



white paper

ClearPath OS 2200 Application Modernization

By: Robert Vavra, ClearPath OS 2200 Systems, Unisys Systems & Technology

strategy

A Real-Time Infrastructure (RTI) is a long-term vision of what business and IT can achieve in the future. An innovative blend of technology, governance and operational processes are to be combined in an IT strategy aimed at creating a proactive infrastructure over time. One that is self-managing, autonomic, invisible, available, low-cost and totally responsive to the business needs of the enterprise. All in real time.

Take a look at anyone's RTI definition and you'll see that it doesn't throw away what you already have today and make you start all over again. In fact, evolving to an RTI always starts by looking at where you are today and deciding where you want to go in the future.

Your ClearPath systems can take you to places you never dreamed possible. Leveraging the investments you've made in your COBOL application portfolio means first being able to see clearly how these applications map to your business strategy. Then adapting your applications to deliver the agility you need to accommodate change at the speed of marketplace dynamics.

To achieve this agility, the Unisys 3D Blueprinting™ approach can help you map your ClearPath OS 2200 applications with your business strategy – making visible connections you may never have otherwise discovered. Then you can transform these applications to become part of a Service-Oriented Architecture (SOA), enabling you to reuse their components to easily expand the applications and develop entirely new ones. And to become critical elements of the Real-Time Infrastructure you're building.

Unisys has launched the ClearPath Application Modernization program, with the objective of helping you meet business challenges by leveraging your existing applications using the latest consulting methods and enabling technologies.

“Consider the value of your [ClearPath] mainframe applications and what it would cost to start over. Leveraging what you have is a huge financial advantage. And it's a lot easier to do than you think.”

Ken Snodgrass, Data Processing Manager, Crescent Electric Supply Company

Table of Contents

Overview	2
Discover: Document, Inventory and Assess	3
Streamline: Simplify and Modularize	3
Transform: Service-enable, Integrate, Wrap and Evolve	4
Transform Technology Solutions	5
Wrapping as a Service	5
Reuse or Build New	5
OS 2200 Technology Solutions	5
Discover Stage	5
Streamline Stage	6
Transform Stage	6
Wrapping Transactions	7
Non-invasive Wrapping of a Screen-oriented Transaction as a Service	7
Modifying a Transaction Program for Wrapping as a Service	7
Creating a Transaction Program to be Called as a Service	8
Wrapping a Transaction Service Program as a Web Service	8
Calling Transactions	9
Calling a Transaction Program from a J2EE Application	9
Calling a Transaction Program from a .NET Application	9
Calling a Transaction Program via MQSeries	9
Calling a Transaction Program via Microsoft Message Queuing	10
Outbound Calls	10
Calling an External Service from an OS 2200 Program	10
Other Transformations	11
Replacing File Transfers by Message Queuing	11
Integrating with an Enterprise Service Bus	11
Replacing Printed Reports with Electronic Data Feeds	11
Summary	12
Biography	12

Overview

The Unisys Application Modernization program addresses a broad spectrum of issues that ClearPath customers are likely to be having with their COBOL applications – ones that have been running the business for 20-30 years or more.

Over time and with many changes applied, even the best-implemented applications can degenerate to the point where they are difficult to change and maintain. Costs trend upward as developers struggle with systems that are complex or lack current documentation. Yet the need to maintain and enhance these applications will persist well into the future as our mainframe customers continue to leverage the huge investments they've made in intellectual assets that took years to both develop and perfect.

Taking advantage of new market opportunities is a hallmark of a successful organization. Doing so depends in large part on the ability to adapt supporting business processes and align them with new strategies. What's more, such alignment means adjusting the underlying enterprise applications that automate these processes in order to suit new requirements.

Today's regulatory requirements, e-business opportunities and mergers are making rapid-fire demands on IT departments. Industry experts, including Unisys, are now recommending the most efficient and risk-averse way for IT to respond: composite applications (applications that are a combination of existing, third-party and new capabilities joined together in well defined interfaces).

To bring about the agility attributable to composite applications, organizations are modeling their applications on Service-Oriented Architecture (SOA). SOA and the related Web Services concept enable the reuse of application components through the ability to call very specific business activities such as "confirm customer address" or "reopen mortgage account" as needed. With software developed on this principle of process building

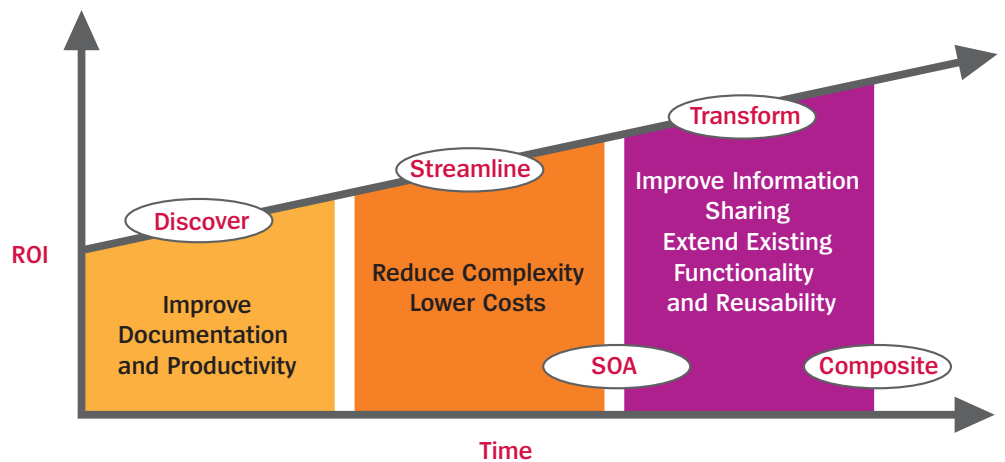
blocks – even when the building blocks were developed using different technologies and dissimilar information models – developers can quickly assemble and reassemble composite applications to suit new and changing business needs.

This paper focuses on ways to expose such building blocks. How is this done? By leveraging the proven business logic and data of existing ClearPath application portfolios. The broader topics of architecting and assembling composite applications using techniques such as Enterprise Service Bus (ESB) will be addressed in future papers.

The Unisys Application Modernization program consists of consulting methodologies and the enabling technologies needed to achieve greater application agility through adherence to SOA best practices. According to Gartner analysts, "By 2008, SOA will provide the basis for 80% of development projects." (May 2005)

SOA is viewed as a risk-averse, cost-effective way to solve immediate business issues such as "expand the sales channel to another country" or "comply with new government reporting" while building applications that will be more flexible in the future. Having more flexible applications translates to lower costs and faster times to market.

Unisys has defined a three-stage process for addressing business demands that affect existing COBOL applications: Discover, Streamline, and Transform.



The Unisys Application Modernization Methodology

Our Application Modernization methodology is designed to deliver a measurable return on investment at each of these modernization stages. Our in-depth knowledge of our ClearPath customers' environments, our enabling technologies and our best-in-class application modernization tools speed up the process considerably. Your staff can apply our modernization tools and technologies to your application portfolio. Or you can engage our consultants to perform parts of the work.

Unisys employs Relativity Technologies' Modernization Workbench solutions to deliver a full range of computer-assisted application modernization. Visit <http://www.relativity.com> for details on Relativity and the Modernization Workbench.

Discover: Document, Inventory and Assess

Modernizing your ClearPath applications starts with the discovery and assessment of the applications you are considering for change and it continues through incremental changes that return "islands" of benefits.

The Discover stage is recommended regardless of whether the COBOL application requires change. Unisys offers products that can wrap your transactions with new technologies – without requiring any changes. Discovery is generally a short-term process that uses the Modernization Workbench Application Analyzer and Application Profiler to rapidly document and analyze your applications. This process demystifies the application through an interactive "walk-through" showing how it works and whether and where there are any interdependencies on other applications and databases.

The benefits at this stage are:

- Makes modification easier and reduces errors, thereby lowering your costs
- Facilitates application training, thus reducing your training costs
- Improves routine maintenance productivity, giving you more time for new strategic applications

The Discover stage focuses on regaining knowledge that may have been lost over time. Accurate design knowledge is a necessary prerequisite to the following stages, which identify opportunities for significant improvement and gauge the level of effort needed to solve any targeted business issues.

Streamline: Simplify and Modularize

The Streamline stage is recommended for applications that are monolithic, complex, costly to maintain and difficult to change. It cleans up and simplifies your application code without changing the externally visible behavior or features of the application.

There are several options to consider during the Streamline stage:

- Obsolete data fields and dead code can be removed
- Redundant copies of code can be consolidated into a single callable routine or service
- Business logic that changes frequently or can be reused in multiple applications should be considered for conversion to a callable routine or service
- Database access should be isolated from business logic, allowing reuse of data access services from various business logic
- Presentation logic should be isolated for more flexibility in adapting to various front-end-delivery tools (Web-enabling)
- Intermediate layers could be created for accessing business logic from new platforms (Web Service, J2EE, .NET, etc.)
- Program code could be updated to take advantage of recent improvements in languages, compilers, operating system features and memory sizes
- A configuration management solution like CMplus can help reliably maintain and deploy your application
- Do nothing to your existing code; wrap transaction programs non-invasively as Web Service components

As modular components, the application logic can be more easily included in a flexible, Service-Oriented Architecture (SOA) that meets your long-term business requirements.

The benefits at this stage are:

- Reduced application complexity by modularizing and eliminating dead and redundant code
- Lower maintenance costs by making applications more streamlined, modular and manageable
- Less costly and time-consuming to apply incremental changes when changes are isolated to specific logic components
- Improved performance and operational robustness when modules are executed and tested individually
- Presentation and user navigation logic is isolated from business processing and data access logic, making it easier to expose the latter as callable services
- Provides a foundation for additional modernization with less risk involved

Transform: Service-enable, Integrate, Wrap and Evolve

The Transform stage is the first one to address specific business issues. This can typically be accomplished by service enablement or integration using middleware. Where practical, it is implemented in keeping with 3D Blueprint standards.

Service enablement and integration are focused on delivering immediate business benefits by allowing the sharing of data and application functionality with internal users, partners, suppliers and customers.

There are several options considered during the Transform stage:

- Useful application functions can be identified and exposed as callable services
- Calls to external services can be inserted into existing code after analyzing the design impact

- Scheduled file transfers and batch jobs can be analyzed end-to-end and assessed for an upgrade to message queuing and triggered processing, accelerating the transfer of information between systems and providing near-real-time or real-time updates
- Your ClearPath application portfolio can be integrated with applications on other platforms to form composite applications, using enterprise architecture concepts such as SOA and ESB
- Printed batch reports and other textual application output can be pushed in electronic form to other parts of the composite application

ClearPath integration solutions provide powerful, standards-based capabilities. Technologies such as Open Distributed Transaction Processing, Microsoft .NET, Java, XML, MQSeries and Web Services are some of the options available to integrate your transaction processing systems and databases with other applications, systems and platforms.

Where appropriate, solutions are delivered by composite applications linked as Web Services. Web Service components may be in the form of wrapped COBOL, J2EE or .NET. Regardless of the technology platform, Web Services can be reused, providing the greatest application flexibility.

The benefits at this stage are:

- Enables data sharing and process integration among customers, business partners and suppliers
- Improves data sharing and collaboration among internal constituencies
- Leverages proven business logic and robust, high-capacity platforms to meet business needs at low risk
- Lowers the costs of customer care and self-service by improving ease of use and end-user satisfaction
- Increases accessibility through a variety of media such as PC-based Web browsers and wireless devices
- Eliminates custom application programming interfaces (APIs)

Transform Technology Solutions

Wrapping as a Service

Unisys SOA wrapping solutions range from completely non-invasive (no changes to the existing application) to enabling technologies and solutions that can customize your applications for greater use and reusability. Using the non-invasive approach to SOA is relatively low-risk and low-cost. And it generally provides a solution within a very short time for a high return on your investment.

ClearPath Plus platforms support the JBoss Application Server and Java Enterprise Edition resource adapters that enable standardized, non-invasive and customizable solutions for integrating your host transactions and data.

Some level of invasiveness may be required for applications that are not in a form or granularity that easily lends itself to use as a callable service. In such cases, the application discovery/analysis tools and consulting services of the Discover and Streamline stages are key to understanding your applications and restructuring them to create cleaner, better-performing and better-understood business services.

Reuse or Build New

ClearPath applications serve as reliable workhorses, supporting core business processes and providing crucial information for day-to-day operations. The same characteristics that make mainframe applications the mainstays of business computing, optimized to exploit capabilities of the hardware and operating system, have also made them seem difficult to integrate with one another and with applications written using newer technologies.

Most core business applications consist of tens of thousands of Function Points. A Function Point is defined as one end-user business function, such as a query for an input. Basic Function Points are categorized into five groups: outputs, inquiries, inputs, files and interfaces. An average programmer can develop about 270 new application Function Points per year. With a cost ranging from \$550 to more than \$2,000 to develop a new application Function Point, it would cost \$5.5M to \$20M to replace an application with 10,000 Function Points. [Source: eAI Journal. Paul Holland. "Building Web Services From Existing Applications." September 2002.] So it makes plain business

sense to reuse existing applications that continue to provide a return on investment rather than endure the cost and risk of rip-and-replace projects.

SOA using Web Services is a best practice for reuse of existing applications. Web Services are a best practice for SOA because they provide standards-based flexibility and reusability. The Web Services model differs from the more general client/server and remote procedure call models in its emphasis on loose coupling between software components. Web Service components can reside on any platform and in any programming language, as long as the communication between them conforms to the industry standards for Web Services. Components from various sources, even those developed using different technologies and dissimilar information models, can be assembled into new business applications. Composite Applications combine existing COBOL applications with Java Enterprise Edition or Microsoft .NET using Web Services.

Many composite applications are being built using an ESB middleware layer to mediate the information flow between requesters and services. Typical ESB products provide capabilities such as routing, logging, translation, security and flow management. You can use the techniques outlined in this paper to integrate your ClearPath application portfolio with third-party ESB solutions.

OS 2200 Technology Solutions

Discover Stage

The Discover stage of our Application Modernization methodology can examine part or all of your OS 2200 application portfolio: transaction programs, batch programs, demand programs, runstreams, databases, data files and system configurations. The Modernization Workbench fully understands RDMS, DMS, ECL, VALTAB, ASCII COBOL and UCS COBOL.

Relationships among modules of a program are traced through working storage usage and call flows. Relationships among programs are traced through database usage, file usage and runstream flows. You can assess the impact of any proposed change by tracing relationships to other affected modules, programs and data containers.

This Discover process recreates application knowledge that has been lost over time as original developers moved on and documentation fell out of date.

Streamline Stage

The Streamline stage of our Application Modernization methodology is the first time it actually starts to modify your OS 2200 application portfolio, simplifying the complexity that has grown over time. General simplifications such as elimination of dead code and consolidation of redundant code are made.

Programs implemented using ASCII COBOL or other Basic mode languages are considered for upgrade to the UCS compilers and extended mode architecture. This removes the size restrictions on memory banks that have constrained program design and data capacity. In extended mode architecture, it is much easier to modularize the code and make use of middleware for composite applications.

The Modernization Workbench offers many displays and reports that help you understand your current portfolio and assess the impacts (both positive and negative) of proposed refactoring changes.

Changes such as these can cut across programmer and team boundaries that exist in most organizations. Perhaps different teams have even been using different procedures and tools for source control, build control and configuration management. This could be a very good time to standardize, making it easier for the modernization work to proceed smoothly in multiple teams.

Unisys offers CMplus (Configuration Management Program Librarian for Unisys Systems). It's an OS 2200-based development environment that provides software configuration management, problem tracking, application building and installation. It can handle Demand and Batch programs, TIP and HVTIP transactions, Fixed-Gate Subsystems and OpenDTP services. CMplus can incorporate existing build procedures and provide consistent programmer interfaces to them. Or it can establish new manageable build procedures to replace existing ad hoc techniques.

We also offer Eclipse IDE for OS 2200, a workstation-based Integrated Development Environment (IDE) containing OS 2200-oriented plug-ins to the open source Eclipse IDE.

This package provides language-specific editors for COBOL, C/C++, Java and PLUS. Some programmers can use these IDE editors while others continue using PC or OS 2200 text editors, depending on personal preferences and the needs of a particular task. The Eclipse IDE can work with many different configuration management tools including CMplus, commercial and open source products and homegrown mechanisms.

Transform Stage

The Transform stage of our Application Modernization methodology opens up the OS 2200 application portfolio for integration into composite applications. General transformations are made at this stage to expose application functions as callable services and to call external services.

As in the Discover and Streamline stages, the Modernization Workbench continues to provide many displays and reports that help you understand your application portfolio and assess the impacts of the proposed changes. Development tools such as CMplus and Eclipse IDE enable your programmers to develop, test and deploy transformative changes – quickly and with confidence.

The following sub-sections of this paper describe several specific transformations that move applications toward SOA and identify the OS 2200-enabling technologies that are used in such transformations. The sub-sections are organized by general topic, such as wrapping or calling, with a specific sub-section for each major technology choice within the topic.

- Wrapping Transactions
 - Non-invasive wrapping of a screen-oriented transaction as a service
 - Modifying a transaction program for wrapping as a service
 - Creating a transaction program to be called as a service
 - Wrapping a transaction service program as a Web service
- Calling Transactions
 - Calling a transaction program from a J2EE application
 - Calling a transaction program from a Microsoft .NET application

- Calling a transaction program via MQSeries
- Calling a transaction program via Microsoft Message Queuing
- Outbound Calls
 - Calling an external service from an OS 2200 program
- Other Transformations
 - Replacing file transfers by Message Queuing
 - Integrating with an Enterprise Service Bus
 - Replacing printed reports with electronic data feeds

Many factors go into choosing particular enabling technologies for use in a given ClearPath environment. Unisys can help you develop a comprehensive SOA roadmap strategy for technology selection and modernization.

Wrapping Transactions

Non-invasive Wrapping of a Screen-oriented Transaction as a Service

An existing TIP or HVTIP transaction that is currently invoked by users through a terminal session might provide a discrete function that would be useful as a callable service. Such a transaction might be wrapped and exposed as a service, without making any changes to the code of the transaction.

Unisys offers CITA (the Communications Interface for Transaction Applications), which enables applications to access OS 2200 transactions over a TCP connection in a request-reply mode. The calling application can be written in any language and run on any platform that supports TCP socket connections, including the OS 2200 Java environment. CITA has a configurable ability to translate between TCP byte streams and transaction messages. The calling application has the primary responsibility to construct a request that the transaction program will recognize and then interpret the response. The calling application can expose a Web Service interface to the transaction.

We also offer the Transaction Wrapper and Heritage Application Access features of Open Distributed Transaction Processing (Open/DTP). Both features enable some existing transactions that use Display Processing System (DPS) for terminal handling to be called as OpenDTP services, with no change to source code. The calling application can be written

in any language and run on any platform that supports OpenDTP-compatible middleware, including the OS 2200 Java environment. The OpenDTP input and output data buffer definitions are generated from the transaction's DPS screen definitions. Additional middleware can be used around the calling application to expose a Web Service interface to the transaction.

Various third-party vendors provide screen-scraping tools that translate between programmatic request-reply messages and OS 2200 terminal session messages. These products can expose Web Service interfaces to suitable existing transactions.

None of these non-invasive wrapping methods provide for the coordination of updates in a global transaction. If a composite application requires such coordination, another mechanism must be used.

Modifying a Transaction Program for Wrapping as a Service

An existing TIP or HVTIP transaction that is currently invoked by users through a terminal session might provide a discrete function that would be useful as a callable service. Such a program can be modified in small or substantial ways and then wrapped and exposed as a service. The degree of modification required depends on the original design of the program and how that design has evolved over time.

Many transaction programs were designed to isolate all terminal-handling and error-message generation in a common subsystem, either Unisys Display Processing System (DPS) or another (perhaps homegrown) package. Such transactions follow a common design pattern for initialization, input acquisition, processing, output creation and finalization. It is often straightforward to create editor scripts to automate most of the code changes that modify the programs from terminal-oriented to service-oriented.

Some transaction programs were not designed to isolate the terminal-handling code or have lost the isolation through changes over time. Such programs will require more work to disentangle and remove the terminal-oriented code. As in the Discover and Streamline stages, the Modernization Workbench continues to help by identifying which areas of code do processing or data access and which do terminal handling or user interface.

The transaction's input and output data buffer definitions are discovered at the boundaries between the terminal-oriented code and the processing code. These definitions are gathered up, instantiated in working storage and used as part of the new service-oriented interface code.

Once the terminal-oriented code has been removed, the program can be kept in its current TIP/HVTIP form or converted to an OpenDTP service. A TIP/HVTIP transaction passes its request and reply messages via COMAPI (Communications API), CITA, MQSeries or MSMQI (Interface to Microsoft Message Queuing). An OpenDTP service passes its request and reply messages via OpenDTP or MQSeries.

If the modified program uses MQSeries or OpenDTP, it can be eligible to participate in a global transaction with coordination of updates. Some composite applications might require such coordination to ensure that updates made by various subordinate services are committed or rolled back as a single unit.

Creating a Transaction Program to be Called as a Service

An application might be structured so that its processing functions are available internally as callable routines. As in the Discover and Streamline stages, the Modernization Workbench continues to help in identifying scattered code that performs a useful function and moving that code to a callable routine. It's easy to provide a callable service by creating a new transaction that calls the appropriate internal routines.

The input and output buffer definitions are created to fit the definition of the service.

The transaction can be created as a TIP or HVTIP transaction or as an OpenDTP service. A TIP/HVTIP transaction passes its request and reply messages via COMAPI, CITA, MQSeries or MSMQI. An OpenDTP service passes its request and reply messages via OpenDTP or MQSeries.

If the new program uses MQSeries or OpenDTP, it can be eligible to participate in a global transaction with coordination of updates. Some composite applications might require such coordination to ensure that updates made by various subordinate services are committed or rolled back as a single unit.

Wrapping a Transaction Service Program as a Web Service

Composite applications can be assembled using Web Service technology as the integration vehicle. The composite application can be implemented using a wide choice of technologies such as J2EE, Microsoft .NET, Enterprise Service Bus middleware, and others. An OS 2200 transaction program that has been wrapped or created to act as a service can be further wrapped to act as a Web Service.

The Web Service wrapper is implemented as a J2EE service endpoint. The Java Platform Enterprise Edition handles the details of defining and publishing the Web Service interface so it can be discovered and bound to by the composite application.

The service endpoint code is a thin layer that invokes the transaction program through an appropriate Unisys provided connector or through the Java Message Service (JMS). A transaction program using COMAPI or CITA would be invoked through TIP-RA, the J2EE Connector for OS 2200 Transactions. A transaction program using OpenDTP would be invoked through OpenDTP-RA, the J2EE Connector for Open/DTP. (A J2EE Connector is sometimes called a resource adapter.) A transaction program using MQSeries would be invoked through JMS.

The preferred environment for deploying the Web Service wrapper is JBOSS-2200, the JBoss Application Server for OS 2200. Running on the same OS 2200 system as the transaction program reduces network latency in the request-reply traffic and also enables easier management of the production environment. However, the Web Service wrapper and the connectors can be deployed on any system that has a compatible J2EE environment.

If the connector used is OpenDTP-RA or MQSeries, the exposed Web Service can be eligible to participate in a global transaction with coordination of updates. Some composite applications might require such coordination to ensure that updates made by various subordinate services are committed or rolled back as a single unit.

Calling Transactions

Calling a Transaction Program from a J2EE Application

Composite applications can be assembled using J2EE technology as the integration vehicle. An OS 2200 transaction program that has been wrapped or created to act as a service can be accessed as an Enterprise Information System from J2EE applications.

The composite application code invokes the transaction program through an appropriate middleware mechanism. A transaction program wrapped as a Web Service would be invoked through the J2EE Web Service mechanisms. A transaction program using COMAPI or CITA would be invoked through TIP-RA, the J2EE Connector for OS 2200 Transactions. A transaction program using OpenDTP would be invoked through OpenDTP-RA, the J2EE Connector for Open/DTP. A transaction program using MQSeries would be invoked through JMS.

A good choice for deploying the composite application is JBOSS-2200, the JBoss Application Server for OS 2200. Running on the same OS 2200 system as the transaction program reduces network latency in the request-reply traffic and also enables easier management of the production environment. However, it is also feasible to deploy parts of the composite application and the necessary connectors on other systems that have compatible J2EE environments.

If the invocation mechanism is Web Services, OpenDTP-RA, or MQSeries, the transaction program can be eligible to participate in a global transaction with coordination of updates. Some composite applications might require such coordination to ensure that updates made by various subordinate services are committed or rolled back as a single unit.

Calling a Transaction Program from a .NET Application

Composite applications can be assembled using Microsoft .NET technology as the integration vehicle. An OS 2200 transaction program that has been wrapped or created to act as a service can be accessed as a callable service from .NET applications.

The composite application code invokes the transaction program through an appropriate middleware mechanism. A transaction program wrapped as a Web Service would be invoked through the .NET Web Service mechanisms. A transaction program using COMAPI, CITA or OpenDTP would be invoked through Unisys Distributed Transaction Integrator (DTI) or Unisys Internet Commerce Enabler. The composite application and the connectors can be deployed on any Microsoft Windows system with .NET, either on an Intel node of the ClearPath Plus system or on a freestanding server. A transaction program using Microsoft Message Queuing or MQSeries would be invoked through the appropriate message queuing API.

If the invocation mechanism is Web Services, DTI or MQSeries, the transaction program can be eligible to participate in a global transaction with coordination of updates. Some composite applications might require such coordination to ensure that updates made by various subordinate services are committed or rolled back as a single unit.

Calling a Transaction Program via MQSeries

Composite applications can be assembled using MQSeries technology as the integration vehicle. Though frequently thought of as an asynchronous store-and-forward messaging system, MQSeries is also used for synchronous request-reply traffic. An OS 2200 transaction program that has been wrapped or created to work with MQSeries can be accessed as a request-reply service.

The composite application code invokes the transaction program by sending a request message to an appropriate message queue that resides on the OS 2200 system. The MQSeries for ClearPath OS 2200 product can trigger the execution of a TIP, HVTIP or OpenDTP transaction program when the message arrives. The transaction program gets the request message, processes it and sends a reply message to a queue associated with the requester.

The transaction program can be eligible to participate in a global transaction with coordination of updates. Some composite applications might require such coordination to ensure that updates made by various subordinate services are committed or rolled back as a single unit.

Calling a Transaction Program via Microsoft Message Queuing

Composite applications can be assembled using Microsoft Message Queuing (MSMQ) technology as the integration vehicle. An OS 2200 transaction program that has been created to work with MSMQ can be accessed as a request-reply service.

The composite application code invokes the transaction program by sending a request message to an appropriate message queue that resides on the ClearPath Plus system. The transaction program uses the MSMQI product to monitor the input queue for the arrival of a new message. The transaction program gets the request message, processes it and sends a reply message to a queue associated with the requester.

This calling method does not provide for coordination of updates in a global transaction. If a composite application requires such coordination, another mechanism must be used.

Outbound Calls

Calling an External Service from an OS 2200 Program

As an OS 2200 application portfolio becomes integrated more fully into composite applications, there will be occasions where it is appropriate to have a transaction or batch program call out from the OS 2200 environment to a service provided by some other part of the composite application.

Unisys provides connectors that enable calling out to services using a wide variety of technologies. Choose the connector that fits with the technology that was used to implement the service.

If the connector used is OpenDTP-RA, MQSeries or DTI, the requester and the called service might be eligible to participate in a global transaction with coordination of updates. Some composite applications might require such coordination to ensure that updates made by various subordinate services are committed or rolled back as a single unit.

Technology	Connector
EJB Web Service	Socket Listener Service of JBOSS-2200 (JBoss Application Server)
J2EE	OpenDTP-RA (J2EE Connector for Open/DTP)
MQSeries	MQSeries (MQSeries for ClearPath OS 2200)
Microsoft Message Queuing	MSMQI (Interface to Microsoft Message Queuing)
COM DCOM Microsoft .NET Java CORBA BEA Tuxedo Fujitsu TPMS	DTI (Distributed Transaction Integrator)
COM DCOM Microsoft .NET	NTSI (Messaging Integration Services)

Other Transformations

Replacing File Transfers by Message Queuing

As an OS 2200 application portfolio becomes integrated more fully into composite applications, there are increasing requirements to shorten “overnight” update cycles and provide data that is continually up to date.

Some parts of an application might have been designed with the assumption that certain data fields such as account balances and inventory levels do not change during the processing day. As in the Discover and Streamline stages, the Modernization Workbench continues to be very helpful in analyzing the read and update patterns of data records and identifying such assumptions. Designers might need to update parts of the application to work with data that will now be changing dynamically during the processing day.

It is quite common for independent applications to synchronize their data by means of file transfers that feed batch update processes on a fixed weekly or daily schedule. The operational schedules for such synchronization processes can be quite complex and usually must be completed on a hard deadline to avoid disrupting the next business day. As the requirement for data currency in composite applications shortens from weekly or daily to hourly or minute-by-minute (near-real-time), the existing batch implementations are increasingly hard to manage from an operational point of view.

Message queuing is a viable alternative to periodic file transfers. The sending application can enqueue new data as it’s generated. The receiving application can dequeue and process new data as it becomes visible. The message queuing system can be configured to batch up messages at either end, to trickle messages through at a low priority or to push messages through at a high priority, whatever is needed to meet data currency requirements. The queue monitoring and management tools can ensure that the data flow continues correctly, raising alerts when problems persist. Message queuing works equally well when the sending and receiving applications are on the same system, are on separate systems in a data center or are separated geographically.

Unisys provides message queuing capabilities that enable OS 2200 applications to use MQSeries and Microsoft Message Queuing.

Integrating with an Enterprise Service Bus

Many composite applications are being built using an ESB middleware layer to mediate the information flow between requesters and services. The OS 2200 application portfolio can be integrated with third-party ESB solutions.

ESB solutions typically build upon a lower layer of integration middleware mechanisms. Choices include message queuing, Web Services, J2EE remote invocation, .NET remote invocation, and others. Previous sections of this paper describe how OS 2200 supports these mechanisms, enabling ESB integration for both inbound and outbound requests.

Replacing Printed Reports with Electronic Data Feeds

Many business processes continue to rely on information from printed batch reports. Such information can be captured in electronic form as structured data files that are ready to import into spreadsheet programs and data analysis tools. A composite application that includes such personal productivity tools can provide a very rich, high-power interface for business process users.

Unisys offers EOM (Enterprise Output Manager) that provides such information capture from print reports and many other functions as well. EOM accepts print reports from OS 2200, MCP and many other operating environments.

Unisys also offers CIFS (Common Internet File System) that enables remote file access among OS 2200, UNIX, Windows, Linux and other systems. Application output files created on OS 2200 can be retrieved over the network by applications and analysis tools on other systems. OS 2200 applications can also create output files over the network on other systems for use by applications there.

Summary

Application Modernization has become a standard business practice. Many organizations realize that abandoning existing applications and writing new ones from scratch is simply impractical. The move to a third-party, off-the-shelf package can be both costly and risky and generally requires a great deal of customization in order to replicate the proprietary logic that has given the organization its competitive advantage. And, in addition to the expense of potentially lost business during the crossover, the need for such customization means that your agility is lost in the long run. That's because you must then rely on a third-party vendor to do updates every time you make a change request.

Unisys offers you both ClearPath consulting services and technology solutions to help you meet your business requirements, future-proof your applications and lower your maintenance costs. The enabling technologies and industry standards that we employ let you easily extend the reach and lifetime of your working systems through service enablement and integration. Besides solving your immediate business needs, the Unisys Application Modernization program can be instrumental in moving your application portfolio forward into Service-Oriented Architecture.

We can also help you develop a comprehensive SOA roadmap strategy for modernization – a cost-effective method for improving usability and realizing ROI for your systems that contain your core business processes. This roadmap can also be a steppingstone to the 3D Blueprinting™ approach strategy that gives you visibility into the way your business processes align with your core business applications.

Biography

Author

Robert Vavra is a software architect and consultant in the Unisys ClearPath OS 2200 Systems group. Bob has more than 30 years of experience working with and leading teams in systems software areas such as databases, middleware, systems management, operating systems, CASE tools and compilers. His architectural experience includes applications distributed across a mix of platforms including ClearPath, UNIX, Windows and Linux. Bob's first distributed application was developed by a small team he led, deployed successfully in 1986 and is still in productive use today at many client sites. He holds B.S. and M.S. degrees in Computer Science from Michigan State University and the University of Minnesota respectively. Bob has also co-authored two patents.

The author would like to thank his former colleague, Stan Wrubleski, for much of the content provided in this paper. Stan's input has been vital to successful communication of the Unisys Application Modernization process.

For more information, contact your Unisys representative.

Or call:

1-800-874-8647, ext. 405 (U.S. and Canada)

00-1-585-487-2430, ext. 405 (Other countries)

In a hurry to learn more? Visit:

<http://www.unisys.com/cp/dorado>

For more details, visit:

<http://unisys.com/cp/ecomunity>

1-800-874-8647, ext. 405 (U.S. and Canada)
00-1-585-487-2430, ext. 405 (Other countries)

In a hurry to learn more? Visit:
<http://www.unisys.com/cp/dorado>

For more details, visit:
<http://unisys.com/cp/ecomcommunity>

This document is not a contract and does not create any binding representations or warranties by Unisys.
All representations are contained only in the applicable agreement signed by the parties.

The information contained herein is subject to change without notice.

© 2008 Unisys Corporation. All rights reserved.

Unisys, the Unisys logo, ClearPath, and 3D Blueprinting are registered trademarks or trademarks of Unisys Corporation. Intel is a registered trademark of Intel Corporation. Microsoft and Windows are registered trademarks of Microsoft Corporation. Linux is a registered trademark of Linus Torvalds. All other brands and products referenced herein are acknowledged to be trademarks or registered trademarks of their respective holders.

