

# ClearPath<sup>®</sup> as an Open System

## A White Paper

By Peter Bye

©2010 Bye Associates

All rights reserved, except that this White Paper may be freely distributed provided no charge is made and that it is distributed unaltered, in its entirety, including this notice.

Unisys and the Unisys logo are registered trademarks of Unisys Corporation. All other brands and products referenced herein are acknowledged to be trademarks or registered trademarks of their respective holders.

## Table of Contents

Executive summary .....	4
Introduction.....	5
What is an open system?.....	5
Perspectives on openness – an analytical framework .....	8
Evaluating ClearPath system open attributes .....	9
Perspective 1: internal interfaces .....	9
Application environments .....	10
Files, databases and storage .....	11
Perspective 2: application development .....	12
Perspective 3: external interfaces .....	13
Communications .....	13
Distributed application servers .....	14
Middleware .....	14
Distributed transactions.....	16
Perspective 4: systems management .....	16
ClearPath system mission-critical attributes .....	17
Comparisons with other platforms: a summary.....	18
More information .....	20
Appendix: glossary of technical terms and products .....	21
Technical terms and acronyms .....	21
Products .....	23
About the author.....	25

## Executive summary

Definitions of an open system from independent bodies such as The Open Group and the World Wide Web Consortium (W3C) centre on the internal and external interfaces it exposes, not on the hardware or operating system platform. 'Open' in this context means support for industry-standard interfaces.

The reasons behind this focus include:

- ❑ Current application environments, especially Java, shield the underlying platform from applications, allowing a very high degree of portability. The platform is not relevant to openness.
- ❑ Systems are increasingly collaborating with others in distributed applications, both intra- and inter-organisation. Inter-system communication across independent groups or organisations requires industry-standard interfaces that make no assumptions about the technologies used to develop and deploy the various systems. Web Services is an example of such a technology.

Based on these definitions, it is clear that 'openness' encompasses a spectrum of open interfaces; a system is not 'open' or 'closed' but somewhere in-between. At one end of the spectrum, there are few open interfaces; at the other end, there are many. Adding more interfaces makes a system more open.

ClearPath systems are open in that they are at the 'open end' of the spectrum. They support an extensive range of standard and *de facto* standard interfaces, allowing import and export of software, and especially participation in distributed environments, including:

- ❑ Implementations of industry-standard application environments, including Java, Java EE, and Open Group DTP. Support for Java allows easy importing of software written in Java. These environments are integrated with the native application servers COMS and TIP/HVTIP, and can access native databases and other data stores.
- ❑ Support for industry-standard communications protocols and FIPS certified encryption algorithms.
- ❑ Provision of all the major industry-standard middleware technologies, enabling a wide range of collaboration options, including participation in a service-oriented architecture (SOA) implementation. The range of middleware supported is particularly rich.
- ❑ Support for distributed transactions across multiple systems of different types.

However, openness is not the only desirable system attribute. ClearPath systems are typically used in mission-critical environments such as high volume transaction processing. Attributes such as reliability, efficiency, responsiveness to rapidly changing conditions and security are essential. Security is particularly critical. If it is compromised, especially if valuable information is damaged or extracted from a database and used for fraudulent purposes, the results can be catastrophic.

ClearPath systems are particularly strong in these attributes, largely due to the integrated software stack. All the necessary components, from operating system to application interface, are optimised for performance and tested together before release. This avoids the need for complex and expensive integration by users, which is required if the products come from a number of different sources, as is typical with many systems regarded as open. The result not only delivers the required mission-critical attributes but is also very cost/effective and allows customers to move more quickly to new software releases. In fact, there is increasing interest in supplying integrated software stacks; it is clear, for instance, that this is the strategy of Oracle and others. Unisys has been providing integrated software stacks in ClearPath systems for many years.

The combination of mission-critical *and* open attributes results in systems which are powerful and secure transaction processing platforms, *and* are able to participate in distributed environments, including SOA implementations. These unique characteristics differentiate ClearPath systems from other platforms commonly viewed as open.

## Introduction

This paper substantiates the view of ClearPath open and mission-critical attributes stated in the executive summary. It examines the open attributes of ClearPath systems in the light of the widely-accepted standard and *de facto* standard interfaces that determine openness. And it considers the role of ClearPath mission-critical attributes in contributing to the systems' primary role as an enterprise server in high performance, secure transaction processing.

The remainder of the paper is divided into the following sections:

- ❑ The first section reviews definitions of open systems in some detail. The definitions originate from a number of well-known independent organisations.
- ❑ Using the definitions as a starting point, a framework for analysing the open attributes of a system is developed.
- ❑ The open attributes of ClearPath systems are then analysed using the framework.
- ❑ The mission-critical attributes of ClearPath systems are then discussed.
- ❑ How do ClearPath systems compare with others? The next section contains a brief summary of comparisons with other platforms and how they support some key open attributes.
- ❑ Finally, some pointers to sources of additional information are provided.
- ❑ A glossary of technical terms used and products mentioned is appended.

## What is an open system?

Definitions of open systems have changed over the years. An early view was that an open system was one where the hardware was obtainable from more than one source. In particular, the availability of IBM plug-compatible hardware systems was regarded as making the S360 and S370 systems open because they were alternative sources to IBM. Later interpretations of open have inclined much more towards software, in particular operating systems (primarily UNIX).

Two factors have affected views of open. First, current application environments, especially Java, provide a high level of abstraction from the underlying environment, making the hardware and operating system platform irrelevant to openness; the internal interfaces are what matter. The second factor has been the rise of distributed intra- and inter-organisational environments, for example in an SOA implementation. Inter-system communication across independent groups or organisations requires industry-standard external interfaces that make no assumptions about the technologies used to develop and deploy the various systems.

Independent organisations – those with no relationship to any hardware or software vendor – focus on the interface characteristics of the system rather than the specific technologies used. For example, a view advanced by the Carnegie Mellon Software Engineering Institute<sup>1</sup> (SEI) some years ago was

*'An "open system" generally calls to mind a system that is flexible and adaptive, and "open" to the use of many products from different sources. To many people, the phrase "open system" carries with it an image of easy "plug-and-play" between components and products that were not necessarily originally designed to work together.'*

The Carnegie Mellon paper refers to a definition used by the US Department of Defense Open Systems Joint Task Force (OS-JTF), which defined an open system as

*'A system that implements sufficient open specifications for interfaces, services, and supporting formats to enable properly engineered components to be utilized across a wide range of systems with minimal changes, to interoperate with other components on local and remote systems, and to interact with users in a style that facilitates portability.'*

---

<sup>1</sup> SEI Monographs on the Use of Commercial Software in Government Systems. COTS and Open Systems (Patricia Oberndorf, February 1998) See <http://www.sei.cmu.edu/library/assets/cotsopensystems.pdf>

This interpretation was later revised by the OS-JTF<sup>2</sup> to the following (current) definition of an open systems as

*'A system that employs modular design, uses widely supported and consensus-based standards for its key interfaces, and has been subjected to successful validation and verification tests to ensure the openness of its key interfaces.'*

Other well-known organisations take a similar line. Even The Open Group<sup>3</sup>, which often tends to favour UNIX, defines an open system without any operating system reference as

*'A system that implements sufficient open specifications for interfaces, services, and supporting formats to enable properly engineered Application Software:*

- a) To be ported with minimal changes across a wide range of systems*
- b) To interoperate with other applications on local and remote systems*
- c) To interact with users in a style that facilitates user portability.'*

As might be expected, the World Wide Web Consortium (W3C)<sup>4</sup> focuses on external interfaces:

*'An **open system** is one in which some producers and consumers are loosely connected or are not controlled by the same organization. The internet is a good example of an open system.'*

Three questions arise from the above.

- 1) What are the interfaces referred to in the definitions?
- 2) Why might they be considered to be open?
- 3) Who defines them?

Consider each question in turn.

The interfaces are both internal, that is within a system, and external, for communicating with other systems. Figure 1 shows internal interfaces between components, for example those labelled C (n) and C (n-1), and external interfaces to other components. Suppose for instance that C (n) is an application component accessing a database. The interface it uses to access the database manager would be something like SQL (Structured Query Language). C (n-1) would be the database manager, which would then have an interface to a file manager. The file manager would interface to a disk handler for the physical I/O, which would access the storage subsystem. Standardising the interfaces provides scope for applications and other components to be imported, exported and replaced.

Figure 1 also shows external interfaces, for example to storage subsystems. External interfaces could be to 'peer' components in other systems, which would involve a protocol for communicating between the components. Readers familiar with the communications architectures, which emerged in the 1970s, will recognise this kind of structure. Architectures such as IBM's System Network Architecture (SNA) and the Open Systems Interconnect model from the International Organization for Standardization (the 'ISO-OSI' model) were defined on a layered principle, where each layer exposes services accessed by the layer above through an interface. Each layer communicates with its peer layer in another system using protocols.

---

<sup>2</sup> See <http://www.acq.osd.mil/osjtf/whatisos.html>

<sup>3</sup> See <http://www.opengroup.org/architecture/togaf8-doc/arch/chap36.html> The Open Group was formed some years ago by a merger of two organisations active in open systems: X/Open and the Open Software Foundation (OSF)

<sup>4</sup> [Editorial Draft] Extending and Versioning Languages: Terminology. Draft TAG Finding 13 November 2007 (Note: this is the latest version of the draft – as reported by W3C in February 2009. Work is on-going.) See <http://www.w3.org/2001/tag/doc/versioning>

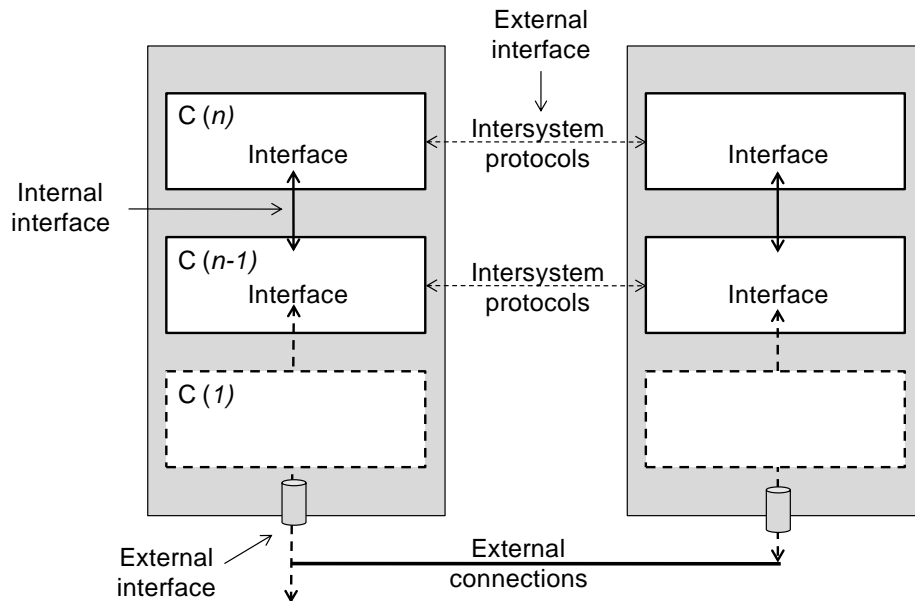


Figure 1: Internal and external interfaces between components

Why would an interface be considered as open? The essential requirement is that the definition or specification is in the public domain in some form and not hidden. Implementations may therefore be available on variety of platforms, both hardware and operating system, and additional implementations may be made in platforms not already included.

Who defines the interfaces that are considered to be open? Perhaps the most genuinely 'open' are standards which are defined by vendor-neutral groups. So standards such as TCP/IP are open, because the Internet Engineering Task Force (IETF), which defines TCP/IP, is vendor-neutral. Web Services standards such as SOAP, defined by groups such as the W3C are similarly open. The International Organization for Standardization (ISO) and International Telecommunications Union (ITU) are further sources of this class of open standards.

A second class of interface is *de facto* standards, which are those that are not defined by a vendor-neutral group but have been widely implemented and accepted. One of the most popular *de facto* middleware standards is WebSphere MQ (formerly known as MQSeries), which is defined by IBM but available on many platforms (including ClearPath OS 2200 and MCP systems), using the IBM portable version. Java standards are *de facto* standards, because although they originate from Sun, they are widely implemented.

Implementations of standards can take two broad forms. First, the specification may be implemented by anyone and released as a product of some kind. In some cases, the implementation may have to undergo a validation or certification process if it is to claim conformance to the specification<sup>5</sup>. Examples include telecommunications software to connect to public networks, and implementations of Java specifications. The second implementation approach is that the specification is implemented by a single vendor, which makes it available as a portable product, either open source or with a licence fee. IBM's WebSphere MQ is an example of this type of implementation, in this case with a licence fee.

One final observation: the use of terms such as 'proprietary' and 'single source' can complicate discussions about openness. Proprietary is frequently interpreted as bad and therefore to be avoided.

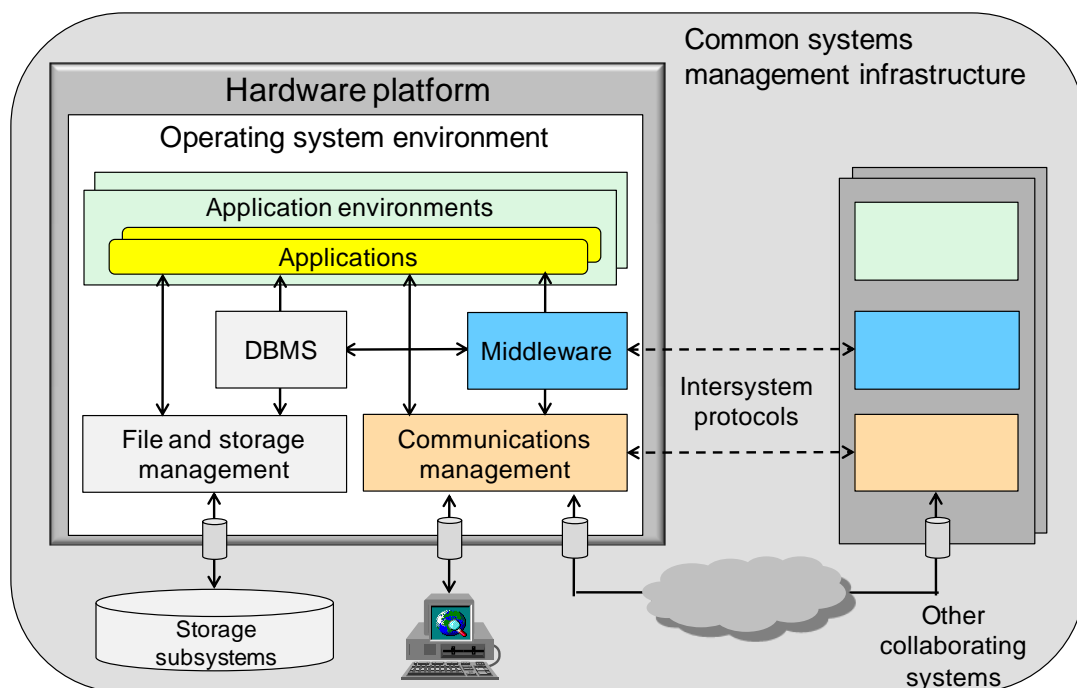
<sup>5</sup> Certification can be costly and time consuming, so some implementations are released without going through the process. The implementation may in fact be complete but cannot claim conformance without undergoing certification.

But all products are proprietary in the sense that they have just one source. What matters is that they have open attributes, such as standard interfaces. Two different products, each proprietary to its supplier, may have identical open standard attributes; Java EE application servers are examples. They can be differentiated on price, performance, and often on features that are not standardised and depend on specific implementations, for instance resilience. But it would make no sense to dismiss them as not open because they are in some sense 'proprietary'.

## Perspectives on openness – an analytical framework

One point that emerges from the above discussion is that systems should *not* be considered as either open or not open; rather, openness encompasses a spectrum of open attributes, in particular interfaces. At one end of the spectrum, there are few open interfaces; at the other end, there are many. Evaluating any particular system or class of systems therefore requires finding out the open attributes it possesses – where it sits on the spectrum in other words. The richer the set of open attributes the system possesses the more open it is.

Evaluating the openness of any particular system requires some form of analytical framework identifying the particular attributes or attribute types to be considered. The approach used in this paper is to view a system from four perspectives, which together encompass the important open attributes. Two of the perspectives consider the system on its own. It will have a network of end users but it is not part of a wider distributed environment. The interfaces are mainly concerned with the ease of importing software from elsewhere, and developing applications to run in the systems. Figure 2 shows the key components and the interfaces they provide.



*Figure 2: The key Components and interfaces used in the analytical framework*

The perspectives are:

- 1) The internal interfaces available, which include application environments, file and database access, and storage subsystems.
- 2) The application development tools and processes that can be used. This perspective goes beyond the definitions of 'open' established above because development tools and processes are not standardised. However, users of systems considered as open tend to expect

particular development approaches and the availability of popular tools, for example graphical interfaces and integrated development environments (IDEs) such as Eclipse.

The other two perspectives view the system as part of a distributed environment, where it collaborates with other systems, for example by participating in a service-oriented architecture implementation. The system is considered as a 'black box' exposing a number of external interfaces and protocols; how it works inside is not important. The perspectives are:

- 3) The external interfaces supported, including communications, distributed application servers, middleware and the facilities provided for distributed transaction processing.
- 4) Systems management: can the system be managed using the technologies and products able to manage other common systems, or at least collaborate with them?

## Evaluating ClearPath system open attributes

Unisys has supported the development of freely available, open standards over many years by participating in their development on standards committees. For example, it contributed significantly to the specification of XML for instance. It has backed up this commitment by implementing the resulting standards in the ClearPath systems and their predecessors, the A- and 2200-Series.

The following sections evaluate the openness of current ClearPath systems, using the analytical framework developed above. Both ClearPath system families generally implement the same technologies and standards although the implementation approach may vary. Any significant differences between OS 2200 and MCP systems are noted.

### Perspective 1: internal interfaces

Open internal interfaces allow software and applications written in other systems, but conforming to the same standards, to be imported. Figure 3 shows the key components and interfaces for ClearPath systems, which are discussed in the following paragraphs.

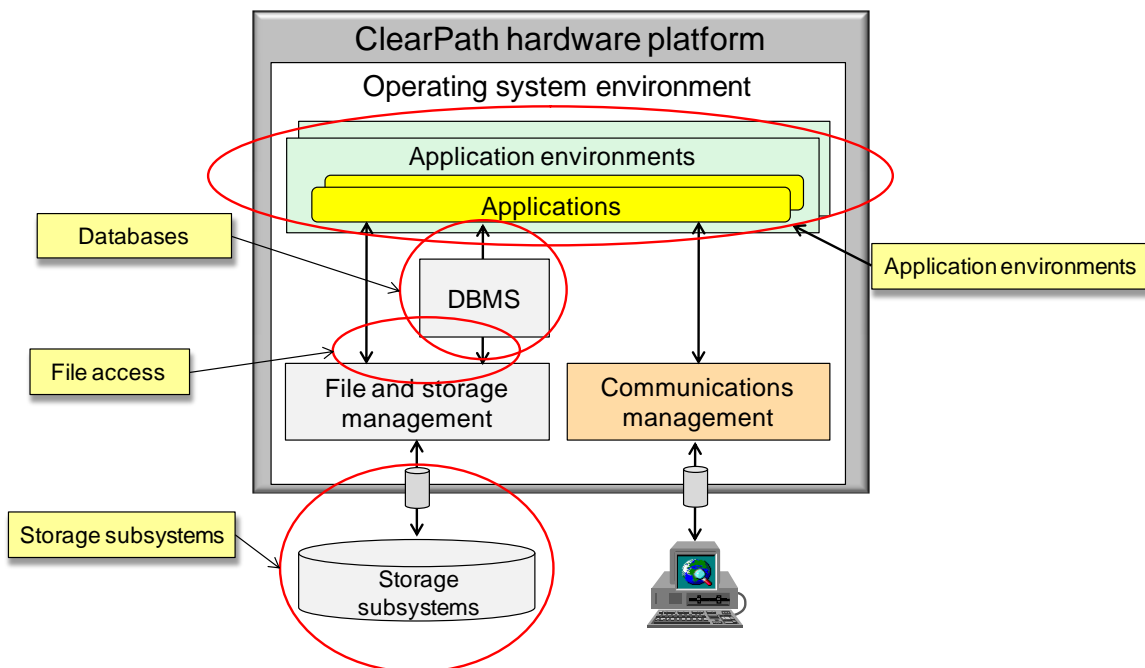


Figure 3: Internal components and interfaces

## Application environments

Both ClearPath families support a number of native application environments, conforming to Unisys' own definitions. They also provide two industry-standard environments, which are implemented by other organisations on a wide variety of platforms. Consider each class of environment in turn.

Native application environments support both batch and transactional applications. The transaction processing environments (COMS with MCP and TIP/HVTIP with OS 2200) are integrated with the operating system and other software, providing an extensive set of features, especially for recovery and security. Applications may be written in a variety of industry-standard languages such as COBOL and C. Support of standard file and database APIs, as explained later, facilitates the import of applications and software written elsewhere. An example is the original implementation of IBM's WebSphere MQ. The portable version released by IBM is written in C.

Both system types provide a native Web server, supporting the development of Web-based applications. A variety of features is available for Web-enabling existing applications originally developed for use with terminals such as the T27 (MCP) and UTS (OS 2200).

The industry-standard application environments are Open Group DTP<sup>6</sup> and Java, including Java EE. The Open Group DTP (Distributed Transaction Processing) model defines an environment for transaction processing, which, as its name suggests, includes transactions distributed across multiple systems implementing the same model. The implementations on ClearPath systems, called Open DTP, are extensions to the native integrated transaction processing environments (COMS and TIP/HVTIP). They include support of two-phase commit to ensure consistency of databases and queues within a single system, and across multiple systems. The implementation is widely used by ClearPath customers, especially in the OS 2200 base.

Both systems support Java implementations by providing a Java Virtual Machine (JVM), and Java EE application servers. The application servers qualified for ClearPath systems are the Open Source JBoss Application Server and the Apache Tomcat Web Server. JCA-compliant resource adapters are provided to allow access to the native application environments: COMS (MCP), TIP/HVTIP and BIS (OS 2200), and Open DTP (both systems) as well as databases. In the case of applications, the connections are two-way, that is, either end may initiate a request. This allows applications running under TIP or COMS, for example, to invoke Java applications as well as Java invoking the TIP or COMS applications.

The original Java releases included a JVM running under the MCP or OS 2200 operating systems. Subsequent developments provide the Java implementations by using embedded Specialty Engines, the OS 2200 and MCP JProcessors, in the ClearPath enclosure. They provide a secure and high-performance environment for the execution of Java applications or other Java software. Examples include Hibernate for persistence management, and Wily Introscope for performance management and problem resolution. As will be seen later, Specialty Engines are used to implement other functions as well.

The Java environments may also run partly or completely outside the ClearPath platform, using the resource adapters to connect to applications and data storage within the ClearPath system. If running off-platform, any Java-compliant application server may be used, as the resource adapters conform to the appropriate Java standards. Figure 4 shows the possible Java configurations.

---

<sup>6</sup> The model was originally defined by the X/Open organisation, which subsequently merged with the Open Software Foundation (OSF) to form The Open Group. The model was derived from Tuxedo and is very similar.

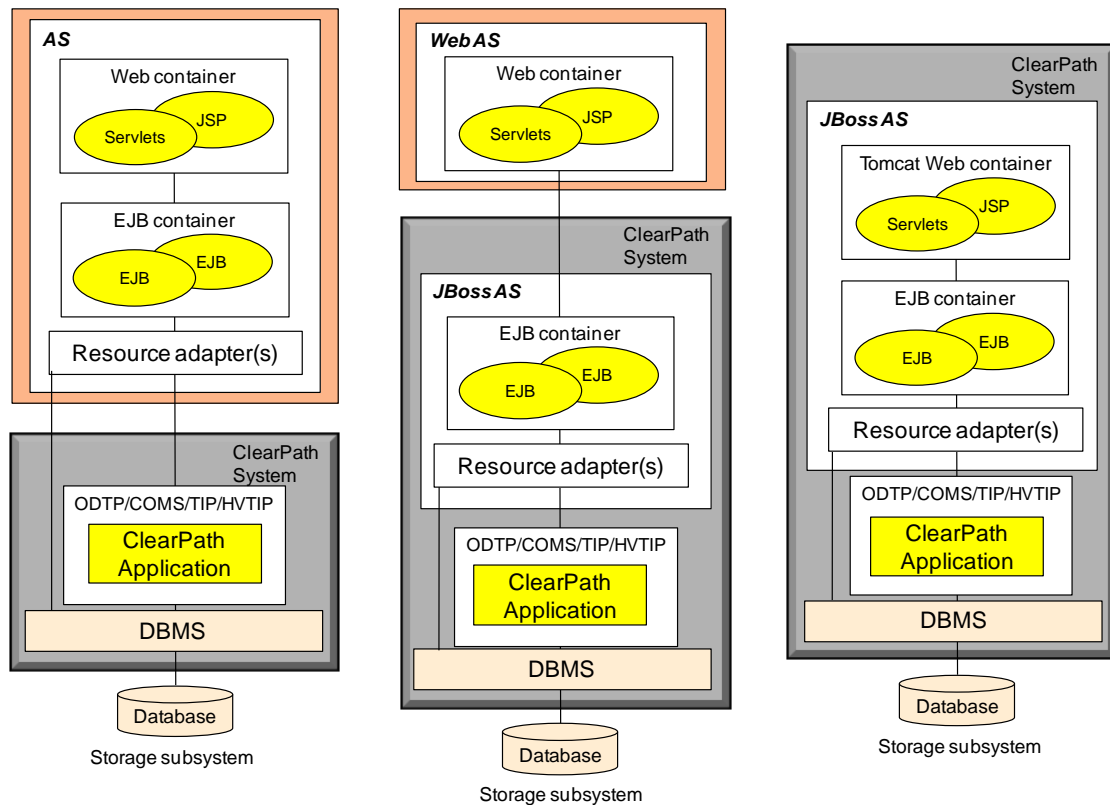


Figure 4: Possible Java configurations with ClearPath systems

Microsoft's .NET is a popular application environment but, unlike Java, can only run under Windows and not other operating systems. As will be discussed in more detail later, there are various options for integrating .NET applications with applications running in other environments within the ClearPath system.

### Files, databases and storage

ClearPath systems implement the Common Internet File System (CIFS), which provides access to files stored on the ClearPath system from clients such as Windows Explorer as well as programs running on the ClearPath system. CIFS was originally developed by Microsoft and is based on the Server Message Block (SMB) protocol for sharing resources such as files and printers. CIFS uses a POSIX-like hierarchical file structure.

The implementation makes it possible for developers and others to use PC tools such as Word, saving the results on a ClearPath system. They would typically do this to take advantage of security and existing file back-up procedures characteristic of systems in a data centre. The implementation also simplifies importing applications and other software written in languages such as C and COBOL, and using the CIFS file structures. Such applications and software require minimal modification to run in ClearPath environments. CIFS is also used in the Java implementation.

ClearPath system databases provide widely accepted interfaces from programs running on the platform. RDMS on OS 2200 provides a native SQL access. RDMS and DMS on OS 2200, and DMS II on MCP, all allow Java programs to access the databases through resource adapters, as shown in table 1. The open source product Hibernate may be used for managing persistence in Java environments.

Data contained in ClearPath-resident databases and other data sources can also be accessed directly from remote systems, either as part of a transaction or for analysis purposes, for example to provide management information. The Java resource adapters can run in an external Java

environment, as shown in figure 4, as well as within the ClearPath system. Other options use products from Unisys or third parties running in Windows environments, as shown in table 2.

ClearPath family	Resource Adapter	Comments
MCP	JDBC for ClearPath MCP driver	DMS II data
OS 2200	RDMS JDBC Driver (RA)	
	DMS RA	
	BIS RA	BIS data and BIS scripts

*Table 1: Java resource adapters for direct access to ClearPath data*

ClearPath family	Product	Function
MCP	ODBC Access (formerly call Data Access)	Allows SQL access to DMS II and other non-relational data stores
	Enterprise Database OLE DB Data Provider for ClearPath MCP	Provides OLE DB access to DMS II data
	XML Provider for ClearPath MCP	Enables DMS II and other data sources to interoperate with other XML-enabled applications and data sources
	DataBridge	Data extraction and transfer from DMS II
OS 2200	UniAccess for RDMS	Provides ODBC access to RDMS
	Data Access	Provides SQL access to DMS and other non-relational databases

*Table 2: Remote direct access to ClearPath data*

Data can be encrypted for storage. Industry standard FIPS certified (governed by FIPS PUB 140-2) data encryption algorithms are supported using the Cipher API on OS 2200 and Cryptolib on MCP. Encryption algorithms supported include Data Encryption Standard (DES), Triple DES (3DES) and Advanced Encryption Standard (AES). Key lengths can vary from 64 to 256 bits, depending on the standard.

Finally, ClearPath systems use industry-standard storage subsystems, for example those from EMC<sup>2</sup>, and access technologies such as Storage Area Networks (SANs).

## Perspective 2: application development

Application development technology is not standardised, and therefore it cannot strictly be considered as a characteristic of open systems. However, many developers and their managements come to expect certain styles or approaches to be available with systems they regard as open. In particular, graphical development tools and IDEs are regarded as necessary, although some developers may need persuading to move from more 'traditional' approaches based on preferred text editors.

One of the most popular IDEs available for different platforms is the open source product Eclipse. It was developed for Java applications in the first place but has been widely extended using plug-ins. Unisys supports Eclipse for both ClearPath families for Java, and also for C, COBOL, ALGOL and other languages. Unisys-supplied plug-ins provide the necessary integration with the ClearPath systems on which the applications will be deployed.

Unisys has long provided tools for rapid application development. These tools work at a high level of abstraction from the ultimate deployment system, an approach which allows developers to focus on

the business problem, not the underlying technology. The resulting applications can be executed in different run-time environments, including systems other than ClearPath.

The current products are Business Information Server (BIS) on OS 2200, Enterprise Application Environment (EAE) on both families, and Agile Business Suite (AB Suite) on MCP. EAE and AB Suite can generate for other platforms, typically Windows. On ClearPath systems, they can generate for Open DTP as well as COMS (EAE and AB Suite) and HVTIP (EAE). AB Suite provides a model-driven development environment based Visual Studio .NET. BIS is a scripting language, with BIS 'engines' for runtime on OS 2200, Windows and Linux. BIS supports JavaScript as well as its own scripting language.

### Perspective 3: external interfaces

Although some references in the previous sections were made to interfaces with other systems, the primary focus was on ClearPath systems in relative isolation, with just a network of end users and external storage subsystems. However, IT environments almost always comprise a number of collaborating systems, both within a single organisation and externally, including clients, suppliers and government. ClearPath systems support the required interfaces to allow them to participate fully in a distributed environment, increasingly based on a service-oriented architecture (SOA) approach.

Perspective 3 considers the external interfaces in four areas, as shown in figure 5:

- 1) Communications.
- 2) Distributed application servers.
- 3) Middleware: environment integrators and loosely-couple middleware.
- 4) Distributed transactions.

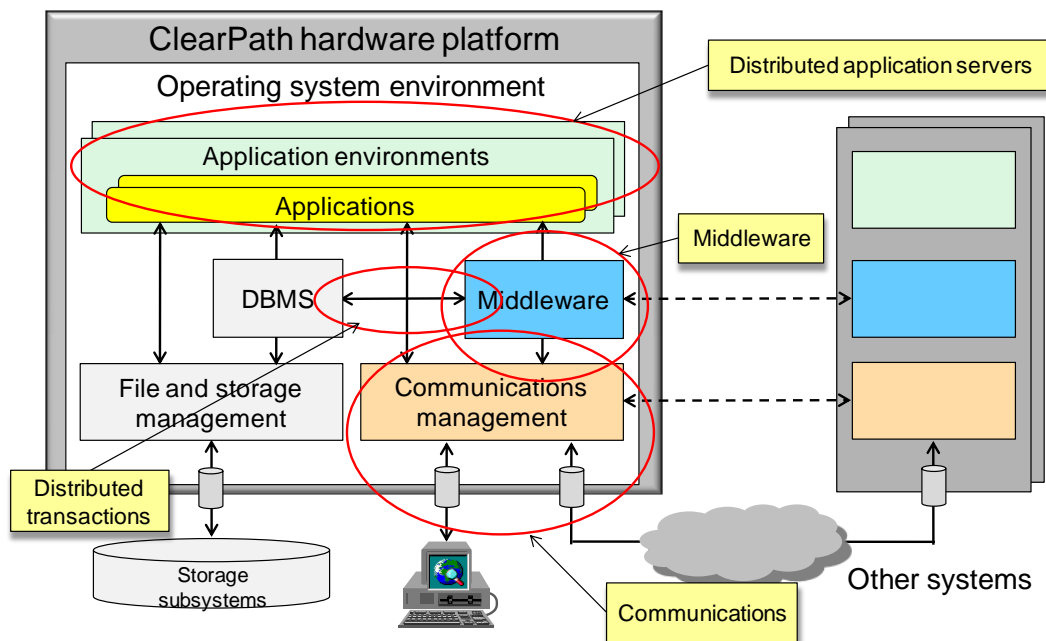


Figure 5: External interfaces

### Communications

Communications standards from the Internet Protocol Suite (aka TCP/IP), as defined by the IETF, are now ubiquitous in the industry. In common with other vendors, Unisys has progressively used TCP/IP to phase out a long history of communications implementations, including:

- ❑ Its own communications standards, ranging from terminal and workstation protocols, to layered communications architectures, Burroughs Network Architecture (BNA) and Distributed Communications Architecture (DCA, from Sperry).
- ❑ Implementations of other vendor standards such as BSC3270 and SNA from IBM.
- ❑ Industry-specific standards, mainly for airlines. Some of these standards remain for Unisys airline and travel industry clients, implemented in a third party product.
- ❑ Extensive implementations of the ISO-OSI model, although some parts of the model have remained.

ClearPath systems contain implementations of the major relevant RFCs<sup>7</sup>, together with the necessary electrical and physical network interfaces defined by organisations such as the IEEE (formerly known as the Institute of Electrical and Electronic Engineering) and the International Telecommunications Union (ITU, formerly the CCITT). Common supported standards include:

- ❑ Internet Protocols versions 4 and 6 (IPv4 and IPv6).
- ❑ Telnet and File Transfer Protocol (FTP)
- ❑ Industry-standard encryption algorithms for data in motion (SSL/TLS).

ClearPath systems also support standards for the Internet developed by the W3C and other bodies. The Internet standards use TCP/IP for lower level communications.

### **Distributed application servers**

As their name suggests, distributed application servers support distributed applications spread across more than one system, where each system uses the same application server technology. The Java EE and Open DTP application servers available in ClearPath systems support distributed environments, or ecosystems as they are sometimes called. Thus, ClearPath implementations may participate in either Java or Open DTP ecosystems, where the collaborating systems implement the same application server technology. The platforms may vary as both Java and Open Group DTP are implemented under a number of operating systems. (.NET also supports distributed ecosystems but only under Windows.).

### **Middleware**

Few organisations of any size can use only one set of standards; in many cases, different technologies have to coexist. This can be done by environment integrators, which provide gateways or bridges between different standards. (It can also be done using loosely-coupled middleware, as explained below.) Environment integrators bridge between two standards by converting each standard to the syntax and semantics required by the other. For example, suppose a ClearPath system is running Open DTP and needs to be integrated with a .NET environment. The integrator would convert between the two standards: the Open DTP side thinks it is in an Open DTP ecosystem, while the .NET side sees it as a .NET ecosystem.

Unisys provides a number of environment integration products, which are summarised in table 3. The product name and the two types of environment are shown. The Tuxedo integration product has been used to connect other Open Group DTP implementations, specifically the Fujitsu (ICL) product, to Tuxedo.

---

<sup>7</sup> RFCs (RFC = Request For Comment), are the documents used by the IETF to define standards and to provide a vehicle for commentary and discussion. RFCs defining standards are referred to as *normative*, to distinguish them from those just providing commentary and discussion.

Integration product	Other environments	ClearPath environments
Java Resource Adapters	Java, Java EE	Open DTP, COMS, TIP/HVTIP, BIS
Distributed Transaction Integration (DTI)	Various, including .NET, Java and some packages	Open DTP, TIP/HVTIP, OS 2200 batch
BEA eLink OSI-TP	Tuxedo	Open DTP
Component Enabler	.NET	EAE, AB Suite
COMTI	.NET	COMS
ClearPath e-Portal	Web browser, Smartphones (e.g.iPhone), Web Services)	COMS, Open DTP, TIP/HVTIP

*Table 3: Summary of environment integrators*

In distributed systems using loosely-coupled middleware, the component applications communicate by exchanging messages. No assumptions are made about how an application is written or the environment in which it is running. All that is required is support of the appropriate middleware, and agreement on the form and content of the message. Distributed environments comprising applications running in different technologies – TIP/HVTIP, COMS, Java EE, Open Group DTP, .NET, or CICS, for example – may therefore be constructed using loosely-coupled middleware<sup>8</sup>.

Application components in ClearPath systems may collaborate with other applications using the following loosely-coupled technologies. Both ClearPath system families support all the technologies although the implementation approaches may vary.

- Message queuing using IBM's WebSphere MQ, formerly called WebSphere MQ.
- Message queuing using Microsoft Message Queuing (MSMQ).
- Web Services, as defined by the W3C and others. This technology is specifically targeted at SOA. Its use of XML (eXtensible Markup Language) for data and protocol (SOAP) makes it ideally suited to heterogeneous environments.
- File transfer using FTP. Although FTP may not always be considered to be middleware, it is well suited to shipping large quantities of data. It has the advantage of being ubiquitously available, and plays a significant role in electronic commerce.

The implementation of WebSphere MQ in OS 2200 systems is different from the approach used for MCP. The original implementation was to import IBM's portable version to run entirely under OS 2200. More recently, a Specialty Engine, the OS 2200 QProcessor, has been introduced for OS 2200 systems. The interfaces have remained the same so the applications see no differences. The MCP version implements the WebSphere MQ queue manager under Windows in a Windows partition, with the interfaces provided under MCP for MCP-resident applications. For MSMQ, the queue manager runs in Windows for both systems, with interfaces provided from the OS 2200 or MCP environments.

Web Services with ClearPath systems are provided in various ways. The complete Web services stack – SOAP and so on – can run in the Java environment, using the resource adapters to access applications and data in the MCP or OS 2200 environment. Web Services interfaces can also reside in a Windows environment, using environment integrators or loosely-couple middleware to interface to the applications. And Web Services, among other standards, are provided by the ClearPath e-Portal

---

<sup>8</sup> Although loosely coupled middleware is well suited to interconnecting applications running in different technologies, it can of course be used in homogeneous environments. For example, two Java EE systems could communicate using Web Services or message queuing. There are good reasons for using message queuing, for instance to guarantee delivery even when the receiving end of a connection is not available.

Specialty Engine. The choice of product depends on the client's requirements and current environment.

Finally, data can be encrypted before transmission to other systems using the industry-standard encryption algorithms accessed by the Cipher and Cryptolib APIs introduced earlier in Perspective 1, Files, databases and storage.

## Distributed transactions

Any transaction changing data must be able to guarantee data integrity: the data must remain in a consistent state at the end of the transaction. All the updates have to be applied or none is; if the transaction is not completed, the databases and other data stores must be left in the same state they were at the start of the transaction<sup>9</sup>. If multiple data stores are involved, either within a single system or spread across systems, a process called two-phase commit (2PC) is used to manage the consistency and maintain data integrity. Databases supporting 2PC conform to an industry standard called XA; they are referred to as XA-compliant. ClearPath databases and WebSphere MQ message queues in ClearPath systems are XA-compliant, as are most other commercial databases such as Oracle and Microsoft SQL Server.

Open DTP applications in ClearPath systems may collaborate with other Open DTP implementations in a distributed transaction, maintaining data integrity using 2PC. Java EE applications in ClearPath systems may take part in distributed Java transactions while maintaining data integrity. In some cases, distributed transactions across different technologies, connected by environment integrators, may also maintain data integrity. For example, Open DTP with DMS II, RDMS or DMS, and Tuxedo with Oracle, can participate in a 2PC transaction because the integrator, eLink OSI-TP, passes the relevant information required for 2PC. Similarly, Open DTP and .NET environments may also participate in a distributed transaction.

Unisys provisions for external interfaces in ClearPath systems, as explained above, are comprehensive. They embrace all the major standards required for participation in distributed environments, and are one of the significant strengths of the two ClearPath families. The 'More information' section at the end of this paper points to sources for detailed explanations of the technology and products provided, illustrated by a variety of case studies.

## Perspective 4: systems management

Participation in distributed environments such as SOA implementations requires comprehensive systems management and a high level of automation, including cross-system automation. It is therefore important that management tools can span the systems involved as well as integrate with other management products.

Unisys supplies two systems management products able to work across multiple systems. They are Operations Sentinel (formerly known as Single Point Operations (SPO)) and Enterprise Output Manager (EOM). Operations Sentinel provides automation for multiple systems types, including both ClearPath families. EOM manages output, again from multiple system types, ensuring its delivery in a wide variety of options.

Other organisations supply management products that can be used with ClearPath and many other systems. They include performance management tools from TeamQuest and Sightline, and workflow management with SMA's OpCon/xps product.

---

<sup>9</sup> This is sometimes explained by referring to a transaction as having 'ACID' properties: **A**tomicity: A transaction should be completed and committed, or rolled back if there are problems during its execution: it is an atomic entity in that it is indivisible. **C**onsistent: Changes made to databases must be from one valid state to another, not something indeterminate. **I**solated: The results of a transaction should be invisible to other transactions until the transaction is completed. **D**urable: Changes to databases, which have been made during a transaction, should be permanent and survive future media and other failures. Maintaining the ACID properties is what is meant by transactional integrity.

All the above can integrate with other management products likely to be found in the data centre, including CA NMS (formerly CA Unicenter), IBM Tivoli and HP OpenView. And both ClearPath families provide Simple Network Management Protocol (SNMP) agents, generating events which can be picked up by most management products.

## ClearPath system mission-critical attributes

The definitions of open systems used in this paper are independent of the hardware or operating systems. However, that does not mean that the underlying platforms are unimportant. In the case of ClearPath systems, the platforms provide attributes that make them ideally suited for their primary role of mission-critical transaction processing, for operational and economic<sup>10</sup> reasons. These attributes are not found in the platforms usually associated with openness.

They include:

- ❑ *System efficiency.* The systems can handle multiple, critical applications of different types under a single instance of the operating system; the ClearPath operating systems MCP and OS 2200 have always provided a virtual environment for applications. Processor loads of close to 100% can be sustained, together with high levels of security. Thus, the amount and complexity of equipment in configurations, and hence energy and space consumption, are low compared with alternatives.
- ❑ *Software integration.* The tightly integrated software stack supplied by Unisys makes for great efficiency and reduces systems programmer resource requirements, reducing time, cost and risk.
- ❑ *Metering.* Charging based on Capacity on Demand and Pay-for-Use business models, based on metering technology, can reduce costs – around 30% in a number of cases – while increasing performance significantly. Unplanned changes in demand leading to revenue opportunities can be accommodated immediately. And batch windows are reduced, sometimes to the extent of eliminating an operations shift. Metering is unique to ClearPath systems.
- ❑ *Reliability.* System reliability, with fast recovery in the rare event of a failure, maximises availability. The time between system failures is measured in years in some cases.
- ❑ *Security.* The high levels of security reduce the risk of any data vulnerability, with consequent adverse economic effects, to near zero.

The final point on security is an important one for all organisations and critical for many. If security is compromised, especially if valuable information is extracted from a database or damaged for fraudulent or other reasons, the results can be catastrophic for the organisation concerned. It may be destroyed or subjected to very heavy costs in the form of penalties and compensation. In other cases, life may be lost as a result of security breaches.

ClearPath systems, and indeed mainframe-class systems in general, protect valuable data from compromise. Evidence for this can be found in a National Institute of Standards and Technology (NIST) database<sup>11</sup>, which records security vulnerabilities. As of mid-2009, the number of vulnerabilities reported in the database were 1560 (Windows), 1263 (Linux), 181 (UNIX), with just one for a ClearPath system (MCP), recorded in December 2002. It was a denial of service attack in which no data were compromised. The problem was quickly corrected.

---

<sup>10</sup> A White Paper on the economics of ClearPath systems '*Delivering Value: The Economics of ClearPath Systems*' covers the subject in detail. Pointers to how to find this paper and others are provided in the 'More information' section of this paper.

<sup>11</sup> The database can be found at <http://nvd.nist.gov>

## Comparisons with other platforms: a summary

This paper has discussed the open attributes of ClearPath systems. How do they compare with other platforms? Table 4, below, summarises some of the key open attributes of ClearPath systems, as well as UNIX, Microsoft Windows, Linux and IBM z/OS. The attributes are grouped according to the four perspectives used in the paper. Notes accompanying the table provide additional information and clarification; they are indicated in the table by numbers in parentheses.

Perspective	Attribute class	Standards	OS 2200	MCP	UNIX	Windows	Linux	z/OS	
Internal interfaces	Industry-standard app. environments	Java/ Java EE	Yes (1)	Yes (1)	Yes	Yes	Yes	Yes	
		Open Group DTP (ODTP)	Yes	Yes	Yes (2)	Yes (2)	Yes (2)	No	
	File/DB/ Storage	CIFS	Yes	Yes	Yes	Yes	Yes	Yes	
		JDBC	Yes (3)	Yes (3)	Yes (4)	Yes	Yes (4)	Yes	
		ODBC	Yes (3)	Yes (3)	Yes (4)	Yes	Yes (4)	Yes	
		SQL	Yes (3)	Yes (3)	Yes (4)	Yes	Yes (4)	Yes	
Application development	IDEs	Eclipse	Yes (5)	Yes (5)	Yes	Yes (6)	Yes	Yes	
External interfaces	Communications	TCP/IP, IEEE..	Yes (7)	Yes (7)	Yes	Yes	Yes	Yes	
	Distributed apps. Servers (8)	Java EE/ODTP	Yes	Yes	Yes	Yes	Yes	Yes (9)	
	Middleware (10)	FTP	Yes	Yes	Yes	Yes	Yes	Yes	
		MSMQ	Yes	Yes	Yes	Yes	Yes	No	Yes
		WebSphere MQ	Yes	Yes	Yes	Yes	Yes	Yes	Yes
		Web Services	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	Distributed transactions	XA-Compliant databases	Yes (11)	Yes (11)	Yes (12)	Yes	Yes (12)	Yes	
Systems management	Standards & tools	SNMP	Yes	Yes	Yes	Yes	Yes	Yes	
		Common tool I/f	Yes	Yes	Yes	Yes	Yes	Yes	

*Table 4: Some key open attributes of ClearPath and other systems*

#### *Notes to table 4*

- 1) The native transaction processing environment may be integrated with other environments in the system. For example, Java/Java EE integrate with ClearPath COMS and TIP/HVTIP, and also databases using resource adapters.
- 2) Tuxedo, from which Open Group DTP was derived, is available on all these platforms.
- 3) ClearPath database managers support JDBC, ODBC and SQL access.
- 4) UNIX and Linux do not have native databases but any database used would support these functions.
- 5) Plug-ins add the necessary features needed for ClearPath with Eclipse. Languages supported include C, COBOL and ALGOL as well as Java.
- 6) Developers using .NET would normally use Visual Studio .NET for development.
- 7) Encryption using industry standards such as SSL/TLS is supported.
- 8) That is, distributed applications can be created across multiple systems using the same application server – Java EE or Open Group DTP/Tuxedo – on the same or different platforms.
- 9) z/OS can participate in Java EE distributed applications but not Open Group DTP.
- 10) The middleware technologies shown in the table are all loosely coupled. ClearPath systems also provide an extensive range of environment integrators for connecting COMS, TIP/HVTIP, Open DTP, BIS, EAE and AB Suite to Java EE, .NET, Tuxedo and packages such as SAP.
- 11) The environment integrators in ClearPath middleware enable distributed transactions using 2PC between Open DTP applications on ClearPath, and Tuxedo, Java EE and .NET on other platforms.
- 12) Commonly chosen databases such as Oracle are XA-compliant.

## More information

Additional information about standards, technologies and products is widely available, at both strategic and detailed levels. The following are just a few of the useful sources. URLs are included, but mergers, acquisitions, and reorganisations can change the links. Search engines such as Google are now so powerful that it is often simpler to enter a query and let the search engine find the exact location required within an organisation's website.

There are many Independent, vendor-neutral organisations, working on standards and other collaborative activities, and usually under the domain .ORG. Here is a selected list of organisations, a number of which are mentioned in the text of this paper:

- The Apache Software Foundation: <http://www.apache.org>
- The Carnegie Mellon Software Engineering Institute (SEI): <http://www.sei.cmu.edu/>
- IEEE: <http://www.ieee.org>
- The International Organization for Standardization (ISO): <http://www.iso.org>
- The International Telecommunications Union (ITU – formerly CCITT): <http://www.itu.int>
- The Internet Engineering Task Force (IETF): <http://www.ietf.org>
- The National Institute of Standards and Technology (NIST): <http://nvd.nist.gov>
- The Open Group: <http://www.opengroup.org>
- UDDI.org (for information about Web Services): <http://www.uddi.org>
- World Wide Web Consortium (W3C): <http://www.w3.org>

Information about ClearPath middleware and other Unisys products and services can be found at: <http://www.unisys.com/index.htm> . In particular, there are many White Papers explaining different aspects of ClearPath systems, such as middleware, security and economics. The White Papers can be found by going the URL shown above and selecting: Research & Innovation>>Publications>>White Papers.

Additional information can be found in the Unisys eCommunity at: <http://ecomunity.unisys.com>

## Appendix: glossary of technical terms and products

### Technical terms and acronyms

**AES:** Advanced Encryption Standard – a data encryption algorithm.

**API:** Application Programming Interface

**AS:** Application Server

**BNA:** Burroughs Network Architecture.

**BSC3270:** A half-duplex, multi-point bisynchronous protocol from IBM for the 3270 terminal system.

**CIFS:** Common Internet File System.

**Container:** This term is used in this paper to indicate an application environment that can be a web container, in which JSPs or Servlets execute, and an EJB container, in which EJBs execute.

**DBMS:** Database management system.

**DCA:** Distributed Communications Architecture. This is the Unisys layered communications architecture, which is closely related to the ISO Open Systems Interconnect model. The Telcon system implemented DCA (and other protocols) and ran in the Distributed Communications Processor (DCP) family. Like other vendor architectures, it is superseded by TCP/IP and various middleware standards.

**DES, 3DES:** Data Encryption Standard and Triple DES – data encryption algorithms.

**DTP:** Distributed Transaction Processing. Open Group DTP model is a reference model for distributed transaction processing, defined by X/Open originally.

**EJB:** Enterprise Java Bean. A software entity defined by Java EE. It executes in an EJB container.

**FIPS:** Federal Information Processing Standards. The standards are published as FIPS Publications, abbreviated to FIPS PUBS.

**FTP:** File Transfer Protocol. A standard defined by the Internet Engineering Task Force (IETF).

**IDE:** Integrated development environment.

**IEEE:** Now just called IEEE. Formerly **the** Institute of Electrical and Electronic Engineering.

**IETF:** Internet Engineering Task Force.

**ISO-OSI:** Open Systems Interconnect model from the International Organization for Standardization.

**IP:** Internet Protocol. Used for routing data packets. IPv4 (version 4) has been superseded by version 6 (IPv6), which adds a number of features, but mainly increasing the address size; IPv4 was running out of possible addresses.

**ITU:** International Telecommunications Union. Formerly called the CCITT.

**Java/ Java EE:** Java EE is Java Platform, Enterprise Edition. It was called J2EE (Java 2 Platform, Enterprise Edition) up to release J2EE 1.4; with release 1.5, it was renamed Java EE 5 rather than J2EE 1.5. This is a set of standards defining a Java environment suitable for large-scale applications. Implementations are called Java EE application servers.

**Java SE:** Java Platform, Standard Edition, called Java 2 Platform, Standard Edition (J2SE) up to release 1.5. With release 1.6, it was named Java SE 6 rather than J2SE 1.6.

**JDBC:** Java DataBase Connectivity. This is the standard defined in Java SE for accessing databases.

**JSP:** Java Server Page. A software entity defined in Java EE. It executes in a web container.

**JVM:** Java Virtual Machine. This is a software environment in which Java programs execute. It can be implemented on any computer. Implementations are available on ClearPath systems.

**Linux:** This is a UNIX-like Open Source operating system originally created by Linus Torvalds. It is available from a number of suppliers.

**Middleware:** computer software which connects software components or applications, including both control connections and data exchange

**NIST:** National Institute of Standards and Technology.

**ODBC:** Open DataBase Connectivity. This is an API for accessing database management systems, independent of programming language, database system, and operating system.

**OSI-TP:** A protocol defined by the International Organization for Standardisation (ISO) for distributed transactions. Includes facilities for ensuring consistency of multiple database updates.

**OS-JTF:** US Department of Defense Open Systems Joint Task Force.

**POSIX:** Portable Operating System Interface (for UNIX).

**SAN:** Storage Area Network.

**SEI:** (Carnegie Mellon) Software Engineering Institute.

**Servlet:** A software entity defined in Java EE. It executes in a web container.

**SMB:** Server Message Block protocol for sharing resources such as files and printers.

**SNA:** IBM's System Network Architecture.

**SNMP:** Simple Network Management Protocol.

**SOA:** Service-oriented architecture.

**SOAP:** A protocol defined by the W3C for interconnecting nodes in a distributed service environment. It was originally an acronym for Simple Object Access Protocol, but is now just a name.

**SQL:** Structured Query Language, a language for accessing databases.

**SSL/TLS:** Secure Sockets Layer/Transport Layer Security. Standards for providing secure communications between systems using encryption.

**TCP/IP:** Transmission Control Protocol/Internet Protocol. TCP and IP are important protocols defined by the IETF force for constructing networks. The term 'TCP/IP' is also used to indicate the entire set of standards for networking defined by the IETF.

**UNIX:** This operating system was originally developed by AT&T at Bell Laboratories in the early 1970s. The specification is now owned by The Open Group. There are many variations of UNIX.

**Web Service(s):** A set of technologies and standards defined by the W3C and others for implementing SOA. Web Services are specific case of SOA, not the only case.

**W3C:** World Wide Web Consortium.

**XA:** This is a system-level interface between a resource manager (RM) and the transaction manager (TM) in The Open Group DTP model. XA provides the 2PC capability. A database or other resource manager that supports it is called XA-compliant.

**XML:** eXtensible Markup Language is a language defined by the W3C for encoding messages and protocols in a machine-independent way.

**2-PC:** 2-Phase Commit. A technique for ensuring that different databases, possibly in different systems, are consistent at the end of a transaction, i.e. all are updated or none is.

## Products

**AB Suite:** See EAE.

**BIS:** Business Information Server. It is the latest evolution of a product originally called MAPPER.

**BIS-RA:** Provides Java access to BIS data and BIS scripts (formerly known as RUNS).

**CA NMS:** Network and systems management tool from CA. Formerly CA Unicenter.

**CICS :** Customer Information Control System – an IBM transaction processing environment.

**ClearPath ePortal:** Implemented in a Specialty Engine for ClearPath, it provides a means of connecting applications (originally COMS but now extended) to new clients including web browsers, Web Services and mobile devices such as iPhones. Includes some orchestration facilities.

**Component Enabler:** This product is a Unisys environment integrator product connecting Microsoft COM+/.NET and Java environments into EAE and AB Suite applications.

**COMS:** Transaction processing environment on ClearPath MCP systems, now called Transaction Server.

**COMTI:** ClearPath MCP Interface to Microsoft Transaction Integrator.

**DMS 2200:** The hierarchical database on ClearPath OS 2200 systems; it is now called Enterprise Network Database Server for ClearPath OS 2200.

**DMS-RA:** This product provides Java access to DMS data.

**DTI:** Distributed Transaction Integration is a Unisys environment integrator connecting a variety of external technologies to Open DTP, TIP/HVTIP, and batch.

**EAE:** Enterprise Application Environment is an evolution of a product originally called LINC. A later evolution is called AB Suite (Agile Business Suite). AB suite is not available with OS 2200.

**Eclipse:** Open Source development environment for Java and other languages. It is available with ClearPath systems for C, COBOL, Java and other languages.

**eLink OSI-TP:** An environment integrator developed by Unisys in conjunction with BEA for connecting Tuxedo to implementations of Open Group DTP. Now renamed BEA Tuxedo Mainframe Adapter OSI-TP.

**EOM:** Enterprise Output Manager from Unisys. Delivers output to various media in a choice of formats.

**Hibernate:** Open Source product for managing persistence, mapping objects to relational databases.

**Introscope:** A systems management and debugging tool for Java environments from Wily Technology.

**Java Resource Adapters:** The adapters provide connections between Java and non-Java environments, allowing Java applications to access other applications and databases.

**JBoss AS:** The JBoss Application Server is an Open Source Java EE application server.

**JDBC:** Java Database Connectivity.

**MCP:** The operating system for ClearPath and earlier A-series systems.

**MCP JProcessor:** The Unisys ClearPath MCP Java Specialty Engine.

**MSMQ:** Microsoft Message Queuing is the Microsoft message queuing middleware.

**OLEDB:** Object Linking and Embedding Database from Microsoft.

**OpCon/xps:** Cross-platform process automation and job-scheduling for data centres from SMA.

**Open DTP (OLTP TM2200):** Open Distributed Transaction Processing is the Unisys implementation of The Open Group DTP model in ClearPath systems.

**OpenView:** Network and systems management products from HP.

**Operations Sentinel:** A Unisys systems automation and management tool, formerly known as Single Point Operations (SPO).

**OS 2200:** The operating system for ClearPath and earlier 2200-series systems.

**OS 2200 JProcessor:** The Unisys ClearPath OS 2200 Java Specialty Engine.

**OS 2200 QProcessor:** The Unisys ClearPath OS 2200 QProcessor Specialty Engine implementing WebSphere MQ.

**RDMS 2200:** Relational Data Management System for OS 2200, now called Enterprise Relational Database Server for ClearPath OS 2200, is the relational database management system on ClearPath OS 2200 systems.

**RDMS-JDBC:** Product provides Java access to RDMS data.

**TIP/HVTIP:** Transaction Interface Processor/High Volume TIP is the native transaction processing environments on ClearPath OS 2200 systems.

**Tivoli:** Service management, security and asset management software from IBM.

**Tomcat:** An Open Source Web Application Server from the Apache Software Foundation.

**Tuxedo:** This distributed transaction application server was developed originally by AT&T as a transaction processing monitor for UNIX. Tuxedo is now owned by BEA. The Open Group DTP model is very similar as it was derived from Tuxedo.

**WebSphere MQ:** IBM's message-oriented middleware product, formerly known as MQSeries but now included within the WebSphere family.

**z/OS:** IBM operating system for zSeries.

**.NET:** This is a Microsoft umbrella name covering a number of related technologies, which are concerned with building (transaction) systems from distributed components under Windows. It originated with OLE (Object Linking and Embedding), and COM (Component Object Model), and passed through various stages, including DCOM and COM+, to become .NET.

## About the author

### Peter Bye

Now an independent consultant, Peter Bye was a senior system architect in Unisys, based in London. His special area of interest is networked computing, including communications networking, middleware, and architectures. He has many years of experience in information technology, working as a programmer, analyst, team leader, project manager and consultant in large-scale customer projects in banking, transportation, telecommunications and government. He has also worked in software development centres.

He has worked for extended periods in Sweden, Denmark, Finland, Norway, the USA, France and Spain, as well as the UK. He has presented at a wide variety of conferences and other events and is the author of a number of papers on networking, systems management, and middleware. He is the co-author of a book on middleware and system integration – IT Architectures and Middleware: Strategies for Building Large, Integrated Systems – which was published in May 2004.

### Bye Associates

Bye Associates is a group of highly experienced business and IT professionals who come together as a team to offer expertise and support to organisations in the private, public and third sectors. Their IT capabilities cover business analysis and requirements, systems architecture and technical specifications, project management and training. Their business experience has been formed in a wide range of sectors, from transport to manufacturing, from financial to public services. They have worked in a number of countries and are sensitive to cultural differences and the challenges of change management.