An Open, Distributable, Three-tier Client-Server Architecture with Transaction Semantics

by

Norman G. Depledge, William A. Turner, and Alexandra Woog

ABSTRACT

This paper describes a distributable, three-tier client-server architecture for heterogeneous, multivendor environments based on the integration of Digital's ObjectBroker and ACMSxp transaction processing monitor products. ObjectBroker integration software provides the flexibility to decouple the tight association between desktop devices and specific legacy systems. The ACMSxp transaction processing monitor provides the transaction semantics, system management, scalabilty, and high availability that mission-critical production systems require. Combining these technologies and products in a three-tier architecture provides a strategic direction for the development of new applications and allows for optimal integration of legacy systems. The architecture complies with industry standards, which facilitates vendor independence and ensures the longevity of the solution.

INTRODUCTION

Almost all large global enterprises have developed separate systems to address specific business needs. Frequently, these systems are on disparate platforms from different vendors. Users may have to log in to several systems in order to process a single service request from a customer. To improve customer service and develop new products, new applications must integrate existing environments and must be capable of accessing and integrating data from existing platforms.

End users may be faced with an array of inconsistent and incompatible user interfaces that are difficult to learn to use. This source of inefficiency directly impacts the level and cost of service provided to customers and the time-to-market for new products and services.

An analysis of the above problems leads to some fundamental conclusions about existing business systems in large enterprises. Generally, the in-place applications are mission-critical legacy systems that record transactions performed by the businesses. These systems demand superior transactional integrity and operational reliability. They serve hundreds to thousands of users and yet provide good response at high levels of performance. Systems designers will not introduce changes to them that would compromise these exacting requirements. Consequently, enterprises do not readily replace their legacy systems but, instead, look for other solutions that integrate them with new systems.

To improve the effectiveness of existing legacy systems, major enterprises are seeking to reengineer the user interface. The goal is to reface the applications with a modern, consistent, easy-to-use interface that directly reflects the users' and customers' needs. The new interface must be fully articulated; that is, it should allow any desktop to access any permitted application, regardless of its location or the platform on which it is running. The solution should allow the composition of new compound business functions by combining existing application transactions from multiple legacy systems and possibly new or downsized applications. The new user interface should accomplish this without disrupting the level of service provided to the users.

All these requirements indicate the need for an intermediate architectural layer that provides for isolation, switching, transaction semantics, composition of function, and location transparency. The resultant architecture has three tiers: the clients, the intermediate layer, and the existing legacy systems and new servers.

Such an architecture is expected to last a considerable number of years. It is, therefore, essential that the architecture be based on modern but stable technologies and be flexible enough to accommodate technology evolution.

THE THREE-TIER ARCHITECTURE

The three-tier architecture consists of the following separate layers of systems and software:

1. Clients

2. Transactional middleware

3. Systems of record (legacy systems and new systems)

The attributes of the proposed intermediate layer make this three-tier architecture more flexible than traditional two-tier client-server architectures.

Tier 1 systems (clients) provide a desktop graphical user interface (GUI) to the end users. These systems have seamless access to a set of abstract transaction services in a location-transparent manner through an object request broker (ORB). The interface between tier 1 and tier 2 operates in a client-server manner. The security services in the ORB provide a security perimeter around the client.

Tier 2 is the middleware layer that provides application services to the clients. These services appear to the clients as business functions and may be transactional in nature.

A single tier 2 business function can be composed of one or more transactions residing in different applications on tier 3. The location and native interfaces of these applications are hidden from the tier 1 desktop GUI clients by the tier 2 middleware.

Tier 2 is designed to support the evolution of the application interfaces and protocols between tier 2 and tier 3. These interfaces will change as new technologies such as the Open Software Foundation's (OSF's) Distributed Computing Environment (DCE) and the Object Management Group's (OMG's) Common Object Request Broker Architecture (CORBA) mature and become more widely available, and as tier 3 applications are modified or new ones added.[1,2] Figure 1 shows the disposition of functions with intertier communications paradigms.

Figure 1 Disposition of Functions with Intertier Communications
          Paradigms

```
                                         INTERFACE COMPLIANT
   TIER 1       CORBA-       TIER 2      WITH MESSAGE-BASED     TIER 3
                COMPLIANT                  PROTOCOLS, DCE,
+------------ + INTERFACE +---------------+  OR CORBA    +----------------+
| DESKTOP GUI |<--------->| TRANSACTIONAL |<------------>| NEW AND LEGACY |
| CLIENTS     |           | MIDDLEWARE    |              | SYSTEMS        |
+-------------+           +---------------+              +----------------+
```

A STANDARDS-BASED ARCHITECTURE

Digital implemented the three-tier architecture using standards-based software to offer the highest level of interoperation with systems offered by other standards-compliant vendors. Standards compliance also facilitates the porting of applications across platforms.

The standards organizations most relevant to this architecture are

- International Organization for Standardization (ISO)
- American National Standards Institute (ANSI)
- Open Software Foundation
- Object Management Group
- X/Open Company Limited
- Nippon Telegraph and Telephone's (NTT's) Multivendor Integration Architecture (MIA) and the Network Management Forum (NMF's) Service Providers' Integrated Requirements for Information Technology (SPIRIT), together referred to in this paper as MIA/SPIRIT[3,4]

ISO and ANSI are true standards bodies. The other organizations are either influential industry consortia aimed at defining common standards for important emerging and maturing technologies driven by user needs or customer-driven consortia chartered to define common purchasing standards backed by substantial purchasing power.

OSF's Distributed Computing Environment

The Open Software Foundation's Distributed Computing Environment is an important standard that defines a set of services and tools that support the creation, use, and maintenance of client-server applications in heterogeneous multivendor environments. The OSF has defined and assembled this technology; Digital is a major provider of components. As illustrated in Figure 2, the components of OSF's DCE are

- Remote procedure calls (RPCs)
- DCE threads, which is a standardized multithreading service
- Distributed time service, which synchronizes clock time

across globally distributed systems

- Cell directory service (CDS), which provides authentication, access control, and encryption, and uses a Kerberos-based private key security model

- Global directory service, which provides directory services between cells using the X.500 standard

- Distributed file service, which provides location-transparent access to files across a network

DCE has been rapidly adopted as a technology for distributed systems and is now available on a large number of vendor platforms, including Digital, IBM, Hewlett-Packard, Sun, and Microsoft.

[Figure 2 (OSF's Distributed Computing Environment) is not available in ASCII format.]

CORBA

The Common Object Request Broker Architecture is a standard specification for the central communication and integration of software objects at the enterprise level and across enterprises. CORBA and its specification were developed by the OMG, a consortium of information systems vendors, including Digital, Hewlett-Packard, HyperDesk, Symbios Logic (formerly NCR), Object Design, and SunSoft.[5] Digital's CORBA-compliant product, namely ObjectBroker integration software, has been ported to the industry's leading range of platforms.[6]

The ObjectBroker product reduces the time and costs associated with providing access to critical business applications across multivendor platforms. It allows legacy applications to be integrated into heterogeneous client-server environments without source code changes.

Microsoft Corporation has developed a parallel approach as evidenced in its Object Linking and Embedding (OLE) software, which is focused on integrating objects in a desktop environment.[7] Microsoft and Digital are working to integrate the CORBA and OLE software into a combined architecture called the Common Object Model (COM), which allows the full interoperation of applications developed under either constituent architecture.

X/Open Distributed Transaction Processing

The X/Open distributed transaction processing (DTP) committee is defining standards for DTP systems that use flat transactions. In Figure 3, the TX interface allows applications to coordinate global transactions via the transaction manager (TM); the XA interface connects the TM to resource managers (RMs), typically relational databases or file systems; and the XA+ interface connects the TM to communications resource managers (CRMs). The interface between an application and a CRM is specific to the CRM type, of which three are defined.

- Transactional remote procedure call (TxRPC), which is derived from the work led by Digital for the MIA/SPIRIT remote task invocation protocol (discussed in more detail later in this section).

- XATMI, which is a non-RPC-based client-server that originated with Unix System Laboratories' transaction processing monitor for the UNIX operating system, namely the Tuxedo product.

- Peer-to-peer, by which messages are exchanged between applications. The messages are sent and received in an order based on prior agreement between the implementers of the applications. Peer-to-peer uses Common Programming Interface for Communications (CPI-C), which is derived from IBM's System Network Architecture (SNA) message-based protocol of the same name.

[Figure 3 (X/Open Distributed Transaction Processing Model) is not available in ASCII format.]

MIA/SPIRIT

MIA is a software architecture developed by a consortium of five vendors under the sponsorship of NTT: Digital, IBM, Fujitsu, Hitachi, and NEC. MIA adopted existing industry standards and defined standards in areas where none were available. One of the areas most lacking in standards was DTP. NTT requested technology proposals and received responses from all the vendors in the consortium. Digital submitted its Application Control and Management System (ACMS) transaction processing monitor model and was selected to lead the development of the specifications because of ACMS' modern, highly structured model and transaction processing application programming interface (API).

MIA achieves application portability and interoperability across a variety of vendor operating systems and platforms by using standardized APIs as integrative constructs and by using standardized systems interconnection interfaces (SIIs) for communication.

Two significant MIA standards that Digital contributed are

- Structured Transaction Definition Language (STDL), which is a high-level programming language suited to transactional client-server programming [8,9]

- Remote task invocation (RTI), a service definition and protocol for RPCs that are in a multivendor environment and that use the two-phase commit protocol

As a follow-on to NTT's MIA, the work in the field of transaction processing standards has passed to the SPIRIT consortium, which is managed by the Network Management Forum. NMF's list of members includes telecommunications service providers, such as AT&T, BT, Deutsche Telekom, ETIS (itself a consortium that represents 27 European Postal, Telegraph, and Telephone Administrations), France Telecom, KDD, Telecom Italia, and Telefonica; computer vendors, such as Digital, Hewlett-Packard, Fujitsu/ICL, Hitachi, IBM, NEC, Siemens Nixdorf, and Unisys; and software vendors, such as Microsoft and Oracle. The goal of the SPIRIT consortium is to produce a common, agreed-upon set of specifications for a general-purpose computing platform for the telecommunications industry by July 1995. The combined annual computing expenditures are estimated to exceed $20 billion.

MIA/SPIRIT standards are working their way into international standards bodies. X/Open and the NMF have extended their collaborative agreement to include the work of SPIRIT in acknowledgment of the difficulties that diverging standards would create. X/Open publishes the SPIRIT documentation alongside its own CAE specifications and guides. Furthermore, after conducting a survey of major transaction processing users, X/Open recently voted to use its fast-tracking process to accelerate progress in the adoption of STDL as an X/Open standard.

Digital delivered a platform that supports STDL in January 1993, IBM offered STDL on the CICS platform in the second quarter of 1993, and Hewlett-Packard has made STDL available on Transarc Corporation's Encina transaction processing monitor. NEC, Hitachi, and Fujitsu have already shipped STDL platforms. Unisys plans to demonstrate a SPIRIT platform with STDL in October 1995.

In July 1994, an interoperability demonstration using STDL was conducted successfully in Tokyo, Japan. The demonstration, which also included RTI, involved systems provided by Hewlett-Packard and Fujitsu on Transarc Corporation's Encina transaction processing monitor, Digital on its Application Control and Management System/Cross-platform (ACMSxp) transaction processing monitor, and IBM on both the MVS/CICS and OS/2 platforms.

ARCHITECTURE COMPONENTS

Figure 4 illustrates the overall three-tier client-server architecture. This section discusses the various components.

[Figure 4 (Overall Three-tier Client-Server Architecture) is not available in ASCII format.]

Tier 1 Desktop Environment

The architecture must provide for the connection of a wide variety of desktop platforms to the server layer, i.e., the tier 2 middleware services. This connection must be accomplished in a secure, extensible, reliable, and location-transparent manner. Standards-based solutions are always desirable and more effective over the multiyear life of an enterprise-wide solution. Digital therefore selected its CORBA-compliant ObjectBroker software as the mechanism to connect tier 1 clients to tier 2 middleware servers.

CORBA provides a flexible approach to developing a distributed application by decoupling the client and server portions of the application. CORBA specifies a common set of interfaces that allows client programs to make requests to and receive responses from server programs without direct knowledge of the information source or its location. CORBA defines the ORB as an intermediary between clients and servers that delivers client requests to the appropriate server and returns the server responses to the requesting client. Figure 5 shows how the ORB allows a client application to request a service without knowing where the server is located or how it will fulfill the request.

[Figure 5 (CORBA Client-Server Request/Response Flow) is not available in ASCII format.]

In the CORBA model, client applications need to know only what requests they can make and how to make the requests; they do not need to be coded with any implementation details about the server. Server programs need to know how to fulfill the requests but not how to return information to the client program. Clients using objects to request a service do not need to know which server will fulfill that request. The server fulfilling the request does not need to know which client initiated the request. The GUI clients can be developed using any tool that provides a call-level interface or an object-oriented interface to CORBA-compliant client services on the specific platform.

Communications are conducted through RPCs. The RPCs are carried over a network transport, e.g., a transmission control protocol/internet protocol (TCP/IP) or a DECnet transport. The RPC connects with ObjectBroker's ORB, which then reroutes the RPC directly to the selected service instance.

Digital expects future versions of its CORBA-compliant ObjectBroker product to support OSF's DCE and thus provide standards-based directory, security, and RPC services. DCE provides rigorous security services for authenticating users, granting privileges, and controlling access to important networked resources. These services are based on the highly secure Kerberos model, which is the standard security model for many financial institutions and a major reason why they have standardized on DCE. All interaction from tier 1 clients must go through the Kerberos-based DCE security perimeter. Desktop and mobile computer users log in to the DCE cell to gain their credentials for performing their business. DCE authenticates users and grants them the appropriate privileges and controlled access to the authorized business functions. No clear-text passwords are required, even for mobile users who access the middleware layer by means of dial-up lines. Remote or mobile users are able to perform DCE login over a serial line internet protocol (SLIP) connection. Confidentiality is ensured by data encryption.

Tier 2 Middleware Services

The tier 2 middleware of this architecture is founded on the ACMSxp transaction processing monitor. The ACMSxp software product for transactional applications conforms to the X/Open DTP and MIA/SPIRIT standards previously described. The software is layered on DCE and the transaction manager and Structured File Services of the Encina toolkit. The primary API to the ACMSxp product is STDL, as defined by MIA/SPIRIT.

STDL is used to define the control flow and transaction demarcation of applications in a highly structured and modular fashion. In addition, STDL supports the features needed for RPC-based DTP.

The ACMSxp software is structured in a three-part model, which is shown in Figure 6. The model separates client presentation functions from transaction flow control and data access and processing functions, which are typically SQL database code or communications code to access legacy systems. A very important feature of ACMSxp is that the STDL compiler generates complete DCE client-server stubs, thus freeing the developer from having to perform complex programming at the DCE level.

[Figure 6 (ACMSxp Three-part Model) is not available in ASCII format.]

ACMSxp applications comprise clients that call application task servers using DCE RPCs. The application task servers execute the STDL programs in DCE multithreaded processes, maintaining the context of the users and performing the control flow and transaction semantics (two-phase commit protocol). Application tasks are free to call other tasks in the same group or in remote task groups that are reached through DCE directory services, either transactional (composable) or nontransactional (noncomposable). Tasks can also call processing procedures that are grouped in single-threaded procedure servers. Processing procedures normally provide data access, typically by means of C, COBOL, and SQL in conjunction with X/Open DTP-compliant databases. Processing procedures are also ideal for applications that communicate to legacy systems by means of message-based protocols, such as IBM's SNA Logical Unit 6.2 (LU 6.2).

Figure 7 shows client tasks accessing data by means of Digital ACMSxp servers. Note that ACMSxp supports and manages pools (groups) of servers in a given class, i.e., servers that provide the same set of services. Through the system management interface, operators can set and change dynamically the desired number of servers in a given pool. If a server fails, ACMSxp system management logs the event and automatically starts and activates a replacement server.

[Figure 7 (ACMSxp Application Components) is not available in ACSII format.]

Digital's ACMSxp transaction processing monitor, in conjunction with the ObjectBroker software, is used to implement the tier 2 middleware layer. ACMSxp clients are no more than simple business function call statements stripped of DCE programming by the STDL client run-time services. ACMSxp clients are encapsulated in wrappers to make them CORBA objects accessible through the ObjectBroker software. These wrappered client processes can be declared as servers to the sophisticated ACMSxp transaction processing monitor system management. The system management can then be instructed to set dynamically the desired number of instances, to maintain that number in the event of failures, and to raise alerts. These ACMSxp client processes are wrappered to include the code that registers them as objects with the ORB as instances of CORBA implementations. When activated by ACMSxp system management, these processes register with the ORB and wait for requests for the tasks that they invoke. When stopped by ACMSxp system management, the processes deregister from the ORB.

This architecture can be instantiated using Digital's CORBA-compliant ObjectBroker product with the ACMSxp transaction processing monitor, presenting the desktop clients on tier 1 with an object-oriented interface to business functions executing on tier 2. As shown in Figure 4, ObjectBroker is substituted for CORBA and tier 2 is implemented on the ACMSxp transaction processing monitor. An unlimited number of tier 2 nodes can be configured dynamically, started, and stopped. Transactional business functions can be distributed and replicated across the nodes as required. The clients gain access to these business functions in a completely location-transparent manner. The interface to legacy systems is provided by application code that resides in processing servers.

The business functions are written as STDL tasks and can be composed of multiple legacy application transactions. When tier 3 applications support standards-compliant TxRPC, transactions can be called directly as tasks in STDL from the tier 2 business functions.

Security into tier 2 is handled by the ObjectBroker software. Within tier 2, security is enforced according to the rules of

OSF's DCE. Security between tier 2 and tier 3 is mandated by the rules of each specific legacy system.

To provide operational support for production applications, sophisticated system management features were built into the ACMSxp product. A system management interface is available to any authorized operator on any node in the DCE cell. Through a single director, all ACMSxp objects can be managed in multiple transaction processing systems on all nodes in the network. The managed objects include transaction processing systems, event logs, request sessions, servers, processes, interfaces, and procedures. For example, system managers can examine and change the properties and execution state of servers. The number of instances of a given server class can be set and changed dynamically without stopping the system. ACMSxp system management can be induced to adopt servers that are normally external to its domain, such as the ObjectBroker method servers that provide the connection between the desktop clients and the transactional task servers in the ACMSxp product.

Tier 3 Legacy Application Interfaces

Intercommunications issues related to the differences between hardware and software architectures on disparate platforms are addressed by technologies such as DCE. DCE supports RPCs that enable applications on different platforms to interoperate by means of simple call statements with fully typed arguments. Data type differences between hardware architectures are bridged by the marshaling process that converts data to a canonical form and then to the target form as a normal process. Message-based protocols, such as LU 6.2, cannot adequately deal with mixed data types and place a burden on the application programmer in a multivendor environment.

The advent of reduced instruction set computer (RISC) architectures has exacerbated these problems. Gaps are frequently left in memory between variables in structures and records that contain mixed data types. These gaps in buffers, when processed by compilers on RISC machines, render the buffers unmappable unless redundant filler variables are added to the structure definitions.

Each legacy application method is encapsulated in an ACMSxp server class that is invoked transactionally by a simple STDL call. Thus, the developer of the STDL transactional business functions is shielded from the complexities of the native interface to the legacy data. This approach permits future update of the method without affecting the existing business functions.

The designer must select the most appropriate communications protocol for each tier 3 legacy system. Whenever possible, an application interface should be selected that avoids the so-called "screen scraping" techniques, in which the application emulates a user interacting with existing terminal screen forms.

For IBM mainframe systems, the SNA LU 6.2 protocol with Syncpoint Level 1 or 2 is often appropriate for interoperating with IBM transaction processing environments. This protocol may also be the appropriate choice for legacy systems from other vendors. If the application message protocol is designed in a manner that simulates a simple procedure call, future migration to an RPC model will be simplified.

Recently, IBM has made DCE available on MVS---OpenEdition and has provided application support for both the CICS and Information Management System (IMS) transaction processing environments. This feature allows DCE client programs to invoke transactions on the IBM mainframe by way of a DCE application server provided by IBM. An appropriate DCE client could be included in a data access processing procedure of an ACMSxp processing server as an alternative to SNA LU 6.2.

It should be noted that the desired throughput level for a given legacy system connection can be adjusted dynamically. An operator can use the system management of the ACMSxp transaction processing monitor to reset the number of active servers in the pool that implements that connection. Also, any number of tier 2 nodes can be configured to provide that service within the middleware layer. New nodes can be placed in service without interrupting currently running nodes.

SUMMARY

A three-tier, object-oriented client-server architecture that includes an open systems transaction processing monitor can provide a basis for connecting users and customers to existing enterprise transaction processing systems by means of reengineered desktop systems that support GUIs. This approach provides

- A clear separation of function, i.e., client activities are separate from middleware control and management functions

- Data location transparency

- Location transparency for application interfaces and topological independence

- A means of defining new business functions by compounding existing transactions on different platforms, regardless of location

- Flexibility to support the continuous evolution of systems without disruption to end users

- Resilience to enhance overall availability

- Unrestricted scaling of the system (through replication of components) for performance adapted to the business growth

- A set of reusable objects to the tier 1 client

REFERENCES AND NOTE

1. Open Software Foundation DCE Application Development Guide, Document No. ED-DCEAPDEV1-1092-2 (Cambridge, Massachusetts: Open Software Foundation, 1993).

2. Common Object Request Broker Architecture Specification, draft 29, revision 1.2 (Framingham, Massachusetts: Object Management Group, Document No. 93-12-43, December 1993).

3. Multivendor Integration Architecture Technical Requirements, Division 1, Overview (Tokyo, Japan: Nippon Telegraph and Telephone Corporation Technology Research Department, Order No. TR550001, 1991).

4. X/Open Consortium Specification, SPIRIT Platform Blueprint (SPIRIT Issue 2.0, Volume 1.0), ISBN 1-85912-059-8 (Reading, U.K.: X/Open Company Ltd., Network Management Forum, Document No. J401, 1994).

5. Symbios Logic is the former NCR Microelectronic Products Division of AT&T Global Information Solutions Company.

6. ObjectBroker Overview and Glossary (Maynard, Massachusetts: Digital Equipment Corporation, Order No. AA-Q9KJA-TK, 1994).

7. Microsoft Windows & MS-DOS Version 6.0, Chapter 11 (Redmond, Washington: Microsoft Press, 1993).

8. SPIRIT Platform Blueprint, SPIRIT STDL Language Specification Platform (SPIRIT Issue 2.0, Volume 3.0), ISBN 1-85912-063-6 (Reading, U.K.: X/Open Company Ltd., Network Management Forum, Document No. J403, 1994).

9. SPIRIT Platform Blueprint, SPIRIT STDL Environment, Execution and Protocol Mapping Specification (SPIRIT Issue 2.0, Volume 3.0), ISBN 1-85912-064-4 (Reading, U.K.: X/Open Company Ltd., Network Management Forum, Document No. J404, 1994).

BIOGRAPHIES

Norman G. Depledge

As manager of the Transaction Processing Design Consulting Group within Digital's layered software organization, Norman Depledge is responsible for managing the technology transfer interface between the TP engineering functions, the field organization, and strategic customers on a worldwide basis. His background in computers spans 33 years. He has held management positions in electrical engineering, software engineering, and marketing. For the past 18 years, he has specialized in on-line transaction processing at Honeywell, Bull, and Digital. Norman has an Honors Degree in electrical engineering from Manchester University, England, and holds three patents in electronic controls.

William A. Turner

William Turner is a consultant in the Transaction Processing Systems Group. He successfully constructed a working model and demonstration of an open, distributable, three-tier client-server architecture with transaction semantics. In previous work, William was a consultant in the Northeast Region and in the New York Production Systems Resource Center. Before joining Digital in 1987, he held positions as a systems manager for Electric Mutual and as a technical support manager for Honeywell Information Systems. William received a B.S. in mathematics from Villanova University in 1966.

Alexandra Woog

As a consultant with Digital's Transaction Processing Systems Group, Alexandra Woog consults with customers on the design and implementation of production systems. In prior work, she was the product manager for Digital's Remote Transaction Router (RTR, in Switzerland), a unit manager for consulting services, and an operating systems specialist.

TRADEMARKS

ACMS, ACMSxp, DEC OSF/1, DECnet, Digital, ObjectBroker, and VMS are trademarks of Digital Equipment Corporation.

AT&T is a registered trademark of American Telephone and Telegraph Company.

BT is a registered trademark of British Telecommunications plc.

Encina is a registered trademark of Transarc Corporation.

Hewlett-Packard is a registered trademark of Hewlett-Packard Company.

IBM and OS/2 are registered trademarks and CICS is a trademark of International Business Machines Corporation.

Kerberos is a trademark of Massachusetts Institute of Technology.

Microsoft is a registered trademark of Microsoft Corporation.

NCR is a registered trademark of NCR Corporation.

Oracle is a registered trademark of Oracle Corporation.

OSF and OSF/1 are registered trademarks of the Open Software Foundation, Inc.

Sun is a registered trademark and SunSoft is a trademark of Sun Microsystems, Inc.

Tuxedo is a registered trademark of Unix Systems Laboratories, Inc.

UNIX is a registered trademark in the United States and other countries, licensed exclusively through X/Open Company Ltd.

X/Open is a trademark of X/Open Company Ltd.