



Application Modernization

An HP NonStop Perspective

Table of contents

Introduction	2
Overview	2
Risk reduction	2
Cost reduction	7
Improving agility	8
Application modernization domains	8
Modernizing Green Screen application user interfaces	9
Why modernize?	9
User devices and networks	9
Invoking application services	10
The Web GUI	12
Database modernization	13
Limits relief	13
Standards compliance	13
New functionality	14
Access	14
Migration and database refactoring	15
Architecture and integration	15
Architecture	15
Integration	16
Platform, development tools, and frameworks	17
Platform migrations	17
Development tools	18
Frameworks	18
The steps to application modernization success	19
Assessment	19
Establish objectives	20
Create a plan	21
Execute the plan	22
For more information	23
Call to action	23

Introduction

Application modernization can provide significant benefits for HP NonStop customers, beyond improved applications. Application modernization encompasses software development activities that are undertaken to reduce risk, lower costs, and improve agility. Risk reduction and avoidance, and improved agility are the principal benefits from an application modernization effort. Cost reduction is a common goal of application modernization, but may be a long-term benefit, if there are significant up-front costs. Application modernization does not guarantee improved agility, but a focus on improving agility throughout the application modernization planning phase can enable better agility. There are some risks associated with application modernization itself, but with careful planning, those risks can be effectively managed.

Overview

This paper takes a broad view of NonStop application modernization, and is not limited to specific kinds of applications, vertical markets, or geography. This paper also does not address the separate, but closely related area of operations modernization. The potential value of each application modernization project can be estimated by reviewing the major activities within each application modernization domain, assessing the expected benefits, and determining what is realistically achievable. Before looking more closely at those aspects of legacy applications for which these benefits may be achieved, the benefits themselves should be more clearly defined. The potential benefits are the basis for undertaking an application modernization effort.

The next section identifies major risks and the general application modernization activities that not only reduce the risks, but also provide tangible benefits. Costs, which can be reduced or eliminated by undertaking certain application modernization activities, are then identified. Finally, significant improvements in agility that can be achieved through application modernization are identified.

Four major domains or aspects of application modernization are then introduced and examined in detail. Finally, a few basic steps are provided to show how best to achieve the benefits of a successful and low risk application modernization effort.

Risk reduction

Application modernization activities are undertaken to address specific risks. The benefits of risk reduction vary by the nature of the risks and the strategies employed to reduce them. The following examples illustrate some typical risks that HP NonStop customers must address, specific application modernization activities that can reduce those risks, and the benefits that can result.

Risk 1

Users experiencing poor performance, inability to expand application capacity to accommodate increased loads, lack of availability of HP NonStop hardware components and products approaching their “end of support” dates.

These risks are usually due in part to discontinuing support for older hardware or software and the limitations that exist on older platforms, such as the lack of product enhancements and new product releases, especially for software. HP NonStop product support status and important dates for product support are provided in the Software Products Maintenance list and the Hardware Products Maintenance List. These lists are updated periodically and provide advance notice of the dates when the product support levels change.¹ Java and TS/MP (Pathway) are examples of products that are available on older platforms, such as the S-series, but which have enhanced capabilities and increased limits in the versions available on the Integrity NonStop platforms. The maintenance costs for products nearing their end of support may also increase due to increased support costs.

¹ Current maintenance lists: <http://h20223.www2.hp.com/nonstopcomputing/cache/76970-0-0-121.html>

Actions

Undertake an application modernization effort to upgrade applications to take advantage of current platforms and software. The basic form of this type of application modernization is a migration. The objective of the migration is to move the applications and databases from the old platform and software infrastructure to a current platform environment, with minimal changes. Even if other application modernization activities are planned, limiting the first phase to a migration of the current applications and database to an Integrity NonStop system or a NonStop BladeSystem can make the effort less complicated.

Benefits

The application capacity can be easily expanded, performance will be improved, application availability increased and the risk of outage significantly reduced. There are two additional benefits; maintenance costs may be significantly reduced and the total cost of ownership (TCO) of new HP NonStop systems will be less.

Risk 2

Inability to find technical staff with NonStop application development skills and experience, lack of knowledge of TNS applications and development tools, inability to make timely changes or to make them at all.

These risks are partly due to the small number of developers that are familiar with old NonStop products and older NonStop development processes. The lack of newer open source tools and standard application programming interfaces (APIs) could make development more difficult and could extend the development lifecycle. Few new developers are interested in learning and becoming experts in old technologies such as the TNS execution environment, the TAL language and compiler and the old TNS compilers for COBOL, C, and C++ and their associated development tools. There are still many TNS applications in use today and those applications are using out-of-date technology.

Actions

These concerns are common and help drive two on-going efforts by the HP NonStop division. First, the Open System Services (OSS) UNIX[®]-like environment is being continually enhanced. Second, more open source and standards based products are being introduced, including widely used development tools, such as Eclipse, for both Guardian and OSS application development.

A number of available application modernization options can be undertaken in combination or sequentially in a low risk phased approach. First, moving from TNS compilers to native mode compilers provides a significant base for additional modernization. Application source-code translation, such as TAL to C or C++, is another option to consider. Third-party products and services are available to translate TAL directly to native mode C or C++.

NonStop development organizations that have not yet begun to use workstation based integrated development environments (IDEs) like Visual Studio or the open source Eclipse platform, should consider doing so to provide a more attractive development environment for new developers. Visual Inspect should also be installed to make debugging easier. Education and training are readily available for the reduced amount of NonStop-specific knowledge that is still required.

Many newer HP NonStop products require the OSS environment. If OSS is not used currently, it may be required in the future to take advantage of new products and capabilities that are only provided in the OSS environment. The OSS environment implements standard POSIX APIs and enables the use of proven open source software, including many commonly used libraries, tools, and utilities. Migrating application development and execution to the OSS environment may also make development easier and shorten the time required for new developers to become productive on NonStop systems.

Benefits

The pool of potential NonStop application developers is increased due to the use of standard languages and development tools. Developer productivity is improved. Application performance may be improved when migrating to native mode compilers and with careful language modernization the applications should become easier to maintain and more portable.

Risk 3

Inability or difficulty in addressing:

- Legal, regulatory, and standards compliance requirements (for example, security and reporting)
- The need to enhance applications (for example, to enable new markets, new geographies, implementing new business functions, and the like)
- Access from external applications to NonStop applications' functionality
- Access to external applications' functionality from NonStop applications
- External applications need for direct access to the database
- Application availability
- Complete site outages and disaster recovery

These types of risks are usually due to some combination of these problems:

- Application code is difficult to modify
- The database is difficult to modify
- Only proprietary application interfaces are available
- Few developers fully understand the application programs or the database design
- There is no ODBC or JDBC access to the database

The underlying causes of many of these problems and the resultant increase in risks are outmoded application architectures, old application design, the lack of use of open standards and technologies, and the use of less well known programming languages, such as TAL, ADA, PASCAL, FORTRAN, COBOL74, TACL, and pTAL.

Older NonStop applications were designed at a time when proprietary protocols and platform specific languages and software were widely used. These older applications had limited database options, limited security options and fewer standards based connectivity options. TAL and COBOL, as well as other TNS languages, were often used for application development. Although these applications have been in production for many years, they are based on old architectures and designs and many have aged past the point at which they should have begun undergoing modernization.

Actions

Modernization activities that address these risks will be guided by an understanding of enterprise architecture requirements and enterprise standards. The initial activity is an assessment of the current applications and databases. The assessment will help identify the modernization activities that are most urgent and will serve as a basis for developing a broader long-term modernization roadmap.

Service-oriented architectures (SOA) are being widely adopted to increase agility and reuse, but also to address application integration concerns. Security and availability, including disaster recovery are important architectural considerations that may also drive modernization activities at the architecture level. Changes in architecture can be undertaken in phases. The use of standards-based communications interfaces, such as TCP/IP sockets and simple object access protocol (SOAP), and the use of standard programming languages and interfaces allows for easier invocation of external application services, an easier way to expose NonStop application services to external applications, and easier integration of new custom functionality and third party software components. Application integration is discussed in more detail later in this document.

Older databases, such as those built with Enscribe, do not have as many open source or third party reporting options as relational databases, such as NonStop SQL/MX, nor do they provide standards based access methods, such as ODBC and JDBC. NonStop SQL databases also support online reorganization, which can facilitate database changes. The assessment may identify multiple reasons for undertaking database modernization.

Most NonStop databases maintain database consistency with the HP NonStop Transaction Management Facility (TMF). Databases that do not have TMF protection should be re-evaluated. TMF protection may now be appropriate and TMF is a prerequisite for installing the Remote Database Facility (RDF) for maintaining a backup database on a remote system.

Database modernization is discussed in more detail later in this document.

These modernization activities may include significant architecture and design changes, as well as significant program changes. A good modernization roadmap developed from a comprehensive assessment will include the prioritization of the identified changes and show how the changes can be phased in gradually.

Benefits

NonStop applications will comply with new and emerging business requirements. NonStop applications will be easier to enhance or extend, and if appropriate, services and data stored on the NonStop systems can be more easily made available in a standard and secure manner to applications on other systems. The modernization changes should also reduce the time to develop and deploy new products and services.

Another important benefit is the improvement in the availability of the NonStop applications. And finally, an optional benefit from the deployment of the applications infrastructure changes is the provision of the software foundation that is required for replicating data to a backup site and enabling backup site takeover in the event of a major primary site outage.

Risk 4

Old terminal devices and networks are difficult to maintain, application user productivity is reduced because of the user interface (UI), the UI imposes limitations on work flow and makes workflow changes difficult, UI changes are becoming more difficult or impossible to implement, UI device interactions with host applications are not secure, and few developers possess the knowledge and skills required to effectively maintain current UIs.

If these risks are present, they are usually due to the use of older networks, older terminal devices or terminal emulators (for example, 65nn or 3270) and user interfaces that are implemented in SCREEN COBOL (SCOBOL) or in old custom PC client applications. Some older devices and networks may have no immediate support or cost issues, similarly some old applications that are accessed through terminal emulation may also have no support or cost issues. The cost of replacement and the limitations imposed by the use of outdated devices and networks are likely to be growing problems. Most installations using older networks of terminals and devices or terminal emulation on PCs have increasing risks in this area.

Few current developers have experience with Pathway/iTS TCP and terminal programming (particularly SCOBOL), 65nn or 3270 terminals or emulation, intelligent device support (IDS), extended general device support (GDSX), SNA or any other terminal protocols from the era before personal computers.

Old style text-only and block-mode UIs cannot easily, if at all, provide the benefits of a modern GUI.

Actions

Internet Protocol (IP) networks can be phased in to replace asynchronous, bisynchronous, SNA, and other old networking protocols. Wireless IP services are now available in places that still have limited or no fiber or terrestrial line network infrastructure. There are also a number of security options for IP networks. Even if there is no immediate network or connectivity problem, planning should be initiated for upgrading old and outdated network infrastructure for future risk avoidance.

Similarly, planning should be initiated for the replacement of obsolete terminal devices, even those that appear to be functioning well. Specialized devices should be upgraded to the latest appropriate technology. Classic terminal devices are typically upgraded to PCs or workstations.

Network and device upgrades often have to be coordinated and phased in gradually. The network upgrades may require changes in the UIs and will likely require making some application front-end changes and some modifications to the applications may also be required.

Application users have come to expect the availability of graphical user interfaces (GUIs). Modern GUIs can reduce new user learning time and make applications easier to use. There are also many GUI features that can improve user performance, such as:

- Mouse-based navigation
- Tabbed panes
- Drop down menus and lists
- Pop-up windows
- Calendars for date input
- Radio buttons
- Micro-help (including mouseover messages)

With modern GUI technology, user productivity can be increased by reducing multiple screen interactions and simplifying workflow. The ability to control layouts and font style, size, weight, and color can enable improved readability and productivity.

Specialized GUI development tools are built-in or available as plug-ins for all major IDEs. These development tools make it possible to prototype and test new interfaces rapidly or to modify existing ones easily. The tools also enable Internationalization and localization, when required.

Benefits

Standardizing on IP network technologies may reduce network costs, and the use of modern network protocols and open GUI technologies will greatly facilitate the replacement of old proprietary devices with low cost standards based commodity devices. The number of skilled open GUI developers is substantially larger than that of proprietary (for example, 6530, and 3270) GUI developers.

Effective use of GUI technologies can improve user performance. The use of open source GUI technologies and associated tools can reduce maintenance time and new UI development time, which will help improve the agility of the organization.

Table 1: Application modernization actions and benefits

Application Modernization	
Actions	Benefits
<ul style="list-style-type: none"> • Migrate applications and database to new Integrity NonStop platform (single or multi-core) 	<ul style="list-style-type: none"> • Increased application capacity • Improved performance • Application availability maintained or increased • Risk of outage significantly reduced • Maintenance costs reduced • Better TCO
<ul style="list-style-type: none"> • Migrate to standard languages • Migrate from TNS compilers to native mode compilers • Adopt IDEs for software development • Adopt modern debuggers: Native Inspect & Visual Inspect • Install and gain experience with OSS • Migrate application development and deployment to OSS 	<ul style="list-style-type: none"> • Increased number of potential NonStop developers • Improved developer productivity • Easier application maintenance • Easier to develop and deploy new applications • More portable applications • Potential application performance improvements
<ul style="list-style-type: none"> • Assess the current applications and databases environment including business and enterprise IT requirements and develop a migration plan • Phase in modern architectures and designs, particularly with a services approach • Migrate to standard languages, APIs and frameworks • Migrate the database to NonStop SQL/MX • Implement TMF, if not already in use • Establish a remote backup site and install RDF for remote database backup 	<ul style="list-style-type: none"> • Applications can be brought into compliance with new and emerging business and IT requirements • Applications will be easier to maintain and enhance • The reuse of services will be facilitated • Reduced time to develop and deploy new products and services • Services and data stored on the NonStop systems can be easily and securely exposed to external applications using standard APIs and protocols • Applications will be more portable • Application availability can be improved • Application continuity can be maintained even with a primary site outages
<ul style="list-style-type: none"> • Upgrade outdated network infrastructure and replace asynchronous, bisynchronous, SNA and other old networks with Internet Protocol (IP) networks and standard Internet and web protocols • Replace obsolete terminal devices with new standards-based devices • Implement or enhance network and application security • Upgrade UIs to use modern open source and standards-based • Adopt GUI development tools and IDE plug-ins for GUI development, testing and maintenance 	<ul style="list-style-type: none"> • Network and device maintenance costs may be reduced • Network and device support costs may be reduced • Improved application security • Increased number of potential GUI developers • Improved application usability • Improved user productivity • Reduced application UI development and maintenance time • Improved IT organization agility

Cost reduction

Application modernization takes time and requires development and operations resources to achieve the desired objectives. Additional external resources, such as HP service professionals, may also be employed to ensure a timely and successful modernization. All these resources have costs, so the benefit of cost reduction from application modernization is usually based on reducing a specific cost or taking a broader view, such as reducing the TCO over a number of years.

Specific cost reductions may be achieved through the reduction or elimination of license and maintenance fees for terminal emulators, networking products and middleware. These reductions are largely due to replacing proprietary technology with lower-cost standards based products or open source technology.

The TCO may be reduced through a combination of application modernization driven cost reductions, including the specific cost reductions just described. The TCO can be improved by upgrading application platforms from S-series or earlier model systems to Integrity NonStop systems. There are other important considerations besides lower maintenance fees. For example, the TCO can also be improved from the ability to get lower cost “open standards/open source” developers and the reduction in time and effort required to enhance legacy applications or to deploy new applications.

Improving agility

Agility is now an important objective for most information technology (IT) organizations. Programs to improve agility are being developed with an enterprise wide scope. Service organizations such as HP EDS and HP Consulting and Integration can help assess an organization’s agility and assist organizations in developing processes for improving and monitoring the agility of the organization.

When enterprise wide programs are developed to improve agility, they will likely apply to the NonStop applications environment. These programs may have a significant impact on the NonStop applications in several ways. They may propose or establish overall enterprise architectures (SOA, event-driven architecture (EDA), message-based architectures and so on), enterprise wide standards, and preferred development and deployment technologies, including web and application frameworks.

The objectives that drive agility improvement include a better alignment between the business and the IT organization, improved business efficiency, and the ability to respond rapidly and effectively to the many types of changes that can have an impact on the business. NonStop application modernization activities can contribute to these objectives in a number of ways. They can make applications easier to use, easier to extend and easier to maintain, resulting in both a reduction in development time and effort. They can make NonStop application services and data more easily accessible to other applications—including those on other platforms. Finally, NonStop application modernization activities can help achieve compliance with enterprise architectures, standards, and technologies. All of these NonStop modernization benefits help improve the agility of the NonStop applications environment.

Application modernization domains

Application modernization activities are usually focused on making changes that result in the most needed benefits. The HP NonStop Division has identified four domains that account for most HP NonStop application modernization activity.

The first domain is the modernization of old application user interfaces. Many customers still have applications that were developed for old terminal devices. The modernization efforts for those types of applications are called “Modernizing Green Screen Applications.”

The second domain, “Database Modernization,” addresses the modernization of application databases.

The third domain, “Architecture and Integration,” addresses changes to improve agility, ease application maintenance, and facilitate application integration.

The fourth domain, “Platform, Development Tools, and Frameworks,” covers the fundamental modernization of application platforms and the significant benefits that can be obtained from improvements in how applications are developed and deployed.

The first three domains may not apply to every NonStop applications environment; however, the fourth domain is relevant for all NonStop application environments. The following sections will describe each of these domains in more detail and the application modernization activities that are associated with each domain.

Modernizing Green Screen application user interfaces

“Green Screen” applications have text-based user interfaces that were originally developed for terminals with cathode-ray tube screens. Some of these early display devices had green text displays, hence the name “green screen”, although there were terminals with white, yellow, blue or black text and some could display multiple colors. The “green screen” terminals include a range of devices including the IBM 3270 family of terminals and the Tandem 6530 family of terminals, both of which could interact with host systems in block mode and had advanced screen-formatting capabilities. On the low-end, there was a variety of simpler teletype display devices, also referred to as “glass” teletype (TTY) devices, which were used interactively or in conversational mode.

Why modernize?

The modernization of “green screen” user interfaces is usually undertaken because of some compelling reasons, such as:

- Terminal devices are becoming too difficult to replace or repair
- New applications or functions need to be provided to the current terminal users (and the new functionality requires a GUI)
- Old terminal networks are being eliminated and replaced by IP networks
- Data streams between the user and the host need to be secured
- Host applications are being replaced or significantly modified and a GUI is being introduced because:
 - A new application requires a GUI
 - There is an organization standard for GUIs, which needs to be adopted
 - A new GUI will make older applications easier to learn
 - A new GUI will make older applications easier to use
 - The use of GUI standards makes applications more portable
 - Internationalization/localization requirements are easier to address with a GUI
 - Workflow and multiple application access is easier with web GUIs
 - Developers of standards-based GUIs are widely available

The move to standard IP networks and standard protocols and device interfaces also eliminates the need for intermediate device handling software programming on the NonStop systems, such as GDSX coding and the use of some custom device handlers.

User devices and networks

General-purpose terminals will most likely need to be upgraded to PCs or workstations. Although this implies a greater degree of support than was required for old terminals, such upgrades have been occurring for over a decade and the challenges are well understood and solutions readily available. Specialized devices may present more of a problem, since the number of potential suppliers may be very limited and device capabilities and limitations will drive any modernization plan. Another potential problem arises with terminal devices that are not under the direct or indirect control of the organization seeking modernization. In some of these situations, outside parties can be encouraged to upgrade by providing incentives. Other internal divisions may have their own terminal infrastructure, (for example, 3270 devices) which has network connections to the NonStop systems to access NonStop applications. In such cases, it is likely that the other division may also be concerned about modernization. Joint planning is required, but there are multiple approaches to modernizing in such a way that each division benefits.

Although “green screen” modernization is often equated with the replacement of an old text-based user interface with a web browser-based solution, that is just one of the options. For some environments, it may be appropriate to phase in modernization by upgrading the network to allow the use of the IP protocol and initially just replace old terminal devices with PCs or workstations running terminal emulators.

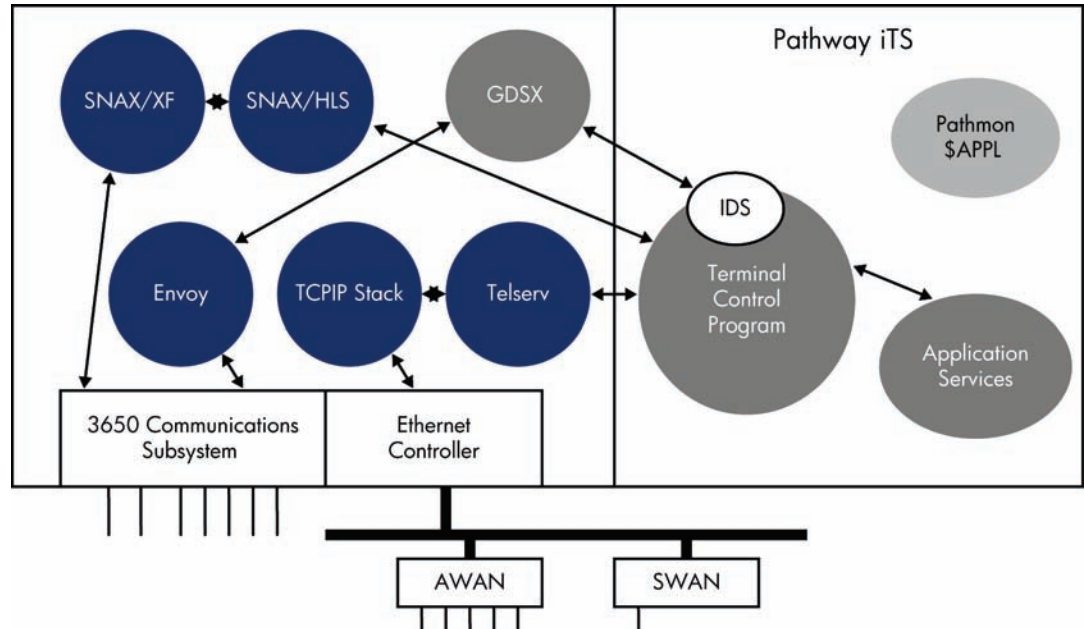
The modernization of terminal devices is closely tied to the modernization of the network. Dial-up, asynchronous, byte synchronous and bit synchronous protocols, and associated networking devices may need to be replaced. Most local area networks (LANs) and wide area networks (WANs) have standardized on IP as the backbone protocol. Dial-up connections also use IP with the point-to-point (PPP) or Serial Line Internet Protocol (SLIP), for example. Modernization, as it relates to the NonStop systems network is very straightforward. The old and proprietary networking devices and protocols should be replaced with IP networking devices and LAN connections and the TCP/IP protocol should be adopted. Ultimately, the primary NonStop network protocol should be IP, and Ethernet controllers or IP CLIMs should be the only networking hardware required as part of the NonStop hardware configuration.

Network changes have to be carefully planned, tested, and phased in so as to minimize risk and disruption of service. New AWAN controller sales stopped at the end of 2008, and that is a consideration that would help guide planning. Both HP ServerNet Wide Area Network (SWAN) concentrators and asynchronous wide area network (AWAN) terminal servers should be carefully phased out as circumstances permit.

Invoking application services

There are different types of “green screen” applications deployed on HP NonStop systems. 6530 and 3270 terminal Pathway applications have user interface screens coded in SCOBOL and executed by a Terminal Control Program (TCP). The SCOBOL programs access business services by exchanging messages with application server programs. These server programs may have been written in COBOL, C, C++, TAL, pTAL, or other languages that are available on the HP Nonstop platform. The business application server processes are not typically aware of the network protocols that are being used between the devices and the TCP, nor are they aware of whether the device is a terminal, a PC with terminal emulation, or a fat client application. The old devices may be connected via many different types of controllers and protocol stacks, and some require the use of GDSX processes and Pathway Intelligent Device Support (IDS) within the TCP, see Figure 1.

Figure 1: Old Terminal Device Networks Accessing Application Services



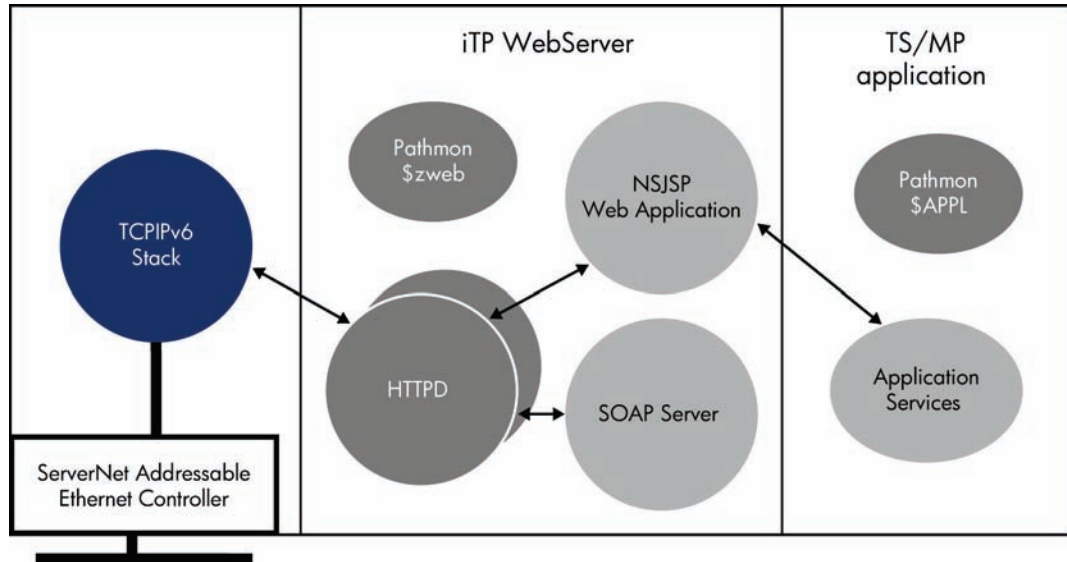
The modernization of the 6530 and 3270 terminal applications is usually accomplished by replacing the old terminals with workstations or PCs running an Internet browser or a fat client. In both cases, the host communication usually takes place over HTTP or HTTPS on a TCP/IP protocol stack. The iTP WebServer is used to handle connections on the host side. The iTP WebServer sends user requests received from a browser or other software to a TS/MP serverclass that processes the user requests, provides host side UI support and invokes specific business application services via Pathsends. The most common serverclasses used to process the user requests are the NonStop Servlets for JavaServer Pages (NSJSP) SERVLET serverclass or the NonStop SOAP server serverclass, which are both shown in Figure 2. These web architectures are highly scalable, highly available, and widely deployed.

NSJSP web applications may be implemented using standard Java APIs, such as JavaServer Pages (JSPs) or open source Java web frameworks, such as Spring (MVC, Web Flow, Web Services or remoting) and JavaServer Faces (JSF) or Axis2 Web Services, for example. There are many other Java frameworks and Java implementation options that may be considered for use.

A SOAP Server could be provided by the NonStop SOAP product or the new Axis2/C standards-based open source web services engine. There are other non-Java implementation options that could also be used.

The web services model is not new. The remote server call (RSC) products provide a similar form of client communication for accessing host application services.

Figure 2: Modern Web Host Environment for Accessing Application Services



The Web GUI

A modern web GUI has a client side and a server side. The client side typically includes a browser, some downloaded script code, and possibly some executable components in addition to static page contents. Stand-alone client programs, which do not require a browser, may also be used. JavaScript (and its dialects, ECMAScript and JScript) is the dominant language used for scripting on the client side, but other scripting languages, such as VBScript and JavaFX Script may be used. There are many different types of client runtimes and executable components, including Java applets, ActiveX controls, Adobe Flash, Adobe AIR/Flex, Silverlight and various component libraries.

Most web GUI clients communicate with the server side using HTTP Uniform Resource Locators (URLs) with argument strings, but some use web services (either SOAP based or representational state transfer [REST] style). There are two widely used formats for encoding information that is passed between the client and server sides of a modern web GUI. The Extensible Markup Language (XML) is used by some asynchronous JavaScript and XML (AJAX) implementations and by clients that use web services. JavaScript Object Notation (JSON) is used by some AJAX implementations and some non-AJAX JavaScript clients that want the benefit of passing simpler data structures.

GUI clients may invoke business application services by passing URLs with arguments to Java objects on the HP Nonstop system. The Java objects, running in an NSJSP container, can access business application services using JToolkit. GUI clients may also use web services to access business application services on the HP NonStop systems. Both SOAP and RESTful web services can be exposed on the HP NonStop system. The server side web services processing can be provided by NonStop SOAP version 3 or 4, which does not require Java, or the Java Axis2 implementation can be used if a Java implementation is preferred.

GUI implementations still include "fat clients" similar to those deployed during the era of client/server architecture. The new "fat clients," which are similar to stand-alone desktop applications, are called Rich Internet Applications (RIAs) and may be implemented in Java, JavaScript or other scripting languages, and Visual Basic or other .NET languages. The "richness" comes from the sophistication of the user interface and the type of content that can be displayed. RIAs may be downloaded from a host system via a request from the browser.

One additional feature that some RIAs provide is the ability to continue to provide the GUI and accept input, even when the network or host system is unavailable. The RIA can queue input to a local file or database during an outage. When service is restored, it can automatically begin the transmission of stored data to the host application for processing. As responses are received from the host, they can be queued for later examination by the user, at their convenience.

Although most applications will see a performance improvement after modernization, others may not. If improved performance is an objective, the choice of technologies and implementation options may be limited, but GUI performance improvement may still be achievable as part of the modernization.

The quality of a UI does not improve simply because modern technologies have been used to implement it. The quality depends upon good UI design and adherence to human factors design principles. It is also useful to let potential users test prototypes of the GUI to identify possible problems and changes that might improve usability. Incorporating GUI changes suggested by the eventual users will help increase user acceptance.

Database modernization

There are four technology drivers for database modernization: limits relief, standards compliance, the need for new functionality, and the need to provide access to data. Business requirements ultimately drive database modernization, but the drivers constrain the selection of NonStop database products and influence their configuration and deployment. Legacy NonStop applications typically use one or more of these database products: Enscribe, NonStop SQL/MP, and NonStop SQL/MX. Some OSS applications do make limited use of flat files within the OSS file system, but rarely to the extent that they are regarded as the applications database.

Limits relief

Both NonStop SQL/MP and NonStop SQL/MX allow over 400 partitions per table (depending on key sizes and available resources), but Enscribe files are limited to 16 partitions. Either an increase in the amount of data that needs to be stored or the need for more partitions for improved performance, suggest an upgrade from Enscribe to NonStop SQL/MP or NonStop SQL/MX. Note that one benefit of an increased number of partitions is the possibility of significantly increasing the amount of table data stored in disk cache. NonStop SQL/MX also has increased limits for other essential database entities, such as the number of tables, view size, and table size.

Standards compliance

NonStop SQL/MX implements ANSI 1999 and ANSI 2003 SQL standards, and NonStop SQL/MP implements 1989 SQL standards. Corporate standards compliance may suggest a need to migrate to NonStop SQL/MX. NonStop SQL/MX implements ANSI standards for catalogs and schemas, identifiers, data types, and views. IEEE Floating point data types and Unicode UCS2 data types are also available in NonStop SQL/MX.

Standards are especially important for database access, and they are addressed in more detail in the following Access section.

New functionality

Legacy applications rarely require changes in how they interact with their database as part of routine maintenance. The need for new database functionality is more likely to be driven by new applications or significant modifications to the current applications. Enscribe and NonStop SQL/MP database products are considered stable and are no longer regularly enhanced. NonStop SQL/MX continues to receive significant functional enhancements and is the recommended database product for new applications. NonStop SQL/MX has more of the features that database designers and administrators expect such as:

- Accessing rows with rowsets
- Publish/Subscribe and queuing services
- A Visual Query Planner
- A suite of built-in functions
- Triggers (before and after)
- Referential Integrity
- Stored Procedures
- Grant/Revoke security
- Full integration with Java technology
- Availability of modern development and management tools

Old applications may have been implemented without transaction protection. Enscribe databases do not require the Non Stop transaction management facility (TMF) protection, nor do NonStop SQL/MP databases. The introduction of TMF can significantly improve database integrity and provide a foundation for synchronization with a backup database using the remote database facility (RDF). Applications that were implemented with unaudited or only partially audited databases can be modified to include transaction logic, but transaction protection can also be phased in with little or no programming change, with the NonStop AutoTMF product.

The addition of RDF to the application environment (and possibly RDF/ZLT to ensure no lost transactions, in the event of a site outage) would make it possible to maintain a backup database. With the appropriate applications environment in place on the backup site, a complete primary site outage would result in a takeover by the backup site using its backup copy of the database, ensuring database integrity and database availability for the applications.

Enhancing database security, modernizing inquiry and reporting, and the need to address data governance are additional reasons to consider upgrading to a NonStop SQL/MX database.

Access

Improving access is another reason to modernize a database. The objective is to enable secure remote access to the database or to modernize access to use current standards based methods, where previously there was only limited access. The access improvements may benefit current NonStop applications, new NonStop application development, third party tools (for example, reporting utilities) that need access to data on the NonStop system and external applications that also require access to data on the NonStop systems. The available options will depend on the database products that are currently used. The NonStop ODBC and JDBC products do not provide access to Enscribe databases. Third party database middleware products or custom developed solutions can provide access to Enscribe, but development and maintenance costs make such solutions less attractive if there are acceptable alternatives.

Both NonStop SQL/MP and NonStop SQL/MX can be accessed remotely from Windows® platforms through NonStop ODBC products. ODBC access to NonStop SQL/MX from UNIX platforms is provided by DataDirect Connect for SQL/MX. NonStop SQL/MP can also be accessed remotely via JDBC, but only through third party products. NonStop JDBC Type 4 drivers for other platforms are only available for accessing NonStop SQL/MX databases.

Providing standards based access to the data on a NonStop system is one of the requirements that suggest consideration of a database migration from Enscribe to NonStop SQL/MP or NonStop SQL/MX, depending on the specific requirements.

Migration and database refactoring

If there are enough potential benefits to justify a migration from Enscribe to a NonStop SQL database, the migration may be phased in through a low risk migration based on the Carr-Scott Escort SQL product. It will allow for the replacement of Enscribe files with NonStop SQL/MP tables and provide the benefits of NonStop SQL/MP without requiring any significant application reprogramming. If the benefits of NonStop SQL/MX are required, it may be possible to use the NonStop SQL/MX product with the SQL/MP tables. If so, applications will need to be modified to use NonStop SQL/MX effectively, prior to migrating the Enscribe database to NonStop SQL/MX.

Upgrades may be appropriate for existing NonStop SQL/MX implementations that are not current or that have applications that can benefit from moving to native MX tables from MP tables.

Finally, even current database implementations may benefit from some database refactoring.

Database refactoring refers to small changes in database schemas including tables, partitioning, indexes, stored procedures, triggers, constraints, and possibly to the data itself. These types of changes are usually undertaken because of performance considerations, new application database access requirements or to enhance legacy applications. The goal is to improve performance and to support new application functionality effectively without requiring any changes in the existing application programs.

Architecture and integration

Architecture

There are many different views of enterprise architecture and especially application architecture. Forrester, Gartner, and other market research firms have analyzed architectural options and surveyed IT organizations regarding their adoption of architectures and have concluded that certain types of architecture can provide significant advantages for business applications. These architectures include SOA, EDA, and web-oriented architectures (WOA). All these architectures can coexist and they are not inherently tied to specific technologies, though there may be very strong associations, such as SOAP and representational state transfer (REST, also called RESTful) web services with SOA. The success of certain architectural approaches has led to the documentation of specific reference patterns (both at the architecture and the design level).

Many legacy NonStop applications already implement a SOA, and most new NonStop applications should also be architected using a SOA approach. There are new standards-based technologies (for example, Axis2/Java and Axis2/C) and corresponding NonStop products, to enable web services and SOA implementations on Integrity NonStop platforms.

Re-architecting applications is often identified as an option for application modernization. First, it is necessary to establish a strategic architectural direction to guide applications development. The architecture strategy will guide modifications to legacy applications as well as the creation of new applications. Adopting different architectural approaches may also provide other valuable benefits. For example, an SOA oriented analysis could provide a better understanding of the business processes.

The scope of architecture changes for legacy NonStop applications, however, should be limited to those that directly support business objectives, are consistent with any established enterprise level architecture and will provide clear benefits, including a good return on the investment in time and effort.

Certain business requirements are important enough to justify legacy application architecture changes. They include the need to improve application availability, improve application security, replace obsolete or proprietary network protocols with TCP/IP, HTTP (and other standard protocols), and the elimination of proprietary middleware when viable open source options exist. The need to meet scalability and maintainability requirements is another reason for re-architecting an application.

Architectural changes are not the only reason for making program changes, although many of the following reasons for making significant code changes do have an architectural aspect. Programs may benefit from refactoring, which can improve program performance, maintainability and facilitate integration. The manageability of programs can be improved by instrumenting them for status/monitoring, online configuration changes, and statistics and performance metrics capture. Adding or enhancing program event logging may provide additional benefits.

Programs can also be modernized by translating them from an old language to a newer or more common one (for example, TAL/pTAL to C, COBOL to C, or C to Java). There are third-party translation products available that can programmatically do much of the translation.

Integration

Integration is another major focus for most application modernization efforts. A focus on integration will make it easier for NonStop applications to access data and to share common services. At an enterprise level, it will make it easier for applications on other platforms to access data and services on the NonStop platform and for applications on the NonStop platforms to more easily access data and services on other platforms. From an implementation perspective, effective integration reduces redundancy and facilitates the creation of new applications and business processes.

There are a number of implementation techniques for facilitating integration. For example, a common design approach can be developed for exposing services. Although there are several web services technologies that may be adopted, design standards for APIs and dynamic link libraries (DLLs) also foster application integration on the NonStop platform itself. The conversion of application programs to native mode or to Java also facilitates local NonStop platform application integration. This is because new products and functionality on the Integrity NonStop platforms are now primarily provided as native mode libraries or in Java.

Web services technologies, such as SOAP and REST, and other remoting technologies, such as Hessian, Burlap, and the Java message service (JMS), which are available with the Spring framework, also facilitate integration between platforms and with external organizations.

Within an enterprise, another technology that facilitates integration is an enterprise service bus (ESB). An ESB can mediate access to services and data by applications running on both NonStop systems and on other platforms. Mediation services can include protocol conversion, request/response translation, mapping or normalization, context management, routing, service scripting, and so on. Most ESBs also provide the ability to expose new services that are themselves composites of services available on one or more platforms. The use of orchestration and choreography techniques to compose new services or business processes is a powerful capability of ESBs. NonStop application services can be exposed to ESBs and NonStop applications can invoke services from ESBs, using web services technologies.

Enterprise application integration (EAI) solutions and older service bus solutions implement both proprietary technologies and standards based technologies. Some newer ESBs are themselves open source and primarily standards based. EAI and ESB solutions may have some significant benefits, but there are many costs, so a cost benefit analysis is required. Open source connectivity solutions such as web services, message based services, and related technologies may provide effective lower cost integration solutions.

Platform, development tools, and frameworks

Platform migrations

Migrating to a new Integrity NonStop platform provides a very basic type of application modernization by increasing application throughput and possibly reducing response times. New platforms also offer larger amounts of memory per CPU, greater amounts of disk storage and possibly improvements in the networking infrastructure. Legacy applications may take advantage of these changes with little or no modifications, and legacy databases may require little or no change to take advantage of a new platform. TNS applications, even those migrating from very old platforms, will likely require few (if any) changes. TNS/R native mode applications require build command changes and recompilation, but minimal code changes. Of course, there are system management and operations considerations, which must be addressed.

The H-series Application Migration Guide provides details on the changes that may be required in moving to the Integrity NonStop platform. However, there is a relatively easy way to determine the overall scope of changes required to migrate applications to an Integrity NonStop system. HP NonStop Services offer several “evolution” services to help migrate to the Integrity NonStop platform. The Evolution Assessment service identifies applications and database changes that may be required to complete a migration, but also assesses six other essential areas to address in migrating to Integrity NonStop systems. An Evolution Infrastructure Planning and Design service is also offered as a follow on option.

From a development perspective, applications that were developed for the Integrity NonStop systems should be deployable on H-series or J-series operating systems and on HP NonStop Integrity BladeSystems. If applications are going to be deployed on J-series systems, it is useful to have a J-series system available for testing. Configurations are likely to be somewhat different between H- and J-series systems, because of the greater capacity of the multi-core logical CPUs.

Migrating application programs and associated operations environments is generally straightforward. Migrating databases however, can range from a simple hardware reconfiguration to an extract and load process, with many options in between. Depending on the migration approach, various NonStop tools and utilities are available to help with the database migration. When a database migration is required, HP NonStop Services can help prepare and implement a migration plan. Once a migration plan has been prepared, an HP NonStop Solution Architect should review the plan.

Platform upgrades are also required to use new products that enable the development of new modern applications as well as the modernization of legacy applications. These products, which are only available on Integrity NonStop or later platforms, include the following:

- TS/MP 2.3 and 2.4
- Java 5 and 6
- The Spring Core
- Axis2 and Axis2/C
- JavaServer Faces
- Hibernate
- C99 standard extensions for the C language compilers
- Code coverage and code profiling utilities
- NonStop SQL/MX 2.3.3 and beyond
- MXCS (supporting the latest ODBC/MX and JDBC/MX Type 4 drivers)
- NonStop Servlets for JavaServer Pages 6.1
- iTP WebServer 6.1 and 7.0
- NonStop SOAP 4.0

Another important platform consideration is the choice of either a Guardian or an OSS application execution environment. OSS is based on POSIX standards and developers from other platforms will find the OSS environment very similar to other UNIX or Linux[®] environments. It could take more time for new NonStop developers to become productive when developing Guardian applications. Open source solutions can more readily be ported to the OSS environment and most new NonStop software products are designed for the OSS environment. Programs that have been written to run as OSS processes or modified to do so are also more portable than Guardian programs.

Finally, TNS programs are compiled into the 16-bit instruction set used by the original Tandem Computers NonStop systems. The object code that is produced by the 16-bit compilers can be translated to native mode instructions, but the objects produced are not as efficient as native mode compiler produced object code. The TNS programming languages are also based on old language dialects. Although TNS languages are fully supported on the Integrity NonStop systems, there are several reasons to consider modernizing TNS application programs. Converting the old languages to native mode C/C++ or COBOL may improve performance and maintainability. Program designs can be updated to take advantage of new APIs, new communications protocols, and possible database modernization. Current staff may still be able to explain program logic and create or improve the documentation for the TNS programs or their replacements.

Development tools

Integrated Development Environments (IDEs)

IDEs make it easier for most developers to create and maintain application programs. They include language aware editors, resource management, compile, and build tools, build project management, debugging tools, and many language specific development tools. IDEs are now widely used for application development, so the NonStop build tools and compilers have been integrated into both the Microsoft[®] Visual Studio with the Enterprise Toolkit—NonStop Edition (ETK), and into Eclipse with the NonStop Enterprise Plugins for Eclipse (EPE). These tools help improve the productivity of new NonStop developers. Many optional components and additional tools are available as plug-ins for each IDE to download and use as desired.

With the x86 cross-compilers and linker, both Eclipse with EPE and Visual Studio with ETK can be used to create C/C++, pTAL and COBOL native programs for Guardian and OSS execution environments. Eclipse, however, also has comprehensive Java and Java framework development capabilities. NetBeans although not officially supported as a NonStop development IDE, can also be used to develop Java applications.

Debugging Tools

The Integrity NonStop platform provides two modern debugging tools for TNS/E programs written in TNS/E native C/C++, pTAL, or COBOL languages for both the Guardian and the OSS environments. These tools provide two significant benefits to Integrity NonStop application developers—ease of use and common user interfaces. Visual Inspect is a client/server debugging option that provides a very easy to use GUI on your workstation. Native Inspect is based on the widely used open-source GDB debugger and the HP WDB version of GDB. Developers that are familiar with GDB or WDB will be able to debug Integrity NonStop programs immediately without having to learn a proprietary tool. Java developers can use the standard jdb debugger from their workstation.

Frameworks

There are well over a hundred frameworks available to software developers. Most of them are based on a specific language, such as Java, C++, or JavaScript and the packaging and usage varies by framework. They may be component oriented, service oriented or just a set of libraries. They may have a server side focus, client or web side focus, or fully support development of both clients and servers. The scope can also vary greatly for different frameworks, from a very broad and comprehensive application scope to a narrow focus on database or remote services access. Multiple frameworks may be used to create a solution, and many frameworks have a layered architecture that allows for choosing and plugging in other supporting frameworks as needed.

Frameworks are used because they ease application development and improve programmer productivity, while helping to maintain a consistent environment and level of quality. Frameworks are able to provide these benefits, through some set of the following features:

- Standard application programming interfaces (APIs), models, and templates for common business tasks
- Built in support for attributes such as scalability, availability, security, transaction protection, and manageability
- Support for production deployment and maintenance
- Allow programmers to focus on business logic versus routine programming concerns
- Open source developer communities which actively maintain and enhance the frameworks

Several of the most widely used open source and free Java frameworks are Spring (an enterprise application framework), Axis2 (for web services), MyFaces (web UI based on the JavaServer Faces specification) and Hibernate (Object Relational Mapping Framework for database access). The NonStop Division has announced plans to test, validate, and offer software support to facilitate the use of these Java frameworks on the Integrity NonStop platform. The Eclipse IDE also has optional plug-ins to aid in Java development with these frameworks.

The use of these frameworks can significantly ease the development of new Java applications for the NonStop platform and the applications will be more easily portable. Older Java applications may also be able to take advantage of frameworks, when they require significant enhancements.

The steps to application modernization success

The objectives of application modernization are to reduce risk, reduce costs, and to increase agility. These objectives can be achieved by well-planned and executed application modernization efforts. Those efforts have been broadly categorized into four domains: modernizing green screen application UIs, database modernization, architecture and integration, and platform, development tools and frameworks. It is especially important to understand that application modernization takes place in a larger business context, which includes business planning, IT operations and network management, and there are many paths to application modernization. Choosing a good modernization path requires an accurate understanding of the current environment, a guiding vision of a more agile and cost effective applications environment, creating and following an application modernization plan, regular assessment of progress and adapting the plan to changing circumstances. The recommended steps to application modernization are the following:

Assessment

The assessment is undertaken to get a current, accurate, and detailed view of the applications environment. The assessment should be documented and the results analyzed to identify any significant issues and risks. The issues and risks also need to be documented and prioritized. The assessment and prioritization provides a foundation for the applications modernization planning. Developing a plan or roadmap for application modernization requires an understanding of:

- Current and anticipated future business requirements that may impact NonStop applications
- Enterprise IT requirements which apply to the NonStop application environment
 - Architecture
 - Standards
 - Security
 - Compliance

- Any requests for access to NonStop applications' services or data
- HP NonStop or third party products that are nearing end of support
- The current NonStop applications development environment and current staff skills
 - Languages
 - Compilers and IDEs
 - Database Access
 - Testing
 - Debugging
 - Middleware
 - Frameworks
 - Web Technologies
- Current training requirements and anticipated new training
- The architecture and design of current NonStop applications and databases
- Any current or anticipated capacity or scaling issues
- Any issues with the production NonStop applications and databases identified by:
 - Business Management
 - Application Users
 - Operations
 - Development
- Third party products that are available to help with application modernization
- HP NonStop Evolution Services for identifying changes required for migrations, planning for changes and assisting in the modernization effort
- HP NonStop account team assistance provided directly by the account team and indirectly by the HP NonStop Advanced Technology Center for application modernization

This assessment list identifies many of the important areas that should be well understood before establishing specific objectives. Other areas of assessment may be required for different application environments.

The assessment information, the list of identified issues and risks, and their prioritization together with the objectives established in the next step, will enable the development of a modernization plan.

Establish objectives

Each NonStop customer can use the results of their assessment to choose and establish as broad or as narrow a set of objectives as they believe is appropriate. The following list provides some general areas for which specific objectives may be established. Note that objectives may be established in phases and they may need to be updated over time.

Architecture

Objectives in the area of architecture should be created to guide applications design, to facilitate reuse and the integration of different applications, and address compliance requirements. Architectural objectives may be created to address very specific areas, such as security, manageability, networking, or GUI design (for example, model-view-controller). High-level architecture approaches, including SOA and EDA do not necessarily imply the use of specific technologies, for example SOAP web services. If specific technologies are preferred as a means of implementing the overall architecture, they can be established as design or technology standards as part of the software development process.

User devices and networks

If user devices and networks require changes to enable application modernization, it is important to identify the new features and capabilities that must be provided by the user devices. Network changes are usually handled by a dedicated organization outside of the applications development organization. Nevertheless, they will need to know what network capabilities are required by the modernized applications and new user devices and how the new devices and applications are expected to be phased in to production. Separate objectives will be required for user devices and for the networks.

Database

Standards compliance and data access requirements will be the primary motivation for establishing new database objectives, such as a migration to NonStop SQL/MX. Limitations or problems with the current database may also suggest the need for some specific new database capabilities.

Software development

This area is likely to be the source of many application modernization objectives, beginning with the development and deployment platform itself. Example areas for which specific objectives could be created include:

- Migrating to the Integrity NonStop platform (single or multi-core)
- Establishing design and technology standards in support of preferred architecture
- Developing and deploying new applications as OSS processes
- Developing and deploying applications with the Java Spring framework and Hibernate
- Improving developer productivity
- Using Eclipse for C and Java application development
- Developing GUIs using the MVC web architecture with NSJSP

The results of achieving an objective should be the resolution of an issue(s), or the elimination or reduction of risk(s). The prioritization of objectives and possible phasing is guided by the prioritization established in the assessment phase. The objectives and phasing should also be documented.

Create a plan

The assessment, the prioritization of issues and risks, and the establishment of clear objectives and phasing provide a solid basis for identifying and prioritizing the required plan activities and developing the overall application modernization plan. The development of the plan requires the skills of a project manager, the knowledge of a developer or architect that is familiar with the current HP NonStop environment, and a staff that understands the new technologies and how to use the products that may be required. There will likely be some costs associated with the applications modernization program, so the project manager will need to be aware of budget considerations and the possible need to prepare a funding proposal with a cost justification and estimated return on the proposed investment.

Knowledge of new technologies and products is essential to developing an effective plan. Although there are many good books and course on the various new technologies that may be candidates to adopt, using outside experts is probably the easiest approach for the planning phase. HP Services can assist with creating a plan and they have significant expertise with all the technologies that may be under consideration. If there is no budget or a limited budget for creating the modernization plan, other “no cost” options are available. Solution architects from the HP NonStop division are good sources of information and they have access to other HP experts to assist with more complex issues or questions requiring specific experience. Third party software providers, which may have products that can help with application modernization, also have significant expertise. There are also many web resources for widely used standards based and open source technologies.

Just as there is a need for technology expertise in creating a good modernization plan, there will be a need for expertise and help in resolving issues with new architecture, technologies, and products. The plan should identify resources in advance, so the development organization knows where to go for help.

The development organization is going to require training. HP Nonstop Education has already developed courses that are directly relevant to applications modernization. They can also tailor courses to meet specific customer requirements and they can deliver training remotely. Tutorials are available on the Internet for many technologies.

Small, well bounded pilot projects or proofs of concept are a good way to gain experience with the implementation of new architectures, new design techniques, new programming languages, and new development methodologies. They also help the development organization become more comfortable with new development tools and technologies and to gain experience with newly introduced NonStop products, such as OSS, NSJSP, and NonStop SQL/MX and third party products.

Changes to existing applications and databases must be planned to minimize the impact on the production operations. Upgrades need to be implemented in phases to minimize risk and the total cost of the upgrade. The development and testing of fallback plans is required in addition to testing the upgrade plans themselves.

Execute the plan

Application modernization can be achieved by following a good plan. During execution of the plan, regular technical reviews will help avoid periods of ineffective work or low productivity and they can present an opportunity to refine and improve the plan. This is especially important for ensuring that subsequent phases in multi-phased plans are completed more smoothly.

Application Modernization will be an ongoing process, since the introduction of new technologies and products is a continuous occurrence. The HP NonStop Division will continue to enhance products and to introduce new products that enable customers to develop and deploy new applications that are scalable, highly available, maintain data integrity and deliver excellent price/performance. The introduction of new technologies, open source and standards-based products, however, increases the agility of application development organizations, enables cost containment or reduction and can reduce or eliminate information technology risks.

For more information

<http://www.hp.com/go/nonstop>

HP Partitioning Continuum, HP, 2008

HP ENSAextended technical overview, HP, 2008

HP Utility Data Center Overview, HP, 2008

Call to action

<http://www.hp.com/go/nonstop>

HP Partitioning Continuum, HP, 2008

HP ENSAextended technical overview, HP, 2008

HP Utility Data Center Overview, HP, 2008

Technology for better business outcomes

© Copyright 2009 Hewlett-Packard Development Company, L.P. The information contained herein is subject to change without notice. The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

Linux is a U.S. registered trademark of Linus Torvalds. Microsoft and Windows are U.S. registered trademarks of Microsoft Corporation. UNIX is a registered trademark of The Open Group.

4AA2-9050ENW, September 2009



Get connected

www.hp.com/go/getconnected

Current HP drivers, support & security alerts
delivered directly to your desktop

