application server can be a separate server. Or, the application can reside on the same server as the database, i.e. the 3rd level.

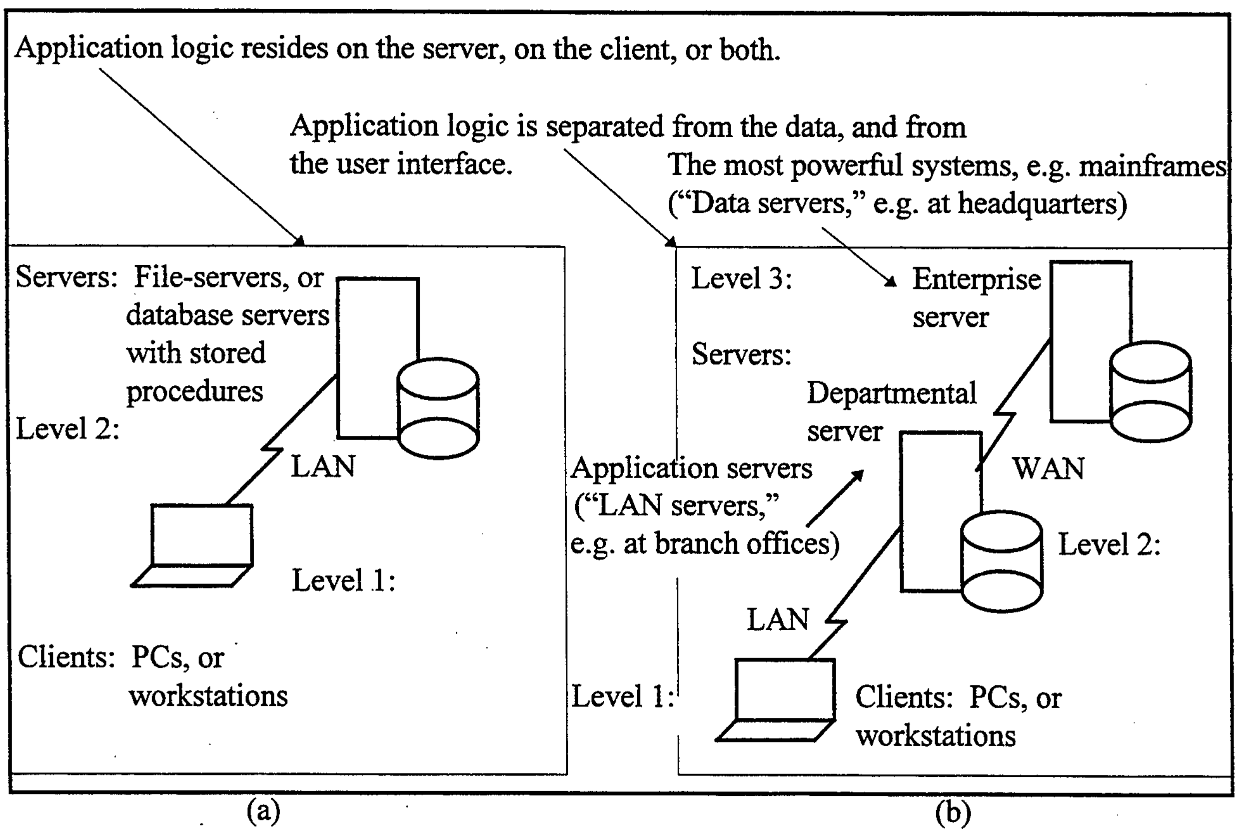


Figure 5-2. (a) 2-level C/S Hardware Partitioning, and (b) 3-level C/S Hardware Partitioning.

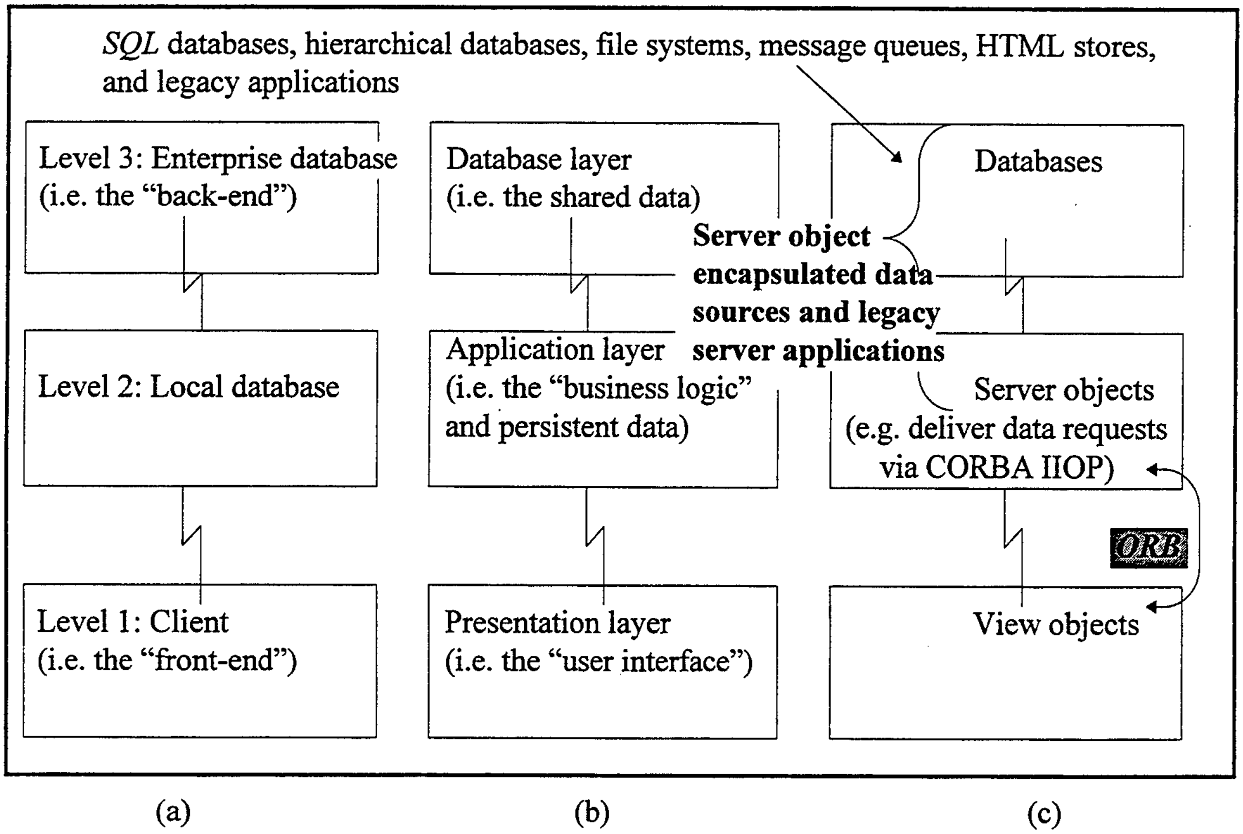


Figure 5-3. 3-level Functional C/S Architecture: (a) Database Model, (b) Partitioning Model and (c) Object Model

The advantages of using a 3-level C/S architecture, consisting of clients on the first level, application servers in the middle, and database servers forming the third level, are:

- It is easier to leverage microcomputer computing power.
- Facilitates a standard user interface (GUI) to the enterprise.
- It is easier to deploy new application functionality .
- Modification and expansion are simplified.
- It is easier to accomodate new controls, such as TP-Monitors.
- Isolation of applications and databases facilitates their management.

To "Webify" the 3-level C/S architecture, the levels are defined as: The end user with his *browser* on the first level, a Web server in the middle, and back-end applications and databases as the third level. The second level typically consists of *Windows NT* or *UNIX* servers, with *Java*-based middleware (1997) connecting to the back-end. See Figure 5-4.

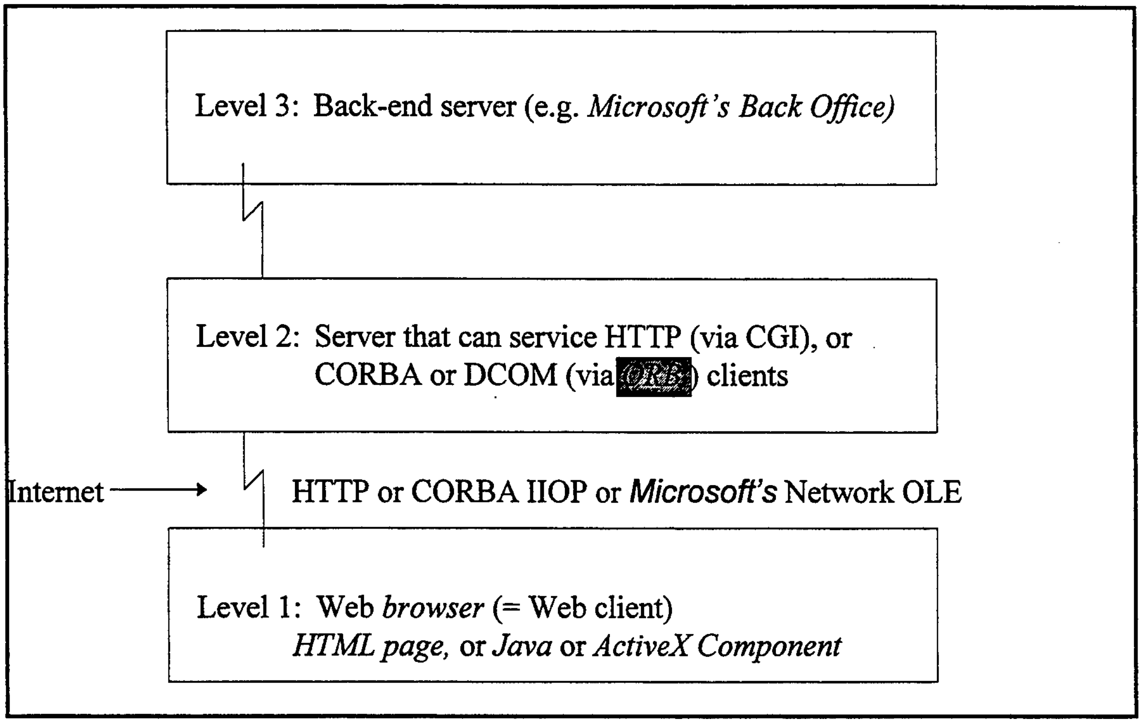


Figure 5-4. 3-level Functional C/S Architecture: Web Model

5.2.3 Clients

Clients and servers can be referenced to as "thin" or "fat." This has very little to do with the hardware, but rather with applications and application positioning. In **thin clients** the application logic resides mostly on the server (and hence we have a "fat server"). In **fat clients** the application logic resides mostly on the client.

"Fat client" examples are provided by the traditional PC, which interfaces to file servers and relational database servers. "Fat server" examples are object database servers and Web servers. A "fat PC" is thus a PC that can act as both client and server.

5.2.3.1 Network Computer?

Total Cost of Ownership (**TCO**) considerations invite looking at Network Computers (NC), or "thin clients." Three varieties of this less expensive alternative to the PC have been proposed: The *Java*-based **NC**, the *Windows*-based **NetPC,** and **hybrid** X-window/*Windows* computers. The hybrid version is X-window technology based, with X-windows servers, or a 3rd party connection to *Windows NT* servers. Applications are executed on the server, with the display sent to the client. Due to graphics speed limitations, for example, it has not had much user support. [14] The choice narrows to:

(a) *Java* thin clients (a *Sun - IBM - Oracle - Netscape* initiative), for example, *Sun's JavaStation,* or

(b) *Windows* thin clients (a *Microsoft - Intel* initiative).

The **NC** has a CPU, but is diskless. It requires lots of local memory, a reliable network, and server power.

The **NC** is a *Java* PC, or *Java* thin client. It downloads and runs the *Java OS* (or *browser, HotJava)*, and downloads *applets* to perform a task. Optionally, a proprietary minimum-capability OS can be downloaded, to download and run existing applications, along with terminal-emulation software. The download includes a *Java Virtual Machine (JVM)*, which contains the *Java Interpreter* and possibly a *Java Just-In-Time (JIT) Compiler*, for running *Java* applications. The NC is considered to be "client oriented."

The **NetPC** is not a true NC. It is a leaner version of the PC, or a "simplified desktop." The NetPC is a *Windows* PC. The *Windows 95 OS* is downloaded over the network, along with the application (or *browser*), and *ActiveX Controls*. With plans to access multi-user *Windows NT*-based applications running on a server, with only the graphics display sent to the client and rendered, the NetPC is considered "server oriented."

The installed base at this time (1997) is minimal. The desirability of installing NCs is an open question.

The choice between installing NCs vs. PCs is driven by the number of users, and the application base. NCs may be considered if the number of users is large, and the lower cost, and centralized administration and application management, are cost reducing considerations. For example, with NCs only the server needs upgrading when installing new applications.

With thousands of desktops the NC easily wins on a TCO basis. The NC can be a dumb terminal replacement, linked to *UNIX* servers or IBM mainframes. Or, the NC can replace PCs running custom task-oriented (e.g. data entry) applications, where access, rather than local processing of information is desired. The NC usage is "low-end," where a general-purpose computer is overkill.

The PC wins if :

(a) speed is needed at the desktop, (e.g. for display-intensive applications requiring instant icon-click response, for application load, or for extensive graphics), or

(b) a lot of hard disk space is required at the desktop, or

(c) administrative and user learning is to be avoided (for including NCs in the system).

If use of a NC is to be seriously considered, questions such as: "Is *Java* fast enough?"; "Is the lack of available *Java* applications a problem?"; "Is ease of administration a major requirement?"; "Is the reliance on the network a robustness issue?"; "Is the reliance on a server an issue?"; and "Will the NC have market staying power?" need to be addressed.

In the GS environment, with a limited number of workstations, the flexibility and control gained from using **PC clients** is warranted. NCs, or NetPCs, are not a consideration.

5.2.3.2 The Standard Desktop

Most of today's (1997) stand-alone desktops belong to *Microsoft's Windows 95* or *Windows NT Workstation.* The competition comes from *IBM's OS/2 Warp Connect OS*, (on the corporate desktop), and *Apple's Mac OS*, and potentially *SunSoft's Java OS*, which would download *applets* over the Web to perform a task.

Microsoft's DCOM standard is supported by *Windows NT Workstation 4.0*, and is proprietary, but so ubiquitous on the desktop it can be considered a de facto standard.

5.2.4 Servers

If mainframe computing power is not a requirement, price/performance advantage dictates the use of **PC servers** in the C/S environment that have the ability to scale.

At the low-end of price/performance the competing server operating systems are *Novell's NetWare OS*, *IBM's OS/2 Warp Server OS*, *Microsoft's NT Server OS* (which has additional features over *NT Workstation)*, and the *SCO* and *Sun's Solaris UNIX OS's*. *UNIX* cluster and *RISC OS's* that provide parallel computing are found at the high-end.

PC server microprocessor platforms include *Intel, DEC's* Alpha, *IBM's* Power PC, and *SGI's* MIPS. (*OS/2's* platform is *Intel.*)

5.3 A COTS Standard-Based Distributed Object Architecture

The typical GS TT&C application has:

3 downlink functions,	TT:
	◉ Data gathering.
	◉ Operator interface/data display.
	◉ Recording.
and 1 uplink function:	C:
	◉ Commanding.

Copies of infrastructure programs (i.e. the middleware) are expected to be identical for communication between all components of each function for the common GS.

5.4 Network Implementation

The *NOSs* are *Novell's NetWare 4.1*, *IBM's OS/2 WarpServer*, and *Microsoft's NT Server 4.0*. *UNIX* has two main *NOSs*, *Sun's Solaris* network services *ONC+*, and *OSF's* DCE. *ONC+* also supports DCE. (DCE is actually "incorporated," as opposed to being a stand-alone *NOS*.)

5.4.1 *LAN*

The operating systems have been extended to include (bundle) network services and are now really "*NOSs*," which in turn have been extended to handle objects under CORBA (and OLE). Most *NOSs* use *RPCs*. *MOM* is not supported. An example of integrating messaging (MOM) functionality within CORBA is provided by *Expersoft's CORBAplus ORB, Enterprise Edition* (see Table 6-1).

5.4.2 *WAN*

Remote access to data is an accepted requirement, and must be provided for.

5.4.3 *Internet*

Component security is a major issue:

An example of component virus checking is provided by *Seattle Software Labs' WatchGuard Security System.*

Virus scanning plug-in for *Microsoft's Proxy Server* is provided by *Trend Micro Inc.*, of Cupertino, CA. Also, firewall Vendors such as *Check Point Software Technolgies, Raptor Systems, Trusted Information Systems, Milkyway Networks*, and *DEC* are expected to incorporate gateways that look for *ActiveX* and *Java* viruses. [13]

5.4.4 *Intranet*

The **intranet** is a private (company internal) IP network.

A good example is the *IBM INN (IBM Information Network),* which connects, vial leased telephone lines, the *IBM* employees spread out around the world.

5.4.5 *Extranet*

The **extranet** is a selective extension of the intranet, with selective access provided to business partners and customers. In the GS environment, selective access (for example, "read-only" access to the telemetry data archive) may be provided to a *satellite constellation user community.* The extension may be via a dedicated T-1 or T-3 Internet connection, and provided by an Internet Service Provider (**ISP**). The ISP must provide information security, and guaranteed port availability, network latency and up-time. Or, the extension may also consist of a **mobile network**.

An example of an extranet is the linking of a health care provider consortium with hospitals, doctors, and pharmacists. In the GS environment, some specialized processing tasks could be off-loaded or accomplished at other sites.

The ISP can also provide access to the corporate intranet for mobile users, by making available dial-in ports, with authentication, and data encryption via an encrypted IP tunnel.

Security could be provided by using DCE services (see 4.3.1).

Examples of "**mobile (or wireless) middleware**" are provided by *IBM's Advanced Radio Communications on Tour (ARTour)* middleware, and *Ericsson's Virtual Office (EVO)*. [19][20] *IBM* uses an RS/6000 platform as a gateway to a wireless service provider. The clients for this *AIX* server can be either *Windows* or *OS/2* radio modem equipped laptop computers. *ARTour* provides network connection at the TCP/IP sockets level, where as *EVO* provides connection for data access (ODBC) or mail-enabled (MAPI) services at the API level. *EVO* is optimized for *Microsoft's Office*. Both support communication protocol resolution for access to the different wireless networks, and encryption. *ARTour* also supports user authentication. Both support wireless, dial-up and LAN connections.

5.5 Robustness Implementation

A computing system is designated as "**robust**" if it exhibits hardware and software **fault tolerance**, and thus has high **availability**, and is easily expanded in storage capacity, the number of supported users, etc., and thus has high **scalability**.

In the GS environment:

- **Scaleability** is used to customize the GS environment.
- **Extensibility** addresses multiple and new missions. For example, OOT is easy to extend.
- **Portability** is needed among different C/S platforms, and middleware and network protocols

5.5.1 Fault Tolerance

Fault tolerance enables computing services to continue when there is a failure. Hardware fault tolerance is implemented by disk mirroring, the use of RAID, and hot and warm standby servers.

Software fault tolerance, in a C/S environment, is implemented by components that are based on a **distributed object model**, that promotes object location transparency. If a server goes down, processing is transferred to a new, state-cognizant, server. Middleware detects the loss of a component, and provides automatic switch-over to the new component. For example, fail-over capability is provided by replicated Name Servers.

Software fault tolerance is also gained by following established design guidelines, such as incorporation of protection against errors. For example, a program should protect against incorrect user inputs, validate arguments, use dynamic memory allocation (to avoid fixed limits), and have the capability (based on debug level) for debugging the code. Execution profile tools should be available, for reporting statistics and for performance monitoring.

5.5.2 Scalability

Scalability is achieved by adding or using the required computing resources as needed. A server should be able to handle an increase in the number of users, in the number of applications, and in the size of the database. A first step usually is to add more memory and/or disk drives, or to upgrade to a faster CPU.

Horizontal (or External) Scalability:
As work-load increases other servers may be added. This is defined as **clustering.** Server availability is increased by implementing "fail-over clustering." In the event of a server failure, processing is shifted automatically to a second node. See Section 7.5.1 for a discussion of the additional system administrative software support required.

Vertical (or Internal) Scalability:
As users are added to a server, performance may be improved by adding processors. If the processors are not pre-configured (i.e. are available to run tasks as needed) this is defined as **symmetrical multi-processing (SMP).** Scaleability here depends on the software's multithreading capability, as well as on the hardware. The operating system must have the capability to direct processes to different CPUs, and the application (e.g. the DBMS) must be coded to allow instructions to be run as multiple threads. Operating systems that support multiprocessing are: *IBM's OS/2 for* SMP, and *Microsoft's Windows NT.*

Some *UNIX* versions can scale efficiently (i.e. can show substantial improvement in performance - of the order of 1.6 x to 2 x - as each processor is added) to 32 processors. In contrast, *Windows NT* currently has been shown to scale well to 4 processors. [5]

For a file server application, such as transaction processing, the sub-second response time requirement to a request depends more on the I/O than the CPU speed. An SMP server may thus only provide a negligible performance improvement.

For a database server application a dual processor "super-server," such as *Compaq's* Proliant, may be used, to provide the same performance for growing data sets.

Scalability is also helped by the use of server systems based on the *Intel* Pentium Pro processor, which is optimized for SMP. [6].

For the GS environment, there is a need for fail-over operation. Since a telemetry data archive server CPU is not expected to be compute-bound, a server cluster with automatic fail-over is the choice.

Also, different computing resources may be needed for different user environments, such as at the desktop, workgroup or the enterprise levels. An application should be able to migrate (scale up) between a variety of hardware and operating system configurations.

Application scalability:
Application scalability includes being able to handle application growth, in terms of the number of processes involved, rate of message traffic, and the number of involved platforms.

5.5.3 Extensibility

Software is termed **extensible** if functionality can be added, such as being able to automatically adjust to take advantage of added resources, or new objects can be added, with minimum code modification. In the GS environment this means that the GS can be easily expanded to handle new satellite constellations/block releases (using OO models to abstract functionality from the implementation). [B]

In the distribution of applications using OO techniques, this means we have the ability to dynamically add or change implementations of objects and methods.

5.5.4 Portability

Middleware should be **portable**, in that it should be capable of running on more than one computer system, and under more than one OS.

5.5.5 On-line Administration

Administration operations, such as backup and recovery, system monitoring, and database reorganization, should be performed without taking the server down.

5.6 Data Access and Legacy Application Migration and Porting (See 3.3.1)

5.7 Systems Management (See 8.)

5.8 Security

It is assumed C2-level [12] security is adequate for the GS.

Both *Windows NT* and the standards-compliant versions of *UNIX* have C1- and C2-level security certification. *DISA's* approach to security for DMS (see 2.3) is to apply either the international messaging standard **X.400** for high-security text communications, or a commercial standard in use today, lacking a high level of assurance, such as the **Simple Mail Transfer Protocol (SMTP)** where possible. Three (3) options are planned, depending on the level of security required: Software encryption, smart card tokens, and the Fortezza cards. [3]

Three (3) WAN alternatives are available for establishing a secure link for GS data access by remote users:

1. Use a dedicated leased (T1) line.

2. Use the Internet, with a firewall (see 3.2.3.2).

3. Create a Virtual Private Network (VPN) by establishing a secure tunnel across the Internet.

6. MIDDLEWARE VENDORS & PRODUCTS *

6.1 Available COTS Middleware, By Functional Category & Specific Service

🕯 Evaluation ranking (see Section 3.4 for Evaluation Criteria), 3=first, 2=second, 1=third.

Table 6-1. Middleware Vendors

Vendor	🕯	Product	Description	Web page/Information/Comments

Category 1: Distributed Processing Middleware

The leaders in robust scaleable COTS middleware are *IBM* and *Microsoft.*

Distributed applications:

IBM		*OS/2 Warp Server*	OS integrated middleware, with planned OpenDoc support and integrated *Java*-language capabilities.	Successor to LAN Server.
	🕯🕯🕯	*Component Broker (CB)*	Distributed application development environment, *CBToolkit*. R untime environment, *CBConnector*, with CORBA-compliant *C++* or *Java ORB* component. With a system management component, *CB/SM.*	http://www.software.ibm.com *CB* provides the infrastructure for the design, implementation, deployment, and management of object applications.
Sun Microsystems, Inc.	🕯	*Solaris NEO 2.0*	Networked object extension for the *Solaris OS.* CORBA 2.0-compliant.	http://www.sun.com 901 San Antonio Rd. Palo Alto, CA 94303 (800) 821-4643
		Solstice NEO	Bundled with *NEO*, supports *Java* graphical tools.	
Microsoft Corp.	🕯🕯	*Windows NT Server 4.0*	Adds network OLE (DCOM) support.	

*** Vendor product and service names may be trade marked or service marked.**

6.1 *Available COTS Middleware, By Functional Category & Specific Service (Continued)*

Table 6-1. Middleware Vendors (Continued)

Vendor	🕯	Product	Description	Web page/Information/Comments

Category 1: Distributed Processing Middleware (Continued)

***UNIX* and *NT* coexistence on the same network :**

The leading *UNIX* Vendors are *Sun, IBM* and *HP*. Challengers are *DEC, NCR, SNI, SGI, Sequent, Unisys, ICL* and *Bull*. The leaders of Wintel platforms are *IBM, HP, Compaq, NCR* and *Dell*. Challengers are *Digital* and *DG*.

Vendor	🕯	Product	Description	Web page/Information/Comments
IBM	🕯🕯		Has CORBA 2.0-compliant middleware that runs across Microsoft *Windows NT*, and *OS/2 WarpServer*, *AIX* and *OS/390* platforms.	
DEC		*AllConect*	For *HP-UX* and *NT* coexistence.	
HP	🕯	COM/CORBA proposal for *UNIX* System/ *NT* integration	DCOM/CORBA bridging technology, in agreement with Microsoft.	http://www.hp.com 3000 Hanover St. Palo Alto, CA 94304-1181 (800) 752-0900
SCO		*Vision97*	Provides Microsoft *Windows Desktop* access to any *UNIX* server.	http://www.sco.com 425 Encinal St. Santa Cruz, CA 95061 (800) SCO-UNIX
Data Focus, Inc.		*NuTRACKER* porting kit	Allows porting of *UNIX* applications to *Windows NT*.	http://www datafocus.com 12450 Fair Lakes Cr. Suite 400 Fairfax, VA 22033 (800) 637-8034
		NuTRACKER interoperability	Allows *NT* workstations to plug seamlessly into *UNIX* environment, and run X-Window-based applications.	

6.1 *Available COTS Middleware, By Functional Category & Specific Service (Continued)*

Table 6-1. Middleware Vendors (Continued)

Vendor		Product	Description	Web page/Information/Comments
Category 1: Distributed Processing Middleware (Continued)				
***UNIX* and *NT* coexistence on the same network (Continued) :**				
Sun Microsystems Inc.		*NEO Connectivity*	For *Microsoft Windows*	Integration between applications running on *Windows* desktops and *Solaris NEO* applications on the back-end server.
Softway Systems, Inc.		*OpenNT*	A native *UNIX* (emulator) environment on *Windows NT.*	http://www.softway.com 185 Berry St. Suite 5514 San Francisco, CA 94107 (415) 896-0708
RPC:				
OSF		*DCE/RPC*		(617) 621-7300
Sun Microsystems, Inc.		*ONC/RPC*		
NetWise		*RPC Tool*		http://www.netwise.net 10284 Page Ave. St. Louis, MO 63132 (314) 423-4855
NobleNet, Inc.		*NobleNet RPC 3.0*	Automatically generates the C/S network *C* source code for all program data structures and APIs. (A compiler tool kit that can distribute APIs across a network.)	http://www.noblenet.com 337 Turnpike Rd. Southboro, MA 01772 (800) 809-8988

6.1 Available COTS Middleware, By Functional Category & Specific Service (Continued)

Table 6-1. Middleware Vendors (Continued)

Vendor		Product	Description	Web page/Information/Comments

Category 1: Distributed Processing Middleware (Continued)
RPC (Continued):

Seer Technologies, Inc.		*NetEssential*	Middleware communications services. Allows linking of legacy applications with newly developed applications.	http://www.seer.com 8000 Regency Pkwy. Cary, NC 27511 (800) 499-SEER

MOM:

IBM		*MQ Series*	Message-oriented queue-based middleware. Supports distributed applications. Sends and receives data as messages. Available for over 25 different *OS* Platforms, including *Microsoft* Windows, *HP-UX*, *Sun Solaris,* using a single multi-platform API.	Most widely used messaging architecture in the enterprise environment. Plans to add Internet gateway. Load balancing is an advantage. (800) 426-3333.
DEC (Product line sold to *BEA* in 1997)		*MessageQ* (or *DECMQ*)	Supports real-time display via message queuing.	Also has *MessageQ*-to-*MQ Series* software bridge, for *VAX-MVS* interoperability (operates on dedicated *HP-UX or AIX* server) (800) 344-4825
BEA Systems, Inc.		*MessageQ*	*BEA's ObjectBroker 3.0* can utilize *MessageQ* for asynchronous or synchronous communication with other applications.	
Sybase, Inc.		*dbQ*	Expected to challenge *IBM's MQ Series.*	Queues Web-based applications.

6.1 *Available COTS Middleware, By Functional Category & Specific Service (Continued)*

Table 6-1. Middleware Vendors (Continued)

Vendor		Product	Description	Web page/Information/Comments
Category 1: Distributed Processing Middleware (Continued) *MOM (Continued):*				
Covia Technologies *HP* Business Partner:		*Communication Integrator (CI)*	A message-oriented software package for application communication by name, rather than address. For HP platforms	http://www.covia.com 9700 W. Higgins Rd. Rosemont, IL 60018 (800) 566-1969
NetWeave Corp.		*NetWeave Server*	Allows interconnection of open and legacy systems: Peer-to-peer cooperative application, application migration, and replicated data services.	http://www.netweave.com 2006 Chancellor St. Philadelphia, PA 19103 (215) 496-1540
New Era of Networks, Inc. (NEON)		*NEONet 2.2*	A messaging middleware cross-platform product for integrating C/S, Internet/intranet applications. Includes a customizable "rules engine," that has been integrated with *IBM's MQSeries*. Supported by *Sun Solaris*, *HP*-UX, *IBM's AIX* and *MVS*, and *Microsoft's Windows NT*.	http://www.neonsoft.com 7400 East Orchard Rd Suite 230 Englewood, CO 80111 (800) 815-NEON "Application integration software"
Suite Software		*SuiteValet*	CORBA-compliant object-oriented messaging middleware.	http://www.suite.com 801 E. Katella Ave. Suite 210 Anaheim, CA 92805 (714) 938-8850

6.1 *Available COTS Middleware, By Functional Category & Specific Service (Continued)*

Table 6-1. Middleware Vendors (Continued)

Vendor	ð	Product	Description	Web page/Information/Comments
Category 1: Distributed Processing Middleware (Continued) *MOM (Continued):*				
Tibco Software, Inc.		*Tibco Information Bus (TIB),* and *Enterprise Transaction Express (ETX)*	Merges workflow, using publish-and-subscribe, request/reply, and broadcast/reply across LANs and WANs. API support for *C++*, *Java*, and ActiveX. Open platform support for sharing data in real time.	http://www.tibco.com 3165 Porter Drive Palo Alto, CA 94304 (650) 846-5000

6.1 *Available COTS Middleware, By Functional Category & Specific Service (Continued)*

Table 6-1. Middleware Vendors (Continued)

Category 1: Distributed Processing Middleware (Continued)

Application Development Tools:

Vendor		Product	Description	Web page/Information/Comments
Peer Logic, Inc.		*PIPES Platform*	MOM distributed computing solution.	http://www.peerlogic.com 555 DeHaro Street San Francisco, CA 94107 (800) 733-7601
		Orbix + PIPES	Messaging + *ORB* concergence.	
Talarian Corp.		*SmartSockets*	Application development toolkit. Routes messages dynamically, eliminating the need to predefine routes between client and server. Provides publish-subscribe MOM services.	http://www talarian.com 333 Distel Circle Los Altos, CA 94022 (650) 965-8050
Momentum Software Corp.		*XIPC*	Application development toolkit. Provides network-transparent MOM services.	http://www.momsoft.com 777 Terrace Ave. Hasbrouck Heights New Jersey, NY 07604 (201) 871-0077
IBM		*Component Broker, CBToolkit*	Multi-platform application software object model development.	
Applix, Inc.		*ApplixWare* family	Tools for automating the decision-making process. Personal desktop application development. Requires a *Netscape* browser or other *Java*-enabled desktop, *Windows NT* or *UNIX OS*.	For personal desktop application development. 112 Turnpike Road Westboro, MA 01581 (508) 870-0300

6.1 Available COTS Middleware, By Functional Category & Specific Service (Continued)

Table 6-1. Middleware Vendors (Continued)

Vendor		Product	Description	Web page/Information/Comments

Category 1: Distributed Processing Middleware (Continued)
Application Development Tools (Continued):

Rogue Wave Software, Inc.		*DBTools.h++, C++/Java* Interoperability, Etc., *Suites*	*C++* access to databases. Different application development suites, including Math Suite.	http://www. roguewave.com 5500 Flatiron Pkwy. Boulder, CO 80301 (303) 473-9118
NobleNet, Inc.			Tools based on *RPC* services.	
IBM		*VisualAge for C++*	*SQL* data access class builder. Supports *ESQL* and *CLI* access to *DB2*, ODBC databases. Development platforms include *Microsoft's* Windows and OS/2.	
Sun Microsystems, Inc.		*Workshop NEO*	Tools for building CORBA-compliant enterprise applications.	
Bristol Technology, Inc.		*Wind/U 4.1*	Provides identical *Microsoft Windows* and *UNIXs Motif* GUI functionality. Supports *Win32* services, and *Microsoft's* component framework *ActiveX* and object model COM on *Sun's Solaris*, *HP-UX*, and *IBM's AIX OSs.*	http://www. Bristol.com 39 Old Ridgebury Rd. Danbury, CT 06810 (203) 798-1007

6.1 *Available COTS Middleware, By Functional Category & Specific Service (Continued)*

Table 6-1. Middleware Vendors (Continued)

Vendor		Product	Description	Web page/Information/Comments
Category 1: Distributed Processing Middleware (Continued)				
Java Development Tools:				
SuperCede, Inc.		*SuperCede*	Development environment for creating *Windows* applications with *Java*.	http://www.supercede.com 110 10th Ave., NE Bellevue, WA 98004 (800) 365-8553
Sun's JavaSoft		*Java Development Kit (JDK)*	Development environment for creating applications in *Java*. Includes JDBC API.	http://www.javasoft.com (888) 843-5282
Novera Software, Inc.		*EPIC Platform*	CORBA/IIOP *Java* component development and deployment.	http://www.novera.com 3 Burlington Woods Dr. Burlington, MA 01803 (888) NOVERA1
		EPIC Database	Uses JDBC to access data. Integrated with *Visigenic's VisiBroker for Java ORB*	
Prolifics/ JYACC, Inc.		*JAM*	Cross-platform tool for building C/S transactional applications. Works with *BEA's Tuxedo* TP Monitor environment.	http://www.prolifics.com 116 John Street New York, NY 10038 (212) 267-7722
Application Development Tools for the Web				
		JAM/Web	A visual development tool for building connections between Web servers and databases.	

6.1 Available COTS Middleware, By Functional Category & Specific Service (Continued)

Table 6-1. Middleware Vendors (Continued)

Vendor	🕯	Product	Description	Web page/Information/Comments

Category 1: Distributed Processing Middleware (Continued)

The leading CORBA-compliant ORB Vendors are *Iona Technologies, Inc., Visual Edge Software Ltd., Visigenic Software Inc./Borland International, Inc., BEA Systems, Inc., IBM, and Expersoft Corp.*

ORB

Vendor	🕯	Product	Description	Web page/Information/Comments
Iona Techno-logies, Inc. [A]	🕯🕯🕯	*Orbix*	CORBA 2.0-compliant *C++ ORB.* Runs on *UNIX, Windows* (OLE integrated), and *OS/2* platforms.	Leading CORBA technology Vendor [11]. Provides the tools for development of distributed, multi-threaded, scaleable OO applications. A *GUI Toolset* provides support for SNMP-based system management.
		OrbixWeb	Client-side *Java Orbix.* Supports *Sun's* JDK.	Allows creation of downloadable *applets* to access back-end services across the Internet.
			http://www.iona.com 201 Broadway Cambridge, MA	02139 (800) Orbix4U
Sun's JavaSoft		*Joe*	CORBA/IIOP *ORB.*	
Tibco Software, Inc.		*TIB/ObjectBus 2.0*	CORBA Publish/Subscribe *ORB*	

6.1 *Available COTS Middleware, By Functional Category & Specific Service (Continued)*

Table 6-1. Middleware Vendors (Continued)

Vendor		Product	Description	Web page/Information/Comments
Category 1: Distributed Processing Middleware (Continued)				
***ORB (Continued)*:**				
Visual Edge Software, Ltd.		*Object Bridge*	COM/CORBA interoperability solution. Supports IIOP for Communication across the Internet.	http://www. visualedge.com 3950 Cote Verde St. Laurent, Quebec Canada H4R 1V4 (408) 973-7823
			Generates a Just-In-Time (JIT) compiled proxy based a class description.	
Visigenic Software, Inc./ part of *Borland International. Inc./* becomes *Inprise, Corp.*		*VisiBroker for C++*	CORBA 2.0 *C++ ORB*	http://www. inprise.com 100 Enterprise Way Scotts Valley, CA 95066 (408) 431-1000
		VisiBroker for Java	CORBA 2.0 *Java ORB* and servers.	Included in *Netscape's browsers*
Visigenic/ Netscape		*Caffeine* [11]	*VisiBroker for Java* with RMI-like services added on top of CORBA/IIOP	
IBM		(See above)		
Digital		*ObjectBroker* (Transferred to BEA Systems, Inc.)		
HP		*ORB Plus 2.0*	CORBA 2.0/IIOP-compliant *C++* scaleable *ORB.* Runs on *Windows NT,* or *HP-UX* and *Solaris OS* platforms.	
BEA Systems, Inc.		*ObjectBroker*	CORBA-compliant	Integrated with *BEA MessageQ* (see previous *MOM* section).
		ObjectBroker Desktop Connection	Connects ActivX clients to *ObjectBroker*	

6.1 Available COTS Middleware, By Functional Category & Specific Service (Continued)

Table 6-1. Middleware Vendors (Continued)

Vendor		Product	Description	Web page/Information/Comments

Category 1: Distributed Processing Middleware (Continued)
***ORB (Continued)*:**

Expersoft Corp.		*CORBAplus*	CORBA-compliant *ORB* family of products for *C++*, *Java,* ActiveX. Has translation layer to communicate with OLE 2.0 objects. Integrates CORBA *ORB* with *MOM.*	Supports "distributed computing for the Enterprise." http://www.expersoft.com 5825 Oberlin Dr. San Diego, CA 92121 (800) 366-3054
Also: *TV/COM*		Example of *Expersoft's* *ORB* tool kit, used for developing a C/S digital TV system. [18]		http://www.tvcom.com

I-Kinetics, Inc. (See also [A].)		*DataBroker* "server" *OPENjdbc* "client"	Provides CORBA-based enterprise data access, based on *Iona's Orbix.* A *Java* JDBC "driver." Uses CORBA IIOP to talk to *DataBroker.*	CORBA component Vendor. Seventeen New England Executive Park Burlington, MA 01803 (800) I-KINETX http:/www.i-kinetics.com

The leading DCOM ORB Vendors are *Microsoft, Software AG of North America, Inc., Bristol Technology, Inc.,* and *Data Focus, Inc.* [17]*

Microsoft Corp.		*DCOM for Windows NT*	Shipped with *Windows NT 4.0*	http://www.microsoft.com One Microsoft Way Redmond, WA 98052 (800)426-9400

6.1 *Available COTS Middleware, By Functional Category & Specific Service (Continued)*

Table 6-1. Middleware Vendors (Continued)

Vendor	ô	Product	Description	Web page/Information/Comments

Category 1: Distributed Processing Middleware (Continued)
***ORB (Continued)*:**

Software AG of North America, Inc.		*DCOM for Solaris*	Partnered with *Microsoft* to provide *Microsoft's ActiveX component* software architecture for the Internet to *UNIX* platforms.	http://www.sagus.com 11190 Sunrise Valley Drive Reston, VA 22091 (800) 843-9534
Information Builders, Inc. (IBI)		*Enterprise Component Broker*	CORBA 2.0-compliant *Java* application server. Also supports IIOP and JDBC.	

* Some of the Vendors working on porting *DCOM* to non-*Windows* platforms are *Sun (Solaris), Digital (UNIX), IBM (AIX)* and *HP (HP-UX)*.

Category 2. Distributed Data Access Middleware

The leading relational DBMS Vendors are *IBM, Informix, Inc., Microsoft Corp., Oracle Corp.,* and *Sybase, Inc.* Complex data management capabilities (i.e. integrating object technology into a RDBMS) are being added by all the Vendors. These Vendors have also wrapped their DBMS servers with middleware that allows interaction with Web servers.

Remote Data Access:

IBM		*DB2 2.1*	Supports object extensions. Runs on *OS/2* and *Microsoft's NT,* and *AIX, HP-UX* and *Sun's Solaris OS* platforms.	

6.1 *Available COTS Middleware, By Functional Category & Specific Service (Continued)*

Table 6-1. Middleware Vendors (Continued)

Category 2. Distributed Data Access Middleware (Continued)
Remote Data Access (Continued):

Vendor		Product	Description	Web page/Information/Comments
Informix Software, Inc.		*Informix Dynamic Server*	Integrates database server, C/S networking, and Web/intranet services. With *Universal Data Option*, for new data types.	http://www.informix.com 4100 Bohannon Dr. Menlo Park, CA 94025 (650) 926-6300
Microsoft Corp.		ODBC	For accessing *SQL* databases. Based on *Microsoft's SQL Server*	
Visigenic Software		*OpenChannel* "server"* *OpenChannel* "client"* for *Java*	ODBC cross-platform data access. For *Windows NT* servers.	
		VisiChannel for JDBC	A JDBC-to-ODBC interface, using IIOP.	Enables JDBC Programs on client machines to access data in ODBC data server machines.
Intersolv		*Data Direct* *SequeLink 4.0* "server"* for ODBC and *Java*	Point-to-point (2-level) data connectivity middleware. Allows n-level deployment.	http://www.intersolv.com 9420 Key West Ave. Rockville, MD 20850 (301) 838-5000
Information Builders, Inc. (IBI)		*EDA 4 (Enterprise Data Access)*	Provides messaging and queuing service. For sending *EDA* messages,	http://www.ibi.com 1250 Broadway New York, NY 10001 (212)736-4433
		EDA/Message Hub	-across *Microsoft's Exchange Server,*	
		EDA/Message Switch	-across *IBM's MQseries,*	
		EDA/WebLink	-across the Web.	

* "Server" = server-based, "client" = client-based middleware.

6.1 Available COTS Middleware, By Functional Category & Specific Service (Continued)

Table 6-1. Middleware Vendors (Continued)

Category 2. Distributed Data Access Middleware (Continued)
Remote Data Access (Continued):

Vendor		Product	Description	Web page/Information/Comments
Oracle Corp.		*SQL*Connect*	*SQL* gateway software.	http://www.oracle.com
		Oracle8 Objects Option	CORBA *ORB* server. Supports objects or relational data.	
		WebServer	Uses a proprietary API, *Web Request Broker*, to link into *Oracle's* database services. Includes JDBC and *Java* VM. Services are structured as *Cartridges* (comparable to *Illustra's DataBlades).*	500 Oracle Pkwy. Redwood Shores, CA 94065 (415) 506-7000
		Mobile Agents	OFTP middleware.	
Oracle/ BEA Systems			Server certified to run with *BEA's Tuxedo* middleware *for Microsoft's Windows NT OS.*	Wide range of legacy and OO database access products.

Sybase, Inc.		*Object Connect -for C++ -for* OLE	Lets *C++* applications access relational data. Lets OLE components access relational data.	A mapping layer to its *SQL-server* database, to allow front-end objects to work with relational databases. http://www.sybase.com 6475 Christie Ave. Emeryville, CA 94608 (510) 922-3555
		Object Connect Server	Lets CORBA-compliant OLE objects access other objects stored on distributed servers.	
		jConnect	JDBC access to *Sybase's* databases.	
		OmniConnect	Vendor neutral data source connection.	

6.1 *Available COTS Middleware, By Functional Category & Specific Service (Continued)*

Table 6-1. Middleware Vendors (Continued)

Vendor		Product	Description	Web page/Information/Comments

Category 2. Distributed Data Access Middleware (Continued)
Remote Data Access (Continued):

Vendor		Product	Description	Web page/Information/Comments
Simba Technologies, Inc.		*Simba Express*	Data access middleware, ODBC desktop application to database connection. Thin client for any *Microsoft OS*, and a set of centrally managed server-based tools for *Microsoft's Windows NT OS*, and *HP-UX, Solaris* and *AIX, UNIX OSs.* Supports JDBC applications.	http://www.simbatech.com 885 Dunsmuir St. Vancouver, B.C. CA V6C 1N8 (604) 601-5300
Open Horizon, Inc.		Secure enterprise connectivity.	Places ODBC on top of DCE, to allow users to use a single sign-on (ID) to access all databases.	http://www.openhorizon.com

6.1 Available COTS Middleware, By Functional Category & Specific Service (Continued)

Table 6-1. Middleware Vendors (Continued)

Category 2. Distributed Data Access Middleware (Continued)
Remote Data Access (Continued):
TP Monitors:

Vendor		Product	Description	Web page/Information/Comments
BEA Systems, Inc./ Novell		*Tuxedo*	TP Monitor environment for enterprise applications. Most installations are *UNIX* based.	http://www.beasys.com 385 Moffett Park Dr. Suite 105 Sunnyvale, CA 94089 (800) 817-4BEA
		Tuxedo 6.3	Provides integration with *Microsoft's Windows NT OS*	
		Jolt	Extends *Tuxedo* enterprise OLTP applications to the WEB.	
IBM/ Transarc		*CICS, Encina 2.5*	*Encina* is a TP Monitor based on OSF's DCE. *CICS* is mainframe oriented.	http://www.transarc.com The Gulf Tower 707 Grant Street Pittsburgh, PA 15219 (412) 338-4400
			Encina 2.5 provides *C++* classes to build *Encina++*/CORBA or *Encina++*/DCE-based servers to support CORBA-based clients, on top of a CORBA-compliant *ORB.*	

6.1 *Available COTS Middleware, By Functional Category & Specific Service (Continued)*

Table 6-1. Middleware Vendors (Continued)

Vendor		Product	Description	Web page/Information/Comments
Category 2. Distributed Data Access Middleware (Continued)				
TP Monitors (Continued):				
IBM		*CICS*	Enterprise level TP Monitor, for the mainframe OLTP environment.	
Tivoli Systems, Inc. (Independent unit of IBM)		*TME 10 (Tivoli Management Environment)* Modules	For end-to-end management of distributed computing environments.	http://www.tivoli.com 9442 Capital of Texas Hwy. N. Suite 500 Austin, TX 78759 (800) 2-TIVOLI
HP		*Encina/9000 2.2*	For transaction processing.	
Remote Data Access via the Web:				
An application category called "application extension software" is being developed to allow applications to span both the enterprise and the Internet. RDBMS Vendors are using middleware to allow Web servers to interface with RDBMS servers. The database is then called a "Webified" DBMS. [22]				
Active Software, Inc.		*Active Web Integration System*	Web-enabled legacy system integration with *Java* front-end access to RDBMSs. The development tools are written in *Java*.	http://www.activesw.com 3255-1 Scott Blvd. Suite 201 Santa Clara, CA 95054 (408) 988-0414
Bluestone Software, Inc.		*Saphire/Web 2.1*	Automated generation of *C/C++* CGIs needed for dynamic production of the HTML required for a Web page request (where each page is an HTML file).	http://www.bluestone.com 1000 Briggs Road Mt. Laurel, NJ 08054 (609) 727-4600
		SaphireWeb 4.0	*Java* and Web database development and deployment, for bridging C/S and the Web.	The middleware is an integrated part of the development tools.

6.1 *Available COTS Middleware, By Functional Category & Specific Service (Continued)*

Table 6-1. Middleware Vendors (Continued)

Category 2. Distributed Data Access Middleware (Continued)
Remote Data Access via the Web (Continued) :

Vendor		Product	Description	Web page/Information/Comments
Wayfarer Communications		*QuickServer SDK*	Application programming extensions are provided to develop *ActiveX control* client components. The components run standalone in *Microsoft's Internet Explorer* or as plug-ins in *Netscape's* Web *browsers.* Agents are used to maintain a client's state on the server.	http://www.wayfarer.com 2041 Landings Drive Mountain View, CA 94043 (800) 300-8559
Apple Computer, Inc./ NeXT Software		*WebObjects 3.5*	Web application development software, extending C/S applications to the Web.	http://www.apple.com 1 Infinite Loop Cupertino, CA 95014 (408) 996-1010 The middleware is an integrated part of the development tools.
Oracle		*Web Server* with *WRB,* a Web Request Broker	Opens the database connection when the application/Web server is started.	Shipped with *Oracler Server 7.3*
Microsoft Corp.		*Web Assistant*	A tool for creating a HTML page that can be dynamically updated.	Bundled with *Microsoft's SQL Server 6.5* DBMS.
		Internet Information Server (IIS)	For access to ODBC-compliant *SQL* databases.	Bundled with *Windows NT Server 4.0 OS.*

6.1 *Available COTS Middleware, By Functional Category & Specific Service (Continued)*

Table 6-1. Middleware Vendors (Continued)

Vendor	0	Product	Description	Web page/Information/Comments
Category 2. Distributed Data Access Middleware (Continued)				
Remote Data Access via the Web (Continued) :				
Informix Software, Inc.		*Web DataBlade*	Web-based application development environment. Uses *Illustra Information Technologies'* object modules, called *DataBlades,* which encapsulate the data access functions that allow access to *Illustra's* ODBMS.	
Netscape Communications Corp.		*SuiteSpot 3.5*	A suite of 9 Internet software servers designed to scale from the local workgroup to the enterprise.	http://home.netscape.com 501 E. Middlefield Rd. Mountain View, CA 94043 (650) 937-2555
		Communicator	Adds e-mail and other Internet tools to *Netscape's Navigator browser.*	
		Informix's Online Workgroup Server	Uses *Netscape's LiveWire Pro* database access technology, which is part of *Netscape's SuiteSpot.*	With integrated directory and management services.
OneWave, Inc.		*Connector for Microsoft IIS (Microsoft Internet Information Server)*	Provides access to enterprise applications from *Microsoft's* Web server. Runtime: *Windows NT 4.0, IIS 3.0, Active Server Pages* CORBA extension, via: *Iona's Orbix.*	http://www.onewave.com One Arsenal Marketplace Watertown, MA 02172 (617)923-6500
NobleNet, Inc.		*NobleNet Web*	*Browser*-based C/S software deployment. Allows *Windows*-based C/S applications to be deployed across the Internet/intranet.	

6.1 *Available COTS Middleware, By Functional Category & Specific Service (Continued)*

Table 6-1. Middleware Vendors (Continued)

Vendor	ὃ	Product	Description	Web page/Information/Comments

Category 2. Distributed Data Access Middleware (Continued)
Remote Data Access via the Web (Continued) :

IBM		*WWW-Connection*	Provides Web connection for *DB2 V2.*	
		Net.Data V2	Enables Web access to relational data, for creating dynamic Web pages.	
Gradient Technologies, Inc.		*WebCrusader*	Provides enterprise security for Web-based applications.	http://www.gradient.com 2 Mount Royal Ave. Marlborough, MA 01752 (800) 525-4343
		PC-DCE for *Windows*	Provides for *DCE's* DFS secure file sharing, secure transaction processing for *Transarc's Encina,* and CORBA *Security Service (Level 1)* for *Iona's Orbix ORB.*	

6.1 Available COTS Middleware, By Functional Category & Specific Service (Continued)

Table 6-1. Middleware Vendors (Continued)

Vendor		Product	Description	Web page/Information/Comments
Category 3. Distributed Systems Management Middleware				
Automatic fail-over:				
Compaq Computer Corp. Acquired *Tandem Computers, Inc.*		*ServerNet*	Administrative software for automatic fail-over capability.	http://www.compaq.com 20555 SH 249 Houston, TX 77070 (281) 370-0670
Microsoft Corp.			Licensed by *Microsoft.*	
The leading object-oriented distributed systems management Vendors are *IBM, Sun's JavaSoft, Tivoli* and *CA.*				
IBM		*Component Broker Systems Management (CB/SM)*	GUI control point and agent object management.	
Peer Logic, Inc.		*PIPES View*	A GUI management console and management agents.	
BEA Systems, Inc.		*Management Console for Tuxedo*	A secure, *Java*-based console for administering *Tuxedo* over the Internet/intranet.	
Sun Microsystems, Inc.		*Solstice Enterprise Management*	Includes *Solstice Security Manager*	Provides for Single-Sign-On.

6.1 *Available COTS Middleware, By Functional Category & Specific Service (Continued)*

Table 6-1. Middleware Vendors (Continued)

Vendor		Product	Description	Web page/Information/Comments
Category 3. Distributed Systems Management Middleware (Continued)				
Boole & Babbage		*Command Post*	Enterprise level SNMP systems management. Has extension for *IBM's MQSeries*, to monitor *OS/2, AIX* and *MVS/ESA* platform statistics.	http://www.boole.com 2100 River Edge Pkwy. Suite 175 Atlanta, GA 30328 (800) 889-8933
		Command MQ	Gateway software for *SAP's R/3* C/S environment.	
Candle Corp.		*Command Center for MQSeries*	For *IBM's MQSeries* environment, distributed applications management.	http://www.candle.com 2425 Olympic Blvd. Santa Monica, CA 90404 (310) 829-5800
Acquired *Apertus Technologies'*		*MQView*	Centralized administration management.	
HP		*Distributed Enterprise (DE)/ Service Monitor*	*HP UX/OpenView* add-on that monitors the status of DCE services.	Enterprise view of services, rather than a view of nodes.
Computer Associates International, Inc. (CA)		*Unicenter TNG* ("The Next Generation") *2.1*	Integrated, open, end-to-end enterprise management. *Java*-enabled version that operates from a Web browser. (1997).	http://www.cai.com Acquired *Ingres.* One Computer Associates Plaza Islandia, NY 11788 (516) 342-5224

6.1 Available COTS Middleware, By Functional Category & Specific Service (Continued)

Table 6-1. Middleware Vendors (Continued)

Vendor	0	Product	Description	Web page/Information/Comments
Category 3. Distributed Systems Management Middleware (Continued)				
Talarian Corp .		*RTmonitor*	Distributed application monitoring, analysis and debugging.	
Managing Middleware				
BMC Software, Inc.		*PATROL*	Application management family of products. Via a library of *Knowledge Modules (KM)*, for example, for *IBM's* MOM *MQSeries*, *BEA's* TP Monitor *Tuxedo*, and *Oracle's* database. *OpenView*, and *Tivoli.*	http://www.bmc.com 2101 CityWest Blvd. Houston, TX 77042 (800) 841-3031 Integrated with *HP's*

6.1 *Available COTS Middleware, By Functional Category & Specific Service (Continued)*

Table 6-2. Object-Oriented Programming Language and Object-Oriented Application Development Tool Vendors

Vendor		Product	Description	Web page/Information/Comments
Microsoft		*OLE Software Development Kits*		
Taligent, Inc.		*CommonPoint Application System*	- A collection of objects and class libraries for developing workgroup applications, for text and graphics editing, database access and communications. - Company was formed by *IBM, Apple* and *H-P* in 1992. - Aimed at C++ developers. - Initially released for *IBM's* RS/6000 platform.	See http://www.software.ibm.com
ParcPlace Systems, Inc./ becomes *ObjectShare, Inc.*		*SmallTalk*	- OOP language. Runs on PCs, RS/6000s and SPARCstations.	http://www.objectshare.com 999 E. Arques Ave. Sunnyvale, CA 94086 (800) 759-7272
		VisualWorks	- OO application development environment, written in *SmallTalk*.	
Thompson Software Products, Inc.		*Nomad* *Teleuse/Win*	*- 4GL.* - OO *Motif* application development environment. - Runs on *IBM's* RS/6000 platform.	10251 Vista Sorrento Pkwy. Suite 300 San Diego, CA
Trinzic Corp.		*AionDS*, and *ObjectPro*	- OO application development environment, and front-end tools. for developing applications. - Runs on *IBM's* RS/6000 platform. (415) 591-8200	555 Twin Dolphin Dr. Redwood City, CA 94065

6.1 *Available COTS Middleware, By Functional Category & Specific Service (Continued)*

Table 6-2. Object-Oriented Programming Language and Object-Oriented Application Development Tool Vendors (Continued)

Vendor		Product	Description	Web page/Information/Comments
Versant Object		*Versant* *Argos*	- OO database management system. - OO application development environment. - Runs on *IBM's* RS/6000 platform.	1380 Willow Road, Ste. 201 Menlo Park, CA 94025
JYACC/ Prolifics		*JAM Transaction* Object Model	*Tuxedo*-based transaction integrity application (TP Monitor) development tool.	
Seer Technologies, Inc.		*HPS (High Performance System)*	A componentware development environment.	Includes *Seer's NetEssential* middleware communications services.
Informix Software, Inc.		*Web DataBlades*	Application development environment for object module encapsulation of data types.	
Bristol Technology, Inc.		*Wind/U*		
Data Focus, Inc.		*Nutcracker*		
Neuron Data, Inc.		*Elements Environment 2.0*	Rules-driven application development and integration environment for interoperable *C++*, Web, OLE, CORBA and *Java* objects.	http://www.neurondata.com 1310 Villa street Mountain View, CA 94041 (800) 876-4900

6.1 *Available COTS Middleware, By Functional Category & Specific Service (Continued)*

Table 6-2. Object-Oriented Programming Language and Object-Oriented Application Development Tool Vendors (Continued)

Vendor		Product	Description	Web page/Information/Comments
IBM		*VisualAge for C++*	OO visual application builder.	
Black & White Software		*Object/LM*	Management of deployment of distributed applications.	CORBA deployment and diagnostic tools. http://www.blackwhite.com 1901 S. Bascom Ave. Campbell, CA 95008 (408) 369-7400
		Object/Observer	Monitors distributed object communication.	
Visual Edge Software, Ltd.		*UIM/X family*	OO development tools.	

6.1 Available COTS Middleware, By Functional Category & Specific Service (Continued)

Table 6-3. Industry Consortiums and Standards Organizations

Vendor		Product	Description	Web page/Information/Comments
Object Management Group (OMG)		**CORBA** Vendors provide "CORBA-compliant ORBs"	- See Section 5.2.1.1.1. - International software industry consortium formed to promote open distributed processing using OO methodology. - Among more than 800 computer industry companies and end-users, Vendor members include *Apple, DEC, IBM, Novell, SunSoft,* etc. *(SunSoft), etc.*, and newer members *JavaSoft, Oracle, Netscape,* etc. The exception is *Microsoft*, which supports its own architecture.	492 Old Connecticut Path Framingham, MA 10701 (508) 820-4300 http://www. omg.org
Component Integration Laboratories, Inc. (CIL)		**OpenDoc**	- Consortium formed by *Apple, IBM, Borland, WordPerfect, Novell, Oracle, and Xerox* to counter the influence of *Microsoft's* OLE.	P.O. Box 61747 Sunnyvale, CA 94088
SQL Access Group (SAG)		**CLI**	- Standards group of Vendors and end-users formed for promoting SQL standardization.	
X/Open		**CLI**	- Standards consortium for *UNIX.* *(XPG).*	Publishes the *"X/Open. Portability Guide"*
Message Oriented Middleware Association (MOMA)		N/A	- International consortium (Vendor-centric forum) dedicated to enhancing the interoperability of distributed and C/S computing via message-oriented middleware (MOM).	http://www. moma-inc.org (415) 378-6699

6.1 Available COTS Middleware, By Functional Category & Specific Service (Continued)

Table 6-3. Industry Consortiums and Standards Organizations (Continued)

Vendor	Ô	Product	Description	Web page/Information/Comments
Internet Engineering Task Force (IETF)		N/A	- Internet protocol engineering and development organization, consisting of working groups organized by topic (e.g. routing, security, etc.). of the Internet.	http://www.ietf.org Funded by the NSF (National Science Foundation) to facilitate the growth Work is done via correspondance.
Open Software Foundation (OSF)		***DCE/RPC*** *Motif* GUI for *UNIX*	- Not-for-profit R&D organization. Provides software solutions for open systems (Vendor consortium.)	http://www.opengroup.org Cambridge, MA (800) 767-2336
Object Database Management Group (ODMG)		ODMG-93 specification	A consortium of object-oriented database management system (ODBMS) Vendors and interested parties promoting standards for object storage. ODMG-93 guarantees portability between different ODBMSs.	http://www.odmg.org 14041 Burnhaven Dr. Suite 105 Burnsville, MN 55337 (612) 953-7250
Desktop Management Task Force (DMTF)		**DMI**	Industry consortium for defining Desktop standards.	http://www.dmtf.org c/o *MacKenzie Kesselring, Inc.* 1230 SW First Ave. Suite 220 Portland, OR 97204 (503) 294-0739

6.2 List of COTS Midleware Vendors: See Appendix B

(Alphabetical)

7. MIDDLEWARE FOR NETWORKED SYSTEMS MANAGEMENT

7.1 Systems Management

Systems management functions consist of network management and systems management:

(a) **Network Management** entails:

- Message routing.
- Bandwidth maintenance/performance management.
- Fault management (e.g. automatic problem detection, testing and results reporting, via, for example, *alerts*).

(b) **Systems Management** covers administrative, operations management functions, such as:

- Inventory/asset management (e.g. discovery of what hardware and software assets are running on the network).
- Configuration management.
- Change management (e.g. removing unwanted applications).
- Software distribution (e.g. adding automatic updates from the Web).
- Licensing management (e.g. license metering).
- Security management (e.g. anti-viral software).

Total Cost of Ownership (**TCO**) considerations for a C/S - distributed computing - environment demand the use of systems management tools that allow administrators to perform management from a central point. Thus, the management features that are considered standard are GUI-based administration and performance monitoring, and a single workstation to manage distributed servers. This has been implemented by most Vendors.

Systems management products can also be grouped in terms of the systems management market. In this case there are several categories, as shown by Figure 7-1.

- **Data Management.**
- **Storage Management.**
- **Desktop Management.**
- **Distributed Systems Management.**
- **Object Management.**
- **Service-Level Management**, which is typically used to describe how well specific applications (for example, *SQL* databases) deliver services.

and

- **Applications Management**, which is used to describe the management of middleware itself!

Figure 7-1. Available COTS Systems Management Products, By Market Category

7.2 Network Management

SNMP-based network management tools, such as *HP's OpenView*, or *IBM's NetView*, are geared toward managing network devices, such as hubs. However, most network management Vendors today (1997) have extended their products, with the addition of 3rd party tools, to provide systems management suites. Vendors are also moving from providing mainframe system management to providing C/S system management, for example, *IBM*, with *SystemView*.

7.2.1 Explicit vs. Implicit Traffic Management

Most network traffic management today (1997) relies on "implicit" control. That is, traffic flow is controlled by the protocols, such as TCP/IP. If a scaling of the GS's assets to match the processing requirements is desired, "explicit" control must be considered. Explicit traffic control uses feedback to determine the available resources.

> There does not seem to be a need for network traffic management in the real-time telemetry analysis/satellite control environment.

7.2.2 Distributed Systems Management (DSM) Platforms

> In general, today's (1997) COTS available distributed systems management products have been developed for managing large, complex, multivendor networks. Thus there is little applicability, and a minimum of distributed systems management middleware available for use in a GS environment.

The major competitors in the SNMP overall enterprise management platforms arena are *HP's OpenView*, *IBM's* subsidiary *Tivoli Systems Inc's Tivoli*, and *Computer Associate International's (CA) Unicenter TNG.*

A convergence in the basic management platforms to a few Vendors is expected. [11] The trend today (1997) is to extend one of the open system management platforms, such as *HP's OpenView*, and *IBM's NetView*, into distributed systems management by the addition of 3rd party management applications to the base platform. A lag in 3rd party product release, after a platform version update, may be expected, if there is late release of the development code to 3rd parties.

7.3 Distributed Systems Management

Most systems management is based on the use of a *manager/agent* protocol. An integrated set of management applications resides on a central station, and "agents" reside on the managed systems, and provide management information to the central station. **Middleware** provides the agent-to-managing station communication.

The two main mechanisms for retrieving data from the managed systems are *polling*, for data reporting, and *alerts* (or *traps*) for exception reporting.

Support of multiple *UNIX* platforms, as well as *Windows*, is essential in a distributed computing environment. Agents that can do both SNMP and DMI are required.

7.3.1 DSM Frameworks

Comprehensive information must be gathered in real time when running a networked application to track down problems. Debugging and monitoring capabilities required include (1) monitoring information in real time as it changes, without making changes in the application, and (2) monitoring applications in production (instead of creating special debug versions).

7.3.2 DSM Mechanisms

Real-time monitoring tools include:

- A graphical tree, with the root being the application and the branches are the server processes that make up the application.
- A multi-window application development interface that provides real-time views of inter-process communication activity.

DSM mechanisms:

- Are portable across platforms.
- Provide tracking of processes: When they start, when they join an application, and when they fail (or terminate).
- Do message logging (to a file) at the client process (sent and received) interface.
- Do error call-backs when an exception occurs.

DSM mechanisms provide a list of:

- Server nodes, and server application processes.
- Connected clients, and client processes attached to the server processes, with subject (message content) of the client processes.

7.4 Middleware for Distributed Systems Management (See 3.1.2.3, 3.1.3.3, and 3.4.2.3)

An evaluation of middleware management tools must consider:

(a) The availability of **hooks** to other Vendor's tools.

(b) The integration level with other Vendor's solutions through **standards**, such as the Simple Network Management Protocol (SNMP), and the Desktop Management Interface (DMI).

For example, *Tivoli Systems' TME for Windows NT* and CA's *Unicenter for NT* integrate with *Microsoft's* Systems Management Server's (**SMS**) desktop management functions, which provides information to DMI agents.

7.5 Illumination of Single Points of Failure and "Bottle-Necks"?

Multiple paths must be used to eliminate single point failures:

7.5.1 Fail-Safe (24x7 Up-Time)

Mission-critical applications, that can't stand down-time, are run on fail-over clustering platforms. **Clustering** is a group of systems that work together as one. When one system fails, the work is re-routed to the functioning system.

> For the GS environment, *Microsoft's NT* initial offering of 2-node fail-over, for example, is considered adequate for a telemetry data archive server cluster.

The administrative software for automatic fail-over capability is provided by *Microsoft (Tandem's ServerNet)*, or its clustering partners: *Compaq, DEC, H-P, NCR (LifeKeeper for Windows NT)*, or *Tandem* (acquired by *Compaq*, June1997) (originally licensed by *Compaq* - and ported from *UNIX* to *NT - ServerNet)*. [7].

7.5.2 "Bottle-Necks"?

A disadvantage of the C/S model is that a server may become a "bottle-neck," i.e. the servers limited resources may not be able to serve an increasing number of clients.

Another "bottle-neck" may be presented if there is a LAN-to-Internet connection. The bandwidth available to applications is limited by the Internet Service Provider (**ISP**) network connection. The connection is usually T1, at 1.544 Mbps, or less. Individual users may be connected at only 14.4 Kbps for dial-up. This is compared to a typical C/S Ethernet LAN speed of 10Mbps.

The solution in both cases is, of course, to add more servers.

7.6 Middleware Management

Middleware management/application management tools at the present time (1997) are immature. However, a "standard" is being set by *IBM's* complete middleware solution, as provided by it's *Component Broker (CB)* framework. Other Vendors of products for managing distributed objects must match this solution.

For example, *BMC Software, Inc.* has a family of tools for managing applications. The tools display real-time status of performance indicators, for example, for *IBM's MQSeries.*

7.7 Web-Based Systems Management

Systems management via the Web, which provides management information AND allows the execution of management commands over the Web, is being developed using *browser*-based interfaces, based on *JavaSoft's Java.* While not yet available (1997), products are expected from *IBM/Tivoli*, based on the TME10 management platform, and from *Computer Associates*, based on it's *Unicenter TNG* product. Instead of using a central console, systems could then be managed from laptops.

8. CONCLUSIONS, O&M, and MIDDLEWARE PRODUCT RECOMMENDATIONS

8.1 CONCLUSIONS

The blueprint for the next generation, common satellite ground station will show a 3-level C/S architecture. The architecture can standardize its desktops and servers, as well as databases and applications. Networking software follows the general OSI protocol stack. However, from the network up, the competing middleware products do not fit a consistent framework. Competing middleware products may be missing a capability. Also, no Vendor provides COTS development-free middleware products (1997).

Therefore, the characteristics that should guide middleware selection are:

- A single source of middleware services: To increase the chance of all components working together. There are too many Vendors offering middleware services. The key is not making a "patch quilt" of the middleware solution.
- An open solution: Multi-platform and multi-OS capability.
- Multi-level support: Availability of APIs, and also debug and management facilities.

8.1.1 Anticipated Future Technology Changes

Due to the rapid idea-to-market cycles, the focus should be on "what works," not on the latest technology changes. A technology choice can thus never be made! Instead we must decide on the computing framework that allows us to evolve, while maintaining our investment and computational power integrity.

Middleware, the software that is needed to tie together distributed systems, is here to stay. The driving force is economic:

- Computer networks, with PCs and workstations, provide a better price/performance ratio than mainframes.
- Performance can easily be scaled by the use of several network nodes.
- Heterogeneous applications and databases can be used in the C/S model.
- Reliability is increased by the availability of backup nodes on the network.

8.1.1.1 Current State of the Practice

With Internet-driven new technology terms, such as "*applets,*" "*Beans,*" "*cookies,*" etc., coming into use, a complete paradigm shift to OOT is required for effective utilization of middleware. This shift is in process.

8.1.1.1 Current State of the Practice (Continued)

CORBA is well-defined, but lacks in implementation. "Shrink-wrapped" *ORB*-based middleware may not be expected until 2001 [17]. The full potential of object technology for rapid software development, re-use, and reduced cost in deployment, has not yet been demonstrated.

Computing models have evolved, from mainframes-to-minicomputers-to-PCs-to-C/S, and now to Web/intranet.

C/S applications range from small workgroup or departmental systems (the GS environment), characterized by a limited number of users and low transactional volumes, to very large on-line transaction-processing (enterprise environment) systems. Most of the available COTS middleware has been developed for use in the (large intranet) enterprise arena. Middleware for improving GS performance, and decreasing the O&M cost, is thus not mass market. Due to the variety and disparity of the products that claim to be middleware, selection of the right "package" (integrated set of middleware products) is key to getting the best performance out of the middleware.

8.1.1.2 Future Trends

- A gradual merging of the competing middleware standards is expected.
- The Internet can no longer be treated separately. It should be part of all applications.
- Will the back office applications (the database servers) be "Webified"?
- The *UNIX* installed base will gravitate to the market leaders: *Sun, IBM. HP.*
- *UNIX's* role will be reduced to the niche market of (a) as an engineering desktop, and as a back-end **application server**. *Windows'* role will be (a) as the **common desktop**, and (b) as a contender for the server market.
- RDBMs will be extended to handle objects.
- CORBA compliance will be essential.
- Legacy environments will continue to exist. There will never be the available resources to replace or rewrite all of the existing code bases.
- Complex data types, and object databases, for audio and video, are still in the future (2000).

8.1.2 Risk of the Various Middleware Implementations
CORBA-compliance, due to the support provided by the hundreds of coalition member companies, is expected to dominate. The closest competitor is *Microsoft's* DCOM, and *Microsoft* has promised inter-operability with CORBA (1997).

To avoid lock-in to a Vendor (e.g. *Microsoft*), a Vendor's product trend toward open systems and standards must dictate the middleware choice.

8.1.2.1 Alternatives
Middleware for OO application development can be either independent (i.e. works with any) (e.g. *Suite Software's SuiteDOME)* or dependent on a particular set of development tools (i.e. is a built-in service that must be used when developing the application) (e.g. *Seer Technologies' High Performance System*).

> The ideal COTS solution for the GS environment would be the availability of a "complete solution," which provides both, the (telemetry) processing platforms, and also the C/S OOT middleware.

8.1.2.2 Scope of Custom Software Development Required
COTS middleware eliminates the need for network programming skills in application development. However, a learning curve of 4 - 6 months is expected to be required to understand the middleware architecture, and for testing and integration of the middleware product(s) into any GS environment.

Experience ramp-up time is thus an additional consideration to the purchase price.

8.1.2.3 Middleware Support Requirements
The software budget must include Vendor support charges. Creating seamless client to back-end server interfaces today (1997) requires many hours of support by the middleware vendor.

The software budget must also include factors such as an evaluation of the software licensing terms. In addition to the server software license, is the client ("per seat") license an "unlimited use" license, i.e. a "site license"? Or, is the license "per client," or "per Web *browser*," and is it either a "concurrent license," which counts only the users actually logged on to a server, or does it count "every user that has the potential to log on"? For example, *Microsoft* has three licenses: An NT Server license, an NT Workstation license, and a Client Access License, for every terminal that can access the server.

8.2 Effect of Middleware on OPERATION & MAINTENANCE (O&M)
Operation & Maintenance of heterogeneous computing systems (or satellite systems), due to the disparate requirements levied by each of the different systems, costs too much. If we can abstract the functionality from the implementation, we can reduce the O&M cost.

MAGIC proposes to reduce O&M costs by:

Operation:

- Use of a consistent and graphical user interface, to reduce operator training.
- Operator knowledge-base training, for a realistic training environment.
- Operator Expert System-based decision-making assistance.
- Use of COTS software components to obtain a quick and not very steep learning curve.

Maintenance:

- Non-integrated software components, making it easier to replace or upgrade a function without affecting other components.
- All software components would communicate using a standards-based message passing protocol, so that they can be replaced with any component that supports that protocol.
- Use standard PCs and hardware components.
- Use COTS software components, to minimize the need for unique code.

A major decrease in O&M costs can also be achieved by providing for autonomous operation. Middleware can enable the achievement of this goal. It can provide the operator with a consistent, manageable, extensible, single system image of the distributed computing environment presented by the next generation, common satellite ground station.

Integrated end-to-end heterogeneous resource management is required to minimize O&M costs. Resources that can, and must, be managed include networks, systems, applications, databases, and non-IT devices. Application management includes automatic monitoring and management of applications, AND the servers, middleware, and underlying resources that affect application performance.

Middleware allows an application to be logically separated from the implementation of new technology, network expansion, etc. For example, using middleware services allows an application to run on different networked systems with transparent access to the underlying network protocols. For different satellite systems this is accomplished by the use of object oriented programming (OOP) in developing the controller and communications software. Satellite families, and functions such as orbital maintenance, subsystem monitoring trending, and ground/spacecraft communication, can be grouped in classes. Classification abstracts the satellite attributes and operation from the implementation. Reference the Aerojet study [B].

8.3 *MIDDLEWARE PRODUCT RECOMMENDATIONS*

The next-generation, common satellite ground station architecture is expected to be C/S OOT, performing distributed event-driven computing. This design requires middleware. Further, the middleware must be object-oriented, scaleable, extensible, portable, open standards compliant, and providing performance-tuned relational database access and legacy system integration.

The recommended approach to selecting middleware products is:

A. Establish what your needs are.

B. Establish an architecture/top level design.

C. Choose the data management system first. Today's products tend to be Web- and RDBMS-centric. Focus on a single-Vendor *SQL* data access solution.

D. Match everything else to (C). Create a matrix of which required COTS products are compatible with each other, and which are potentially compatible.

E. Develop your applications in the middleware layer, rather than going to a fat client or with stored database procedures.

F. Endeavor to get the database middleware, the *ORBs*, the MOMs, etc. from the same Vendor.

Recommended middleware products, or equivalents, for the next-generation, common satellite ground station are:

Vendor/Product:	Reason for Recommendation:
1. **Distributed processing middleware:**	
• *IBM's CB*	A "complete," cross-platform, CORBA solution. Internet capable. Allows encapsulation of legacy applications.
2. **Distributed data access middleware:**	
• *Oracle Server 8*	Supports the 3-level C/S architecture, and CORBA, OLE, *SQL3* and *Java* objects.
3. **Distributed systems management (DSM) middleware:**	
• *HP's OpenView*	Open systems management framework, with many 3rd party management applications.

APPENDIX A*

NOTES

SunSoft products. *Microsoft* products.

A.1 Middleware Glossary

ActiveX	A class of OO technologies, based on *Microsoft's* COM object model/architecture.
ActiveX Controls	Running on *Microsoft's Windows platforms* components, or component-like programs, written to do specific tasks in *Microsoft's Visual C++ language.* Similar to *applets*, except that the components are compiled with the application. Used to create Web pages with active content.
agents	Pieces of procedural code that are added to produce a particular function on another computer. For example, an *agent* may collect management information. If written in an OOP language, *agents* become "objects."
aglets	*Applets* with *agent* technology.
API	A published list of functions that a programmer can use to perform tasks, or to invoke services.
applets	Portable (platform-independent) components, or component-like programs, written to do specific tasks in *Sun*'s *Java* language. Designed to be distributed on the Web, and downloaded into a *Java*-compatible *browser* each time they are run. Allow the distribution of executable content across the Web along with the data. Used to create Web pages with active content. Can be created with tools like *Sun's Java Developers Kit (JDK)*, *Symantec's Visual Café*, or *Microsoft's Visual J++*.
application	Comprehensive class libraries, containing reusable object frame works components.
application servers	DBMS, TP Monitor, groupware, object, or Web servers.

*** Glossary names and acronyms may be trade marked or service marked.**

APPENDIX A*

NOTES

A.1 Middleware Glossary (Continued)

architecture	A company's IT architecture is a written set of guidelines for a desired future for IT. In the C/S context, the concepts and structural elements that are used as building blocks in the design of distributed computing systems. For software, the organizational structure identifying the components, their interfaces, and a concept of execution among them.
Big COTS	COTS packages that provide many functions that are integrated.
binding	Refers to a client contacting a remote system to have a remote procedure executed. *Early binding:* The client defines the parameters and executes an RPC, via native DBMS's client libraries. (An RPC is generally faster than a *SQL* call.) *Late binding:* The parameters are defined by the server from an *SQL* query (i.e. the *SQL* parsing is performed by the server).
browser	Client software that "speaks HTML." It, primarily, interprets HTML commands in information it receives from a HTML (Web) document server, and displays the text and images using the client platform GUI. It also sends (Web) forms, i.e. service requests, to a HTML (Web) server, using the HTTP protocol. A "*Java*-compatible *browser*," contains a *Java* Interpreter.
business objects	A CORBA description for application-independent concepts that represent end-user "recognizable" entities. These objects have well-defined interfaces (via the IDL language), and can interact (communicate) with other objects (using the CORBA *ORB*). Distributed components.
C2	U.S. government security standard for an OS, requiring user and application authentication before gaining access to any OS resource.
Caffeine [11]	*Visigenic/Netscape's* CORBA-compliant *Java ORB.*

APPENDIX A*

NOTES

A.1 Middleware Glossary (Continued)

cartridges	*Oracle's* manageable objects, with CORBA's IDL-defined interface.
CGI	Provides the HTTP (Web) server interface for a *browser*. The CGI protocol is used to translate service requests and send them to a server application or back-end, such as a DBMS. Results are returned to the HTTP (Web) server in HTML format. HTTP/CGI is the predominant 3-tier C/S model for the Internet today (1997). As CGI server applications are accessed using proprietary APIs, such as *Microsoft's Internet Server API (ISAPI)* and *Netscape's Netscape Service API (NSAPI)*, there is a trend toward open Web server application and back-end service access based on the CORBA specifications.
class	Description for a group of objects that have similar characteristics, i.e. the objects are similar to one another in attributes and behavior.
CLI	Callable *SQL* API for relational database access.
components	CORBA distributed objects. For example, can be designated by a visual object (typically an icon) on a screen. Also, an independent piece of COTS software. As defined by *IBM:* CORBA objects, implemented in either *Java* or *C++*. Originally, called "controls," and having settable properties and methods that can be called (by an application program). Examples are: *Microsoft's* 16-bit *Windows DLL - Visual Basic eXtensions (VBXs)*, and the 32-bit versions - *ActiveX Controls* low-level (developer) components, as well as platform-neutral *Java applets* higher-level (may be used directly by end-users, by downloading via Web *browsers*) components, and *Java Beans* low-level components.
componentware	Small, well-defined application objects that work together to form a broader solution.
compound document	An electronic document which can carry data, images and video. A visual container of components.

APPENDIX A*

NOTES

A.1 *Middleware Glossary (Continued)*

	The framework for deploying components on the desktop, and via the Internet/intranet.
connection	A communication link, between two processes.
container	A component that can embed another component. For example, a compound document, and the *Windows* Desktop.
conversation	A connection between two user procedures.
cookies	*Netscape's* data elements (a few bytes of information) downloaded from a Web site into a *browser*, and generally used by the site to track a *browser's* activity. For example, to maintain session information for a persistent session.
CORBA/*Java ORB*	**CORBA/IIOP** *ORB* written in *Java. Sun's Joe.*
COTS	An item produced and placed in stock by a commercial distributor (Vendor) that is used without modification.
database	A collection of related data stored in one or more computerized files in a manner that can be accessed by users or computer programs via a database management system (DBMS).
data mining	The process of discovering patterns in data based on associations, clustering, occurrences, etc.
data warehouse	An information systems architectural construct consisting of an intermediate server that pulls data from a number of multiple sources (servers). Client systems interact with the intermediate server, which provides a consistent view of the enterprise. The data presented to the user is usually in a consolidated or summarized form. For example, the presented data may consist of copies or subsets (rearranged for fast access) of database data that are periodically replicated or updated under IS control. Users do not query the database directly, thus safeguarding the data. Database services, e.g. for data replication, are considered middleware services.

APPENDIX A*

NOTES

A.1 Middleware Glossary (Continued)

DCOM	Network OLE *ORB* from *Microsoft*. A major competitor to CORBA.
DCOM for *Java*	*Microsoft's Visual J++ ORB* that allows a *Java* object to invoke a remote *Java ORB*, using the DCOM *ORB*. A DCE/RPC-based protocol.
DDE	*Microsoft Window's* shared-memory message-passing facility.
distributed	Implies heterogeneity.
distributed application	Applications are partitioned, with portions running on clients and servers, by defining the C/S processing interface. If the process involves distributed objects, the application is partitioned into user, object, and method features, which eliminates the need to define the C/S interface.
distributed computing	May designate multiple-server C/S, with distributed data sources linked by a networked computing environment.
distributed object	An object which may be hosted on more than one platform. An independent software component, which can be accessed by users across a network. A CORBA distributed object is accessed by a remote client by method invocation.
DMS	An integrated set of computer programs that provide the capabilities needed to establish, modify, make available, and maintain the integrity of a database.
domain	Range of legal and logical values, for a field.
dynamic extensibility	Allows automatic plug-in of software modules (in a *browser*) when needed.
encapsulation	Combines code (also called procedures or methods) and data (also designated as data structures) into a single entity known as an "object" that a programmer can manipulate without knowing the

APPENDIX A*

NOTES

A.1 Middleware Glossary (Continued)

	details of its implementation. The details are "hidden" from the programmer, by restricting access to class member functions.
	A bundling of data and methods.
environment	For example, the framework for developing distributed applications.
extensibility	Software is extensible if functionality can be added with a minimum of code modification. In OOP, extensibility is enhanced by distinguishing public (published) and private (internal to a class) operations.
Extranet	A selective extension of the intranet, with selective access provided to business partners and customers.
fat client	A PC which contains more computing power than most users ever need.
	The majority of an application's processing is accomplished by the client. Used where application response time requires a lot of local hard drive space.
fat PC	A PC that can act as both client and server.
fat server	Contains stored procedures to help in accessing data.
firewall	A computer that sits between the Internet and a (company-internal) protected network. It filters traffic to and from the Internet using security software.
form	A client HTML page with one or more data entry fields, with a service request "submit" button. A form's inputs are collected by a *browser* and sent to a HTTP (Web) server.
framework	(A software environment consisting of) a collection of products and their common protocols.
gateway	For databases, passes a *SQL* statement to a remote database system (typically another Vendor's).

APPENDIX A*

NOTES

A.1 Middleware Glossary (Continued)

A device or PC that connects two dissimilar networks, translating between protocols.

groupware — Collaborative software. The prime example is e-mail.

In the C/S environment, considered as a form of middleware.

HotJava — *Sun's* Web *browser*. Can interpret *Java*-generated code.

HTML — The document formatting language used to build Web pages.

HTTP — The language of Web servers. Provides RPC-like semantics on top of *sockets.*

The communications protocol used on the Web to transmit HTML-encoded pages. The protocol used by *browsers* to download Web pages, *applets*, and images.

HTTP/CGI — 3-level Internet C/S applications model. (See also HTTP and CGI.) Replaced by CORBA.

hyperlinks — HTML commands (tags) that transparently allow a jump to a linked page (or point within a page) by a click of the mouse. Text in a document that is used as a hyperlink is usually highlighted and underlined by a *browser*. Can also be combined with graphics (e.g. *Windows* "buttons").

The link may be to another spot within the same document or same server, or to another server or Web site.

HyperText — Software mechanism that links documents to other related documents, or to other resources such as image files. Text that is "marked up" by structure-describing "tags" and hyperlinks.

Implemented by HTML.

IIOP — The open Internet protocol for communication between networked applications and objects. TCP/IP with CORBA-defined message exchange added.

APPENDIX A*

NOTES

A.1 Middleware Glossary (Continued)

IIS	*Microsoft's* Internet Information Server
infrastructure	The design of the network and the attributes of the physical nodes (the hardware) that exist on the network. Here, defined as the infrastructure software, or "middleware."
intranet	A private (company internal) IP network, usually modeled on and using Internet and Web technology. A private Web, operating behind secure Internet firewalls, and characterized by compliance with Internet protocols and standards.
Internet	A group of (company external) networks that communicate via TCP/IP over telecommunication channels. A Global network. Complies with the *Internet Engineering Task Force (IETF)* standards.
IP	The standard inter-network routing protocol in the TCP/IP protocol stack.
IPC	A mechanism for independent processes to exchange and share data. In an OO environment, communication is based on object types and properties, rather than on static addresses.
Java	An OO programming language from *Sun*, based on C++.
Java Beans	Native component model for *Java*, like *ActiveX Controls*, but platform neutral. Supported by *IBM/Lotus.* A framework similar to OLE ActiveX and OpenDoc parts.
Java PC	A Network Computer (NC) that natively runs a *Java OS*. It downloads *applets* to perform a task.
Java Sockets	Substrate technology for writing a C/S application in *Java.*
Java Studio	*SunSoft's* tool for using *Java Beans* in applications.

APPENDIX A*

NOTES

A.1 Middleware Glossary (Continued)

Java Virtual Machine (VM)	Runtime. Runs *Java* compiled bytecode as if it was machine language.
Java Workshop	*SunSoft's* development environment (toolset) for creating Internet applications and Web pages using *Java Beans.*
JDBC	Provides an object interface to *SQL* databases
Joe	A portable CORBA-compliant *Java* ORB (CORBA/IIOP ORB) written in *Java*, from *Sun/JavaSoft.*
JVM	*Java* Interpreter. May also contain a JIT Compiler.
Kerberos	A trusted 3rd party authentication service based on 4.3 BSD. [23]
Little COTS	COTS packages that provide pieces of functionality, such as a GUI, Expert System, database, etc.
member functions	Implement the different operations on an object that are defined by a class specification.
messaging	A communication system in which a message, consisting of a block of data, is delivered from one communicating entity to another.
message-passing	The means of communication between objects. A "message" serves to initiate processing and request information. It indicates which method to invoke, and passes the arguments for that method to the object (or specifies how the arguments are to be passed).
method	A function or procedure defined for a class of objects. The code element of an object.
middleware	Refers to the various software services for establishing communication between a client and a server. The "idea" of middleware is to make system architectures and the underlying protocols transparent to the application procedures.
model	A physical, mathematical, or otherwise logical representation of a

APPENDIX A*

NOTES

A.1 Middleware Glossary (Continued)

system, entity, phenomenon, or process. An organized presentation of concepts and terminology. A conceptual framework for proposed technologies. A pattern, or standard.

MOM — Provides asynchronous message queues on both the client and server sides. For mobile users.

multitasking — Running multiple programs concurrently. The server services multiple clients concurrently.

multithreading — A "thread" can be used to represent each active object instance. Multiple object are then run concurrently within the same process.

NEO — *Sun's* CORBA-compliant *ORB*.

Network Computer (NC) — Uses a network connection - not a local hard disk - to run server-based applications.

Referred to as a "thin client." Also refereed to as "no desktop," since it is usually diskless.

A user's programs and data are kept at a central location and downloaded across a network to a terminal as needed. The applications may run on the server, or on the desktop, depending on the NC implementation.

(The "terminal" is differentiated from a mainframe terminal by the download process.)

object — An abstraction which combines both the data structure and the procedures that are implemented on the data in a single entity. Objects are reusable and extensible, and encapsulate data and the procedures that can be used to manipulate the data. Objects have clients and provide services to clients.

Data encapsulated by methods.

Operations valid for the object are stored together with the object as its "methods."

APPENDIX A*

NOTES

A.1 Middleware Glossary (Continued)

object databases	Facility for storing complex data, such as BLOBs, sound and video clips, fingerprints, etc.
object model	In the case of distributed computing, a client issues a service request which identifies a service, or operation, to be performed. Selection of the "method" to perform the service is based on either objects identified in the request, or on the requested operation.
OO	A software development method that organizes software as a collection of objects. This approach to software development has required a "paradigm shift" from structured programming.
OOP	Modular code which allows joining of components.
open architecture	Solutions are non-proprietary, i.e. the technology is not controlled by a single Vendor. Examples are: CORBA, OpenDoc and the Web, which are controlled by Vendor consortia.
ORB	Middleware which establishes the C/S relationship between objects. It is software that automatically links the distributed objects that make up an application. It allows objects to dynamically discover each other by exchanging metadata and to interoperate. The *ORB* provides cross-process and cross-platform access to objects. It is the "object bus." Clients have proxies to objects on a server. A client object can invoke, transparently to location, a method on the server object using an *ORB*.
ORBlet	An *ORB* that can be downloaded on-demand, like a *Java applet.* Written in *Java* bytecode.
page	An ASCII text file with embedded HTML commands.

APPENDIX A*

NOTES

A.1 Middleware Glossary (Continued)

paradigm	A model: In an OO paradigm, data are considered primary, and procedures are secondary. As compared to the "functional" paradigm, where functions and procedures are primary, and data are secondary.
plug-ins	*Netscape's browser* extensions, similar functionally to *Java applets.*
process	A program (e.g. an application) being executed by a computer's OS. In C/S terms, a server process is controlled by a client process. Threads run within processes, which in turn run within sessions.
program	An executable file, usually created by a link editor, and residing on a disk.
protocol stack	The communication protocol layers through which network traffic moves.
proxy	In a CORBA client process, the IDL stub that provides an interface to object services. For a remote service object the stub represents a local call, i.e. it is a local *proxy* for the remote service object.
proxy server	A specialized HTTP server application, usually running on a firewall computer and used to shield the internal network user addressing scheme from the external network by performing address translation.
publish-and-subscribe	An application registers (or subscribes) to the data outputs of other applications. The outputs are then automatically messaged to the subscribers.
query	Returns records that satisfy the query formula.
relational databases	Highly structured data that is accessed using *SQL.*
resources	Electronic documents, images, sound clips and programs.
RMI	Native *Java ORB* from *JavaSoft.* A competitor to CORBA.

APPENDIX A*

NOTES

A.1 Middleware Glossary (Continued)

RPC The program requests a procedure to be performed (e.g. open a file) by a remote service (e.g. a file server).

runtime Files needed at run-time to run an application, for example, with *Visual Basic.*

script Small program.

A set of instructions to an application or utility program. Various scripts can be invoked based on events, and allow the customization of applications.

OLE concept of controlling one application (component) by another. A "scripting language" is *Microsoft's Visual Basic for Applications (VBA).*

A "scripting language" is *Perl (Practical Extraction and Reporting Language)*, an interpreted language based on *C* and *UNIX*, which can assemble a string and send it to a shell as a command.

"Self-sufficient objects" use scripts to configure their actions at runtime.

session A connection between two logical units/communication services.

Represents a logically separate unit of screen, keyboard, and mouse activity, and the processes associated with these resources.

A session can have multiple conversations.

Applications take turns, using a single session.

sockets Transport-independent APIs for interfacing an application to the communication protocols, (the multivendor multiprotocol stacks, such as TCP/IP, IPX/SPX).

An example is: *Windows WinSock* for TCP/IP.

Session-layer network communications middleware: CORBA *ORBs* build on top of sockets.

APPENDIX A*

NOTES

A.1 Middleware Glossary (Continued)

SOM	*IBM's* language-independent CORBA-compliant *ORB*. Included in OpenDoc runtime.
SOMobjects	*IBM's* SOM-based implementation of CORBA.
SQL	A database language for defining, accessing, altering and protecting a relational database. The industry-standard English-like relational database query language.
standard	A rule (or set of rules) developed by a committee, or by industry acclaim (a de facto standard).
stub	Application implementation language interface for invoking object services.
TCO	Standardization, server-centric operation, centralized administration, reduced complexity, etc. considerations.
TCP/IP	A packet-oriented communication protocol suite, used to connect Internet computers.
thick client	A "full" PC.
thin client	A Network Computer (NC), usually diskless. In terms of application partitioning, two types are generally defined: <u>Client-oriented</u>, where *Java* applications are downloaded and run on the desktop under the control of a downloaded compact *OS*, which includes a *Java* VM, or <u>server-oriented</u>, where the applications are run on a server under a multi-user *Windows NT OS*. (Reference 6.3.1.)
threads	Provide, by means of priority clauses, the levels of multitasking within an OS. Threads run within processes, which in turn run within sessions. Separate procedures within the same program (or, process, when executing) that execute concurrently. An "event" can be assigned to a thread.

APPENDIX A*

NOTES

A.1 Middleware Glossary (Continued)

TP monitors	Manage transactions across C/S networks. Software which controls execution of transactions. Sold with every mainframe database.
transaction	A sequence of predefined actions, or, logical unit of work, performed on behalf of an application.
VPN	A link across the Internet, created by encapsulating IP (and other protocols) inside IP packets, for secure tunneling.
Web	The graphical portion of the Internet, called the World-Wide Web (WWW). A graphical C/S application environment. Specifically, a Global collection of servers running HTTP.
wrapper	Packages (or encapsulates) legacy code to make it accessible to populate an object state. It allows treating existing applications as services, which are requested by the new / developed applications or clients.

A.2 Acronyms and Definitions

4.3 BSD	*Berkeley* Software Distribution 4.3, *UNIX* version.
ACL	Server Access Control List
API	Application Programming Interface
AppleTalk	*Apple's* transport protocol stack.
BLOB	Binary Large Object data
C2	Command & Control
CCSDS	*Consultative Committee for Space Data Systems*
CIL	*Component Integration Laboratories, Inc.*
CGI	Common Gateway Interface
CLI	*SAG's* and *X/Open's* Call-Level Interface Specifications

APPENDIX A*

NOTES

A.2 *Acronyms and Definitions (Continued)*

COM	*Microsoft's* Component Object Model
CORBA	*OMG's* Common Object Request Broker Architecture
C&S	Command / Control & Status
COTS	Commercial Off-the-Shelf
CRADA	Cooperative Research and Development Agreement
DBMS	Database Management System
DCE	*OSF's* Distributed Computing Environment
DCOM	*Microsoft's* Distributed Component Object Model
DDE	*Microsoft's* Dynamic Data Exchange
DES	Data Encryption Standard
DFS	DCE's Distributed File System
DMI	Desktop Management Interface
DMS	Data Management System
DLL	Dynamic Link Library
DMTF	*Desktop Management Task Force*, industry consortium
DOM	Distributed Object Middleware
DRDA	*IBM's* Distributed Relational Data Architecture
DSOM	*IBM's* Distributed SOM, a CORBA ORB.
DSS	Decision Support System
DTP	*X/Open's* Distributed Transaction Processing, Reference Model

APPENDIX A*

NOTES

A.2 *Acronyms and Definitions (Continued)*

EDA/SQL	*Information Builders, Inc.'s Enterprise Data Access/SQL*
EDI	Electronic Data Interchange
ESQL	Embedded *SQL*
FAPs	Formats and Protocols, for inter-operation
GUI	Graphical User Interface
HTML	HyperText Markup Language
HTTP	HyperText Transport Protocol
IBI	*Information Builders, Inc.*
ICX	*Oracle's* Inter-Cartridge Exchange *ORB*
IDAPI	*Borland's* Integrated Database Application Programming Interface, API standard for SQL
IDL	CORBA's Interface Definition Language
IETF	*Internet Engineering Task Force*
IIOP	CORBA 2.0's Internet Inter-Orb Protocol, Internet application messaging standard (e.g. is built into *Netscape's Navigator 4.0* Web *browser.*
IIS	*Microsoft's* Internet Information Server
IP	Internet Protocol
IPC	Interprocess Communication
IPX/SPX	Internet Packet Exchange/Sequenced Packet Exchange, *Novell's NetWare* transport protocol stack
IS	Information Systems

APPENDIX A*

NOTES

A.2 Acronyms and Definitions (Continued)

ISO	*International Standards Organization*
ISP	Internet Service Provider
IT	Information Technology
JDBC	*Java* Database Connectivity, standard
JDK	*Java* Development Kit
JIT	Just-In-Time Compiler
JOE	*Sun's* Java Object Environment
JRMI	*Java* Remote Method Invocation, for *ORB* access
JRMP	*Java* Remote Method Protocol, RMI transport protocol
JVM	*Java Virtual Machine*
LAN	Local Area Network
LDAP	Lightweight Directory Access Protocol, for accessing online directory services over TCP
MAPI	*Microsoft's* Messaging API, addresses e-mail on the PC-desktop
MIB	SNMP's *Management Information Base*
MIF	SNMP's *Management Information File*
MIME	Multipurpose Internet Mail Extension, of SMTP
MOM	Message-Oriented Middleware
NC	Network Computer
NEO	Networked Object Extension to *Sun's Solaris* Operating Environment

APPENDIX A*

NOTES

A.2 Acronyms and Definitions (Continued)

NFS	*Sun's* Network File System
NOS	*Network Operating System*
OCI	*Oracle's* Call-Level Interface
ODBC	*Microsoft's* Open Database Connectivity, Windows API standard for *SQL*, the standard database transport interface Has a common API for many types of databases.
ODBMS	Object Database Management System
ODMG	*Object Database Management Group*, Vendor consortium
OFTP	Off-line Transaction Processing, replication processing
OLAP	Online Analytical Processing
OLE	*Microsoft's* Object Linking and Embedding
OLEDB	Encapsulation of Database access routines in OLE
OLTP	On-Line Transaction Processing
OMA	*OMG's* Object Management Architecture
OMG	*Object Management Group*
ONC	*Sun's* Open Network Computing Architecture
OO	Object-Oriented
OOP	Object-Oriented Programming
OOT	Object-Oriented Technology
OQL	*ODMG's* Object Query Language
ORB	Object Request Broker

APPENDIX A*

NOTES

A.2 Acronyms and Definitions (Continued)

OS	Operating System
OSF	*Open Software Foundation (X/Open)*
OSI	*ISO's* Open Systems Interconnection architecture, which has defined the popular OSI Reference Model communications protocol stack.
PSM	Publish/Subscribe Middleware
RDA	Remote Data Access
RDBMS	Relational Database Management System
RMI	*Java* Remote Method Invocation on objects, across *Java Virtual Machines*
RPC	Remote Procedure (Program) Call
SAG	*X/Open SQL Access Group*
S/MIME	Secure MIME, e-mail security protocol
SMTP	The Internet's Simple Mail Transfer Protocol
SNMP	Simple Network Management Protocol
SOM	*IBM's* Systems Object Model
S-HTTP	Secure HTTP, Web security protocol
SKIP	*Sun's* Simple Key management for Internet Protocol
SMI	SNMP's Structure of Management Information
SQL	Structured Query Language
SSL	*Netscape's* Secure Sockets Layer, Web security protocol, based on the RSA algorithm

APPENDIX A*

NOTES

A.2 Acronyms and Definitions (Continued)

STK	*Analytical Graphics'* Satellite Tool Kit
TCO	Total Cost of Ownership
TCP/IP	Transmission Control Protocol/Internet Protocol
TLS	Transport Layer Security, Web security protocol
TP	Transaction Processing
TPM	Transaction Processing Middleware
TT&C	Telemetry, Tracking & Command / Control
UDP/IP	The Web's TCP/IP User Datagram Protocol/Internet Protocol
URL	Unified Resource Locator
VIM	*Lotus's* Vendor Independent Messaging
VPN	Virtual Private Network by establishing a secure tunnel across the Internet
WAN	Wide Area Network
WOSA	*Microsoft Windows* Open Services Architecture
WRB	*Oracle's Web Request Broker*
WWW	World-Wide Web
XA	*X/Open* transaction interface protocol, used by TP Monitors

APPENDIX B

Alphabetical List of COTS Middleware Vendors*

* Includes Middleware Industry Consortiums and Standards Organizations

COTS Middleware Vendors*	for Processing	for Data Access	for Systems Management	for Internet
Active Software, Inc.		X		X
Apple Computer, Inc./ NeXT Software				X
Applix, Inc.	X			
BEA Systems, Inc.	X	X	X	X
Black & White Software	X		X	
Bluestone Software, Inc.		X		X
BMC Software, Inc.			X	
Boole & Babbage			X	
Bristol Technology, Inc.	X			
Compaq Computer Corp.			X	
Candle Corp.			X	
Component Integration Laboratories, Inc. (CIL)				
Computer Associates, Inc. (CA)		X	X	X
Covia Technologies	X			
Data Focus, Inc.	X	X		
DEC	X			
Desktop Management Task Force (DMTF)				
Expersoft Corp.	X			
IBM Corp.	X	X	X	X
Gradient Technologies, Inc.	X			X
HP	X	X	X	
I-Kinetics, Inc.	X	X		X
Information Builders, Inc. (IBI)	X	X		X
Informix Software, Inc.	X	X		X

APPENDIX B

Alphabetical List of COTS Middleware Vendors (Continued)

COTS Middleware Vendors (Continued)	for Processing	for Data Access	for Systems Management	for Internet
Internet Engineering Task Force (IETF)				
Intersolv	X	X		
Iona Technologies, Inc.	X		X	X
Message Oriented Middleware Association (MOMA)				
Microsoft Corp.	X	X		X
Momentum Software Corp.	X			
Netscape Communications Corp.	X	X		X
NetWeave Corp.	X			
Net Wise, Inc.	X			
Neuron Data, Inc.	X			
New Era of Networks, Inc. (NEON)	X			
NobleNet, Inc.	X	X		X
Novera Software, Inc.	X	X		
Object Management Group (OMG)				
Object Database Management Group (ODMG)				
OneWave, Inc.	X			X
Open Horizon, Inc.		X		
Oracle Corp.		X		X
ParcPlace Systems, Inc.	X			
Peer Logic, Inc.	X		X	X
Prolifics/JYACC, Inc.	X			
SCO		X		
Seer Technologies, Inc.	X	X		

APPENDIX B

Alphabetical List of COTS Middleware Vendors (Continued)

COTS Middleware Vendors (Continued)	for Processing	for Data Access	for Systems Management	for Internet
Simba Technologies, Inc.		X		
Software AG of North America, Inc.	X			
SQL Access Group (SAG)				
Softway Systems, Inc.	X	X		
Suite Software	X			
Sun Microsystems, Inc./ SunSoft	X	X	X	X
SuperCede, Inc.	X			
Sybase, Inc.	X	X		
Talarian Corp.	X		X	
Taligent, Inc.	X			
Thompson Software Products, Inc.	X			
Tibco Software, Inc.	X	X		
Tivoli Systems, Inc./IBM			X	
Transarc Corp.		X		
Trinzic Corp.	X			
Versant Object	X	X		
Visigenic Software, Inc.	X	X		X
Visual Edge Software, Ltd.	X			X
Wayfarer Communications Inc.				X
X/Open				

BIBLIOGRAPHY

Previous/Other Studies

A. B. Cottman and T. Morin, *Fault Tolerant Components for Space Mission Operations,* SBIR Phase I Final Report, I-Kinetics, Inc., 21 March 1997.

B. Aerojet companion study, Report 11173, *Uniform Interface for Multiple Satellite Systems.*

Documents, Publications, Texts

1. C. Burns, "NT and Unix: Friends, foes or co-conspirators?", *Network World,* April 14, 1997, p. 32. (Also on-line at: http://www.nwfusion.com.)

2. B. Brewin, "GROUPWARE: Lockheed revamps DMS architecture," *Federal Computing Week,* June 23, 1997, pp. 41, 44. (Also on-line at: http://www.fcw.com.)

3. B. Brewin, "MESSAGING: DISA scales back DMS 'grand design'," *Federal Computing Week,* June 16, 1997, p.--.

4. C. Gerber, "TECH BRIEFING: Windows NT and Unix battle for ground in federal agencies," *Federal Computing Week,* June 2, 1997, p.--.

5. D. Simpson, "Can NT Scale?", *DATAMATION,* March 15, 1996, p. 78.

6. D. Simpson, "Go SLOW With the PRO," *DATAMATION,* March 1, 1996, p. 65.

7. J. Cox and K. Essick, "Compaq riding NT's wave with Tandem merger," *Network World,* June 30, 1997, p. 6.

8. A. Berson, *Client / Server Architecture,* 2nd Ed., McGraw-Hill, New York, 1996.

9. R. Schreiber, 3-part series, "MIDDLEWARE Demystified," *DATAMATION,* April 1, 1995, p. 41; July 1, 1995, p. 57; and August 15, 1995, p. 41.

10. USAF Phillips Laboratory's WWW page: http:// www.plk.af.mil.

11. R. Orfali and D. Harkey, *Client/Server Programming with JAVA and CORBA,* J. Wiley & Sons, 1997.

12. R. Orfali, D. Harkey and J. Edwards, *The Essential Client/Server Survival Guide,* 2nd Ed., J. Wiley & Sons, 1996.

13. G. Lawton, "The COMPONENT WAR HEATS UP," *Software Magazine,* May 1997, p.51.

BIBLIOGRAPHY

Documents, Publications, Texts (Continued)

14. Tech Trends, "Buying the Right NC," *Communications Week*, February 10, 1997, p.1.

15. C. Tristram, "Middleware Makes C/S Apps Really Work," *DATAMATION*, August, 1996, p. 78.

16. R. Sudama, "Get Ready for Distributed Objects," *DATAMATION*, October 1, 1995, p. 67.

17. E. Booker, "Using Middleware to Link Business Units," *Internet Week*, February 19, 1995, p. 25.

18. P. Ruber, "Prime Time for Middleware," *LAN TIMES*, August 28, 1995, p. 72.

19. J. Moore, "Middleware eases client/server challenges," *Federal Computing Week*, January 8, 1996, p. 56.

20. RS/News, "RS/6000 Links Mobile Networks," *RS/Magazine*, May 1996, p. 10.

21. B. Robertson, "Coming To The Rescue of Wireless Middleware," *Network Computing*, June 15, 1996, p. 101.

22. R. Tackett, "Webified database servers," *Network World*, December 9, 1996, p. 45.

23. W. Stevens, *UNIX Network Programming*, Prentice-Hall, New Jersey, 1990.

24. R. Yasin, "IBM Unveils Secure E-Business Products," *Internet Week*, May 11, 1998, p. 16.

25. V. McCarthy, "Building a Firewall," *DATAMATION*, May 15, 1996, p. 74.

The Object-Oriented Paradigm

a. J. Rumbaugh, et.al., *Object-Oriented Modeling and Design*, Prentice-Hall, New Jersey, 1991.

b. B. Rao, *Object-Oriented Databases, McGraw-Hill*, New York, 1994.

Further Reading

aa. K. Watterson, "Turbocharging Object Projects," *Sun Expert Magazine*, January, 1998, p. 50: For a list of books on the subject.

DISTRIBUTION LIST

AUL/LSE Bldg 1405 - 600 Chennault Circle Maxwell AFB, AL 36112-6424	1 cy
DTIC/OCP 8725 John J. Kingman Rd, Suite 0944 Ft Belvoir, VA 22060-6218	2 cys
AFSAA/SAI 1580 Air Force Pentagon Washington, DC 20330-1580	1 cy
AFRL/PSTL Kirtland AFB, NM 87117-5776	2 cys
AFRL/PSTP Kirtland AFB, NM 87117-5776	1 cy
GenCorp Aerojet P.O. Box 296 1100 West Hollyvale St Azusa, CA, 91702-0296	1 cy
AFRL/VS/Dr Fender Kirtland AFB, NM 87117-5776	1 cy
Official Record Copy AFRL/VSSS/George Schneiderman Kirtland AFB, NM 87117-5776	2 cys
SMC/CW 155 Discoverer Blvd El Segundo, CA 90245-4692	1 cy
SMC/XR 180 Skynet Way 2234 El Segundo, CA 90245-4687	1 cy
Lockheed-Martin Astronautics Flight Systems Attn: Dr. Noel W. Hinners, MS S80000 12257 State Highway 121 Littleton, CO 80217	1 cy

TRW Space and Electronics
Attn: Joanne McGuire
1 Space Park
Bldg R10, Rm 2826
Redondo Beach, CA 90278 1 cy

Boeing
Attn: Mike Mott
2201 Seal Beach Blvd
Seal Beach CA 90740 1 cy

Hughes Space and Communications
Attn: Dr. Tom Brackey, MS S312
2260 E. Imperial Highway
El Segundo, CA 90245 1 cy