Compliments of comforte®

comForte Special Edition

HP NonStop™ FOR DUMMIES®

A Wiley Brand

Learn:

- More about HP NonStop systems
- Why the platform is great in the areas of high availability and linear scalability
- How to port, develop, or modernize applications
- Ways to integrate the platform into the enterprise

Werner Alexi
Thomas Burg
Bill Sempf

A Note from comForte

The core team of comForte is a bunch of people who have known each other and have worked on the HP NonStop platform for decades, more than 30 years for some. We make our living writing and selling software for this computing platform, which might make us just a little biased – please see www.comforte.com or below on this page for more about the company and its offerings.

We constantly talk to people dealing with HP NonStop in various ways: we meet people new to the platform, we hire people fresh from university and get them to discover the platform, we talk with people who are NonStop veterans, just like us.

Keeping up with technology trends is hard in itself – dealing with a platform that has a legacy of nearly 40 years might be even more of a challenge. Of course there is a wide range of public information available on HP NonStop — but to the best of our knowledge there is no platform primer available prior to this book.

This book is an attempt at presenting a concise, short overview of the HP NonStop platform.

This book was a team effort. Huge thanks go to Werner Alexi who contributed tremendously as an author to this book, to the Wiley team, and to the unnamed early reviewers inside and outside of comForte.

Please provide any feedback via www.comforte.com/ns4d/feedback

Germany, August 2014
The comForte author team

About comForte

comForte provides proven and innovative middleware, connectivity, and security solutions for users of HP NonStop systems. comForte cares about its customers and the HP NonStop platform, offering an unrivaled and unmatched portfolio of software products in the NonStop industry. With customer value in mind, it is comForte's goal to deliver best-in-class products and solutions and to provide customers with the best support possible.

HP NonStop

FOR DUMMIES®

A Wiley Brand

comForte Special Edition

**by Werner Alexi,
Thomas Burg, and
Bill Sempf**

FOR DUMMIES®

A Wiley Brand

HP NonStop For Dummies®, comForte Special Edition

Published by
John Wiley & Sons, Inc.
111 River St.
Hoboken, NJ 07030-5774
www.wiley.com

ISBN 978-1-118-84993-4 (pbk); ISBN 978-1-118-85012-1 (ebk)

Manufactured in the United States of America

10 9 8 7 6 5 4 3 2 1

Publisher's Acknowledgments

Some of the people who helped bring this book to market include the following:

Project Editor: Jennifer Bingham

Acquisitions Editor: Katie Mohr

Editorial Manager: Rev Mengle

Business Development Representative: Sue Blessing

Project Coordinator: Melissa Cossell

Table of Contents

Introduction

Sixteen seconds after an outage of a central computer affecting ATM availability, the event was reported on Twitter. Minutes later, a storm of Twitter messages appeared. And a short time after that, the bank's CEO got a call from one of the country's leading newspapers.

Due to a computer outage lasting only a few hours, a major company in the financial industry lost an estimated amount of US$90 million.

Are events like that inevitable like thunderstorms or earthquakes? Insiders know that they're not: The HP NonStop Server can host applications that run for months or years — without any interruption and without losing a single transaction.

About This Book

This book is for anyone with an interest in HP NonStop systems. This book is for you if you're new to the platform, have spent many years working with it, if you manage people or IT systems, or if you're a system operator or developer on/for the HP NonStop platform.

This is obviously a short book. There isn't a lot of in-depth technical information here. This book will help formulate a strategy for best making use of HP NonStop in a contemporary enterprise environment.

It will not help you design an effective, fault-tolerant nonstop process pair. But it will describe what one is.

This book was written with and for comForte.

How This Book Is Organized

This book is structured into the following chapters:

- **Chapter 1: What is HP NonStop?** Get started with HP NonStop. Learn what it does well and how it can help you.
- **Chapter 2: A Platform Overview:** This chapter explains what makes HP NonStop special and how HP NonStop computing is baked right into the hardware and software layers.
- **Chapter 3: Developing Software for HP NonStop:** Nice server you got there. Who's going to code for it? You will find those answers here.
- **Chapter 4: HP NonStop as Part of Enterprise IT:** Whether you've been running the same application for 20 years or just started afresh, you will want to get HP NonStop playing nicely with all your other servers. Also, you want to make sure the platform is secured properly.
- **Chapter 5: Ten Resources for More Information:** We couldn't fit everything we wanted into this book, so comForte has created some additional content. This chapter includes links to that content.

Icons Used in This Book

Sometimes, important details are called out using icons. These are cool little graphics on the left margin of the page. Keep an eye out for them!

Tips are particularly interesting items in the text that offer you some practical information you can use.

The Remember icon points you to things that are worth remembering.

Beyond the Book

For more information on HP NonStop and comForte, go to `www.comforte.com`.

Chapter 1

What Is HP NonStop?

In This Chapter

- Examining real-world uses for reliable computing
- Looking at the components of HP NonStop

HP NonStop is a set of hardware and software that has been designed and built to survive the loss of components that most other systems would consider fatal. It is self-healing, redundant, and reconfigures itself in case of failure. HP NonStop is used by organizations that need computers that never stop working — thus the name.

Why the World Needs HP NonStop

Wherever you see a system that simply can't be allowed to fail — under any circumstances — you will probably find an HP NonStop server running the show.

Telecommunications, automated teller machines (ATMs), emergency phone services, point of sale (POS) systems, and manufacturing facilities all depend on HP NonStop to reduce global downtime to just a few — or even zero — minutes per year.

The increase of data

The Internet has created a dramatic increase in the amount of data being processed and stored. Add new or newish trends such as Big Data and the Internet of Things and you end up with even more data.

Some data has the two additional characteristics

- Must be available all the time
- Must be consistent at any point in time

Operations on that type of data are called *transactions.* For instance, it's not okay for your ATM to take the money from your checking account and not give it to you. Either the system performs all the steps required to complete the transaction or none of them. Similarly, it isn't okay to tell someone that the police are on the way to the scene of the accident and lose the message that dispatches the police. Money (lots of!), lives, and reputations can be lost by a single message (transaction) getting lost.

Transactions are the lifeblood of corporations.

Fault tolerance and high availability explained

Fault tolerance is the capability of a system to continue to function in a predictable fashion as components fail. HP NonStop systems are designed to be fault tolerant; therefore, no single point of failure will impact the system and, in fact, even many components failing will most likely not impact another one.

Systems that can't be reached can't be used — being unavailable means losing money. *High availability,* as defined by International Data Corporation (which defines such things), means an application is available even in the face of failure in part of the system. High availability means avoiding *any* outages, whether planned or unplanned. Systems are rated from Availability Level 1 to 4. HP NonStop systems use a combination of hardware and software to achieve the highest level of availability (technically termed Continuous Availability or Availability Level 4). Level 4 is often described as *five 9s or higher,* meaning an uptime of 99.999 percent or more, which translates into a total downtime of less than five minutes per year at *five 9s* or about 30 seconds at *six 9s.*

Data integrity, reliability, and RAS

Data integrity is the quality of maintaining data correctly, despite flaws in surrounding systems. HP NonStop is designed from the ground up for data integrity — which is a precondition for *reliability.* A reliable system will not return a result rather than returning a wrong result or corrupting a database.

If a system doesn't have three key properties: *reliability, availability,* and *serviceability* (RAS), it can't be depended on to perform mission critical operations.

The original designers of HP NonStop built the platform for fault tolerance, high availability, data integrity, and RAS. The platform still offers those today and into the years ahead. For more on the original design of HP NonStop, see the nearby sidebar.

A history of HP NonStop

In 1974, Tandem Computing entered the large-scale computing market with the revolutionary idea that computer services should be predictable and dependable. Computing was very different in those days: Computers were powerful, but unreliable.

James Treybig, an HP engineer, assembled some of the best and brightest engineers and designed a set of hardware and software that provided a fault-tolerant online transactional processing (OLTP) environment for business.

The first Tandem/16 system rolled off the line and was installed at CitiBank in 1976, beginning a nearly 40-year relationship between the banking industry and fault-tolerant computing. In 1997, Tandem was acquired by Compaq along with competitor DEC. In 2002, Compaq and HP merged, bringing Tandem back under the company where it got started.

Although the core concepts of HP NonStop have remained stable, the HP NonStop system of today has very little in common with the very first Tandem system of the late 1970s — except the almost unmatched availability.

Why HP NonStop Runs Nonstop

Today, a wide range of hardware platforms run an even wider range of operating systems. And with few notable exceptions, the vendors of the hardware and software pieces aren't the same. HP NonStop is a tightly integrated suite of hardware and software that gathers much of its strength from this exact tight integration.

Fault-tolerant hardware

Quality servers in the Internet space often come with two power supplies and a RAID array. *RAID,* or *redundant array of independent disks,* is a simple and effective way to provide basic fault tolerance to storage.

But even for standard applications, this level of fault tolerance often isn't sufficient, so computer vendors invented *clusters.* A cluster is a tightly coupled set of multiple systems sharing certain resources and protecting against certain failures. Clusters aren't typically easy to set up and manage, especially when data has to be kept synchronous between the individual systems.

HP NonStop hardware takes redundancy to the logical extreme, providing effectively 100 percent uptime. Every hardware component in an HP NonStop system is duplicated. At least two storage units, two RAM busses, and two processors reside in every HP NonStop system. But that's only part of the story.

Fault-tolerant software

From an eagle-eye perspective, a single HP NonStop system can be described as a self-managing, self-healing *cluster in a box.*

This works by having a rather singular operating system concept, combined with hardware redundancy: Each CPU runs independently of each other in a *shared nothing* environment, preventing a sick or malicious program from hurting the system. Critical operating system processes run as what is called *process pairs* spanned over two CPUs — this allows processes to survive failures of the CPU they run in.

Through this design, the software is capable of transparently failing over to a second set of components without interruption of the currently executed operations.

Chapter 2

A Platform Overview

In This Chapter

- Explaining core concepts
- Examining important components of HP NonStop systems
- Looking at benefits of HP NonStop systems

A platform can claim five or six 9s of availability in theory (see Chapter 1 for the definition of five 9s), but backing it up in reality is more difficult. The question is: What about the HP NonStop platform makes it so reliable? This chapter goes into the answer in some depth.

Some parts of this chapter get rather technical for a *For Dummies* book, so if you don't need to understand every piece of the magic that makes HP NonStop systems run nonstop, feel free to gloss over parts of this chapter. We do recommend to at least read the parts about databases, TMF, and Pathway. We also think the section "Key Benefits of HP NonStop Systems" is valuable.

Some Core Concepts

At the core of the functionality of HP NonStop is a *message-based architecture.* This architecture surrounds the two core concepts of the HP NonStop operating system: *processes* and *messages.* The following sections discuss these concepts and also explain why they're so great.

Processes within HP NonStop

All systems have *processes* — they're the basis for contemporary computing. A process is any program running on the system. There are application processes and system

processes: They perform different functions but they behave the same.

For example, if you have a Windows machine in front of you, Task Manager will display the process list for you. Some of those are applications you're currently running. But most are processes that are running without you ever seeing them.

Are you running Linux or Macintosh? Open a command prompt and type `ps -A`. You'll see a similar list of processes — some of those will be apps you might be running right now, but most of them are processes running under the covers.

Of course, HP NonStop has the same capability to list processes, and there are multiple ways to do it. From the command shell, typing **STATUS *** will list all processes (for more on command-line syntax, see Chapter 5).

Figure 2-1 shows a Windows Explorer style view of the processes running on HP NonStop, sorted by the column CPU busy and showing the busiest processes first. Everything in the system is implemented as a process: network connectivity, disks, command shells, transaction monitoring, databases, the kernel itself — *everything*.

Processes running in the background doing important work exist on other systems as well (called *services* in Windows or *demons* in Unix). However, there are two important differences in the way processes work on HP NonStop:

- All critical operating system processes run as process pairs spanned over two CPUs. Any process that encounters a fault is automatically switched to its backup instance. When the backup instance has taken over, after only a few milliseconds it will then become the primary (and create a new backup instance). Process pairs implement all central operating system services. They provide access to the network, storage, and so on. A single failure of any of these services will leave the system as a whole intact.
- Many processes on HP NonStop have well-known names (for example, $DATA1). Named processes can exist concurrently with the same name on two different CPUs. So even if one CPU fails, the process $DATA1 will still be there and usable by the application. *Usage* of a process means exchanging messages with it.

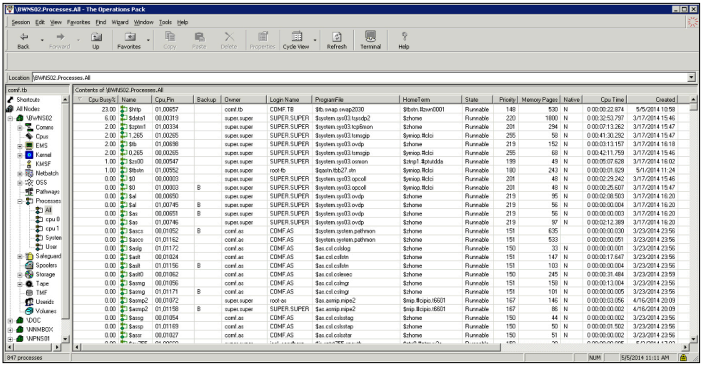

Figure 2-1: Looking at processes using the comForte TOP product.

You have mail: Messages

Every operating system is based on the principal of delegation. If you can't do something that needs to be done, find someone who can and hand the task over. The basic agents within systems are processes.

The main difference between HP NonStop and other systems is the way delegation is implemented. If process A wants process B to do something, A puts together a *letter* describing what is to be done (a message) and sends it to B. B does its work and sends a reply back to A. The method to send messages from A to B is part of the operating system itself. Other types of operating systems use methods for delegation requiring a much tighter coupling of A and B (like sharing common memory).

What is so special about using messages?

- **Location independence:** A message from A to B can travel within the same CPU, to a different CPU, or to a different node in the HP NonStop network, possibly halfway around the globe. Neither A nor B has to know or care. If the CPU B used to work on becomes too busy, put a new one into your system and move B to this new CPU. Problem solved.
- **Fault tolerance when using process pairs:** Technically, B can consist of two instances: B1 and B2 running on different CPUs (a *process pair*). A message from A to B will only go to one instance — say B1. If B1 (or the CPU where B1 runs) fails, the operating system can automatically route the message to B2. B2 does the work and replies. A doesn't even notice that there was a fault.
- **Isolation:** If B becomes internally corrupted by a defect (every program has defects), B can't corrupt A. The reply to A's message might be missing or empty or corrupted, but A will be able to find out and stay intact. This greatly improves the overall robustness and security of the system.
- **Performance manageability:** HP NonStop administrators know their processes and what they're good for. By measuring resource consumption on a process basis, they can easily identify performance problems.

There are a lot more advanced advantages of messages (like easy interceptability or transaction inheritance).

How processes and messages work together

Every process has its own mailbox, called $RECEIVE, which is used to read and reply to messages arriving there. The mailbox maintains a queue of messages, so that the serving process can work on them one by one. This applies to application programs as well as services provided by the operating system.

If an application process wants to write something to disk, it will call a WRITE API function just like on any other operating system. However, on HP NonStop, the kernel will eventually translate that API call into a message landing in the $RECEIVE queue of the proper disk process. The disk process will only mark this message as completed (by a specific REPLY message) after the data is safely cached or actually written to disk.

The message-based processing and complete redundancy complete the picture of a truly fault-tolerant system. If a CPU fails, or a disk drive, a disk controller, or a network card fails, HP NonStop systems keep working, moving their messages to other processes and the work continues.

In Figure 2-1, you can see process $HTTP, which is a simple web server busily serving static pages and writing log messages to disk. It has been configured to write a huge amount of log messages in order to create high disk drive activity. Look at $data1. This is the process associated to the disk drive where the log files are being written to. The next process down the list, called $ZPTM1, is related to the TCP/IP network stack, which is also implemented as a process, interfacing the process $http through message passing.

A self-managing and self-healing cluster in a box

It might seem a lot of effort and maybe even wasteful to implement a web server by passing messages around between the actual web server, the disk (process!), and the network (process!) rather than by moving pointers to buffers through the kernel.

However, it is exactly this design that gives HP NonStop systems their unique capabilities such as extreme reliability and scalability — all baked right into the operating system.

All critical processes on HP NonStop are made self-healing by running as process pairs. Also, all processes can and do run in different CPUs, even in different systems that are physically located far apart that communicate via messages.

When you start developing your own application on HP NonStop, you don't have to worry about process pairs and the message system — the development framework will take care of this for you. See Chapter 3 for more information on this.

Layers or Personalities of the Operating System

HP NonStop systems are powerful beasts that have evolved over decades. That resulted in a complex operating system that has several pieces best understood as layers or *personalities.* See Figure 2-2.

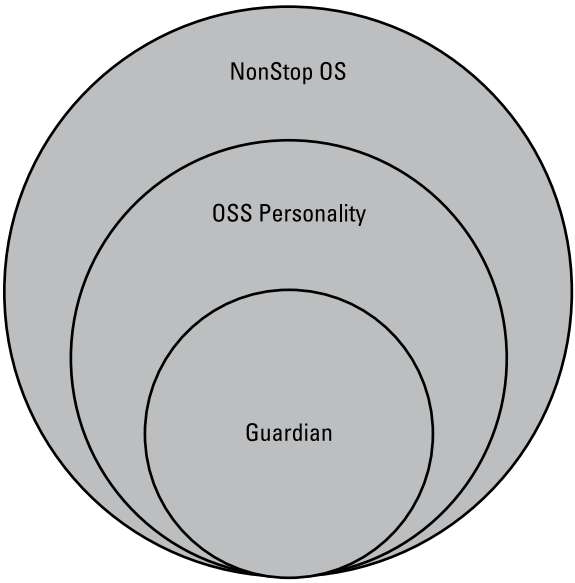

Figure 2-2: Layers of the HP NonStop operating system.

NonStop OS is the name of the operating system

As of today, the name of the unique operating system that runs on HP NonStop systems is *NonStop OS*. It consists of several components working together.

OSS is a Unix-like personality

From an operator or developer perspective, contemporary HP NonStop systems can look a lot like Linux or Unix: The so-called Open System Services (OSS) layer or OSS *personality* provides a Unix-like interface to the underlying operating system kernel. It provides nearly everything that Unix users are used to having at their fingertips. It has a hierarchal file system with long and case-sensitive filenames and directories. OSS also provides most of the Unix utilities, like bash shell, grep, awk, sed, and many of the programming functions.

It works with the NonStop OS and makes use of many of its services like fault tolerance and parallel processing. However, remember that OSS isn't an operating system, and it doesn't replace portions of the HP NonStop OS.

In fact, OSS is so much like Unix that many of the open source tools and applications that are available in the Unix world are available for OSS. See Chapter 5 for details.

A core understanding of *Guardian* (see the next section), the underlying operating system, is still helpful (especially for system architects) because OSS is built on top of it. Every OSS process is a Guardian process and every OSS file is a Guardian file (and vice versa). There are mapping tables that provide the long filenames and sensible process identifiers, but under the covers, the Guardian system is still running the show.

OSS is so powerful that it's possible to start and end many development projects completely within the environment. The Guardian principles are still there, with processes passing queued messages to other processes' $RECEIVE inbox. The development environment, language, and tools, however, look exactly like Unix.

Guardian is the lower layer

Guardian was the original name of the HP NonStop's OS. Today, the term is used to describe an environment that includes:

- The operating system kernel itself
- The set of APIs specific to the platform, including process control, access to files
- Tools and utilities that operate on Guardian objects — files, processes, devices, and so on
- Services that run in the Guardian environment

Guardian includes a number of subsystems that are made up of several communicating processes (no surprise there), each with its own toolset for management and configuration. To mention just a few:

- The **spooler** maintains printers and jobs to be printed. There are HP NonStop configurations that support thousands of printers in nodes all over the globe, and keeping them all working is the spooler's job.
- The **Subsystem Control Facility,** or **SCF,** helps to manage several subsystems in a persistent fashion. SCF manages the network, disks, and even the kernel itself.
- The **Event Management System (EMS)** collects and distributes failure, exception, and information events in a central place, and distributes them as needed.
- **MEASURE** provides performance metrics on a very detailed level. It is an important part of the performance analysis and tuning of an HP NonStop system.

If you find maintenance of the various subsystems in Guardian a chore, comForte (and other companies) have products that give a nice graphical user interface for the whole HP NonStop system.

Working with Files in the Guardian Environment

Files in the OSS environment are pretty much like files on any Unix/Linux system. However, files residing in the Guardian environment differ massively in that there are no directories or extensions. This lack of structure can be somewhat of a shock for folks coming from the Windows or Unix world.

At first glance, the Guardian file system seems very limited and cumbersome. With a little use and using Table 2-1 for reference, it becomes easy to use and organize.

Filenames in Guardian aren't case sensitive, and only contain letters and digits, with a few notable exceptions. Table 2-1 shows how a filename is broken into four parts.

Table 2-1 Parts of a Filename

Part	*Description*	*Example*
Part 1 (optional)	Name of the node in a network of HP NonStop systems. This always starts with a backslash (\).	\NS1
Part 2 (optional)	Disk name. Starts with a dollar sign ($), followed by a letter and can only contain letters and numbers.	$DATA01
Part 3 (optional)	Subvolume name. This is a lot like a directory. It has to start with a letter and can only contain letters and digits.	DOCS
Part 4 (required)	Filename. This can be 8 characters. It has to start with a letter and can only contain letters and digits.	ABOOK

Assuming proper defaults are set (somewhat similar to the `cd` command in Windows or Unix), the exact same file from Table 2-1 can be referenced as `ABOOK` or `DOCS.ABOOK` or `$DATA01.DOCS.ABOOK` or, finally, `\NS1.$DATA01.DOCS.ABOOK`. The last version is the so-called fully qualified filename

and it is unique across a whole network of HP NonStop systems. That's all you get — no extension, no directories.

It should be noted that Part 2 of Table 2-1 can refer to a physical disk or a virtual disk.

Putting the Data in a Base

Transactional databases are one natural endpoint of all the fancy concepts of message-based architecture. Although transactions are supported in IBM's DB2, Oracle, Microsoft's SQL Server, MySQL, and most other databases, the architecture of HP NonStop provides a particular feature that none of the others have: The transactions and the database are managed by the operating system.

Understanding how to properly process transactions is important to understanding the value of HP NonStop. A transaction is a bracket. This bracket allows treating a sequence of database operations as one single operation from an application point of view. The standard example of a transaction is moving money from Alice's account to Bob's. This transaction must have the following properties:

- **Atomicity:** It must happen completely or not at all — under no circumstances may the money disappear from Alice's account without appearing on Bob's.
- **Consistency:** At the end of the transaction, the total amount on Bob's and Alice's accounts must be the same — no money appeared or disappeared.
- **Isolation:** While the money is moving from Alice to Bob, other transactions on other accounts may be going on without interfering with the activity on Alice's and Bob's account.
- **Durability:** After the end of the transaction, the new accounts must be safely stored, so that even switching the systems off will not undo the action.

The first letters of the words in brackets give the acronym ACID. ACID is needed whenever transactions carry really valuable data (like money). ACID is built into the HP NonStop operation system.

TMF: Keeping track of transactions

Transaction Monitoring Facility (TMF) is the subsystem of Guardian that is responsible for managing transactions. Because TMF is part of the operating system, it provides certain features that are rarely found elsewhere (if at all).

For each transaction — as defined by the application — TMF creates a unique identifier, the *transid*.

- The transid can be shared among multiple processes communicating via the message system to execute a single transaction. $A starts a transaction, and $B takes over, using the transid inherited with the message in the $RECEIVE inbox. This concept allows for some remarkable modular application design that allows distribution of work to several processes.
- Processes can switch transaction context within the work cycle, further increasing flexibility in application design.
- TMF provides a common audit trail, providing the basis for products that allow the near-real-time replication of database operations to other locations while the system is running. Also, the audit trail provides a single, atomic view of all database changes, which allows analyzing them or rolling back individual changes.
- In the case of a total system failure, a consistent state of the database can be recovered by TMF in a reasonably short time frame.

This core functionality is baked right into the HP NonStop OS, but the overlying database management system has changed with the times.

Currently there are three database systems available under HP NonStop OS, becoming more and more sophisticated.

Enscribe is the original database

When Enscribe was first written, SQL databases didn't exist other than as a concept. Enscribe works on Guardian files. Most Enscribe files are sets of structured records. The record structure is described in a data dictionary, and defined using a custom Data Definition Language (DDL). It works so well, it still comes as part of every HP NonStop system. Enscribe's creators clearly were doing something right.

Enscribe uses four file types:

- **Key Sequenced Files** contain variable length records found using a key. Keys are data — like customer number or last name — even combinations of fields. Key Sequenced Files support insert, update, and delete operations. They are optionally extended by so-called Alternate Key Files, which allow fast searches against multiple different keys in a single file.
- **Relative Files** consist of fixed-length records that are indexed by a record number. The disk location of the record can be computed from the record number so no lookup is required to retrieve data. Inserts, updates, and deletes are supported. Inserts have to be at the end of the file or can reclaim space from deleted records.
- **Entry Sequenced Files** are variable length records that are stored in sequence. Deletes are rare and usually cause unclaimed space. Updates can be made only if the data length is the same and are rare as well. Entry Sequenced Files are often used for log files and such.
- **Unstructured Files** are long strings of bytes, just like a Unix or Windows file. It is totally up to the application how the file is searched or updated. No deletes are available, because there are no structured records.

That's pretty much it. Enscribe doesn't provide the features you might be used to in terms of joins and views and triggers. Enscribe might be just powerful enough for your purpose, and it is fast and totally reliable.

SQL/MP brought structured queries

SQL/MP was the first true relational database system on the platform. It is still the workhorse of many contemporary HP NonStop applications. The MP in the name stands for *massively parallel* and it isn't kidding. The principles of the message-based architecture are applied to the database management system in such a way that transactional files and processes are distributed over multiple disks, multiple processors, or even multiple systems.

There are still, however, pieces missing from SQL/MP that would make it a truly relational database system. For instance, relational integrity isn't enforced at the data layer — it's the application's problem. Also, there are no triggers or stored procedures.

All of that said, SQL/MP provides linear scalability so you get more throughput in a truly linear fashion when you add CPUs or whole new systems and make them available to your application.

SQL/MX is the gold standard

SQL/MX is a modern relational database system that was designed from the ground up to be a reliable, scalable, and capable database for HP NonStop, making full use of the OSS personality. Among the three databases, SQL/MX is the product that receives the most attention from HP when it comes to adding new features.

Here are a few of the features of SQL/MX that make it the gold standard:

- In general, SQL/MX provides a highly increased compatibility to the ANSI SQL standard when compared to SQL/MP. Examples are the capability to use incredibly complex queries or more *orthogonality* in the query language.
- The record size in SQL/MX is 32K versus 4K for SQL/MP.
- Stored procedures are available.
- Referential integrity is built into the database.

- For ODBC connections, precompiled SQL is kept in cache to prevent costly rebuilds.
- You can use SQL/MX from Java, due to fault-tolerant JDBC drivers.
- There is a publish and subscribe mechanism.
- There are ANSI standard column and table names.

Pathway – the Transaction Monitor

HP NonStop systems have always been famous for their linear scalability. By adding more hardware resources to the system, you get more transactions per second — in a linear way.

This isn't true for most other architectures. Typical Unix or Windows systems will scale to a certain point in a transactional environment. But from then on, putting more CPUs into the systems or more systems into the cluster won't get you more throughput as you add more hardware; the increase in throughput will eventually decrease until it hits a final boundary.

Applications making full use of the scalability can be based on the resource monitor Pathway, which is available from HP and tightly coupled with the OS. Pathway fully leverages the core concepts explained earlier in this chapter and provides a framework to build reliable and scalable applications.

In the original Tandem computers, Pathway would sit in between the green screen apps written in Screen COBOL and the HP NonStop processes, determining which process was the most capable to take on the task requested by the user. Many of these applications are still in use today, decades later, and work very well.

Today's user expects a fancy user interface, or even web accessible tools. Fortunately, the basic Pathway concept still works today — better than ever. That is because from very early on, Pathway provided a clear architectural separation between a *client* requesting units of work and a *server* delivering them. The most important concept is that of a *serverclass.* The term *serverclass* simply means a

name for a set of processes, which all can deliver the same service. Each individual process is called an *instance* of the serverclass. A client program — whether it executes locally on HP NonStop or on a midtier application server or on a PC client — can send a message to a serverclass. Pathway will find out the best instance, send the message to that instance, and reroute the response to the client or, as it was named predating the terminology of client-server, to the requester.

For a better understanding of the concept consider an analogy. Imagine a check-in hall at an airport. Each airline can be compared to a serverclass. So you have a serverclass Delta and a serverclass Lufthansa. If you have a certain application task to do, like "Check in for Lufthansa flight LH123," you find the queue for the Lufthansa counters. Fortunately, there are five agents servicing your requests. These five agents all do the same job — check in passengers for Lufthansa flights. So, the agents can be compared to the instances of the serverclass.

To take that a bit further, imagining that a big flight comes in and suddenly the application has to check in 500 passengers all in a hurry. Scalability can simply be achieved by increasing the number of agents. This is how massively parallel processing is achieved.

In the analogy, Pathway would automatically add more agents (serverclass instances) to the Lufthansa service when the load increases. The serverclass instances would be naturally distributed across the CPUs of the system or even across multiple systems.

Running Two HP NonStop Systems in Active-Active Mode

Working together, TMF and Pathway give you the foundations for very high uptime. However, if your application is running on a *single* HP NonStop system, you will have to plan for downtimes of 15 minutes or more at least every few years: After all, even an HP NonStop system needs to be shut down if you need an operating system upgrade, you're switching to completely new hardware, or you're moving your data center.

The solution for avoiding any downtime in those cases comes in the form of a so-called active-active system: You build a cluster of at least two nodes in the following fashion:

- Both nodes always process transactions, normally sharing the load.
- Both nodes carry a full local copy of the same database, which are kept in sync. The synchronization is done using replication software, typically available from third parties.
- If one of the two nodes fails, the other takes up processing of all transactions.

Active-active isn't the only way to achieve high levels of availability in case of planned downtimes for operating system upgrades, hardware replacements, or the like. *Hot standby,* where a ready-to-go hardware set is kept up-to-date just in case, is often just as good or even better for some implementations. It is certainly easier to set up than true active-active.

An active-active configuration for an application is difficult to build — but it can be and has been done on HP NonStop for decades. Beyond uninterrupted uptime for up to decades (no kidding!), it also gives you a perfect business continuity solution because the two nodes can be very far apart from each other — for instance, one at the west coast and one at the east coast of the U.S.

Key Benefits of HP NonStop Systems

Here are some of the key benefits of HP NonStop.

Common standards, uncommon advantages

The open standards supported by HP NonStop are key to the ease of use and integration of the system. The support of standard technologies such as Java, C++, JSON/REST, SOAP, SASH, and others — all running in a self-managing and self-healing cluster in a box — combine to give enterprises that depend on HP NonStop uncommon advantages.

What uncommon advantages? The most distinguishing features of HP NonStop are RAS (reliability, availability, and serviceability; see Chapter 1) together with its linear scalability (see earlier in this chapter).

The exceptional serviceability of HP NonStop means that with your application running and actively processing transactions, you can replace just about any piece of hardware in the system. This goes far beyond being able to replace a single disk in a RAID array. Also, you can update several layers of software (both at system and application level) without having to take the application down.

In the financial world, it is totally normal for HP NonStop systems to process several thousand transactions per second — for hours and days and years on end, without missing a single transaction. Need to process even more transactions? Thanks to the linear scalability, you can simply add more hardware capacity without needing to worry about hitting any system or architectural limits.

Low TCO (total cost of ownership)

Comparing the smallest NonStop system with an industry standard server cluster running Linux or Windows and a commercial database initially often makes HP NonStop look expensive.

However, add database clustering capability, extended fault tolerance, and top-notch support to the standard server and the picture quickly looks different.

Also factoring in the cost of managing the systems, the low TCO of HP NonStop systems has been proven over and over again. See `www.comforte.com/ns4d/tco_study.pdf` for a white paper containing a detailed TCO study.

Excellent support

HP offers excellent customer support. By calling the HP global support center and declaring a case as severe, you will get a knowledgeable person talking to you in a few hours. This person will not stop working on your issue until it is resolved at least via a work-around.

The infrastructure for this has been built up over decades and works like a charm to this day.

Cloud/hybrid computing

An HP NonStop system can be viewed as a cloud within a single cabinet extendable to other cabinets and into commercial clouds like Amazon, Microsoft, and Rackspace.

One good example of HP NonStop running in a cloud computing context is an implementation where HP NonStop serves as the database backend for a web application provisioned in a private or public cloud. This gives you the benefit of combining a cheap, scalable, and stateless front end (in the cloud) with a reliable backend that is always there to keep state of transactions.

Architectures like this are also called hybrid because multiple pieces (also called tiers) of the application run on multiple, different platforms. Mixing platforms like this allows each platform to do what it does best and to also optimize for cost.

Hybrid architectures like this have been running, fully in production, for various HP NonStop customers worldwide before the term *cloud computing* was even coined.

HP NonStop on X86

Meg Whitman, the chair of Hewlett Packard, has announced that the HP NonStop platform will be modified to run on the X86 chipset without compromising any of the attributes that HP NonStop stands for. When running on X86, NonStop systems will run on commodity HP server hardware, meaning that the same hardware could, in theory, be running either Windows, Linux, or HP NonStop.

Details for time frame, features, and price point aren't available yet. But HP NonStop will continue to play an important role in HP's strategy for mission-critical computing.

Chapter 3

Developing Software for HP NonStop

In This Chapter

- Using the tools you know to build HP NonStop applications
- Modernizing applications
- Migrating applications

No matter whether you're building new applications to run on HP NonStop, maintaining or improving existing ones, or migrating applications to the platform, you'll find that the development life cycle isn't much different from what most developers are used to. Today's languages and debuggers can be used, cross-compilers can make testing and deployment simple, and developers who never thought they could build for what they consider mainframe-type or legacy systems will do just fine.

Developing for HP NonStop Doesn't Mean Developing on HP NonStop

Once upon a time when Tandem was young, Tandem's engineers invented a new programming language called TAL, and for some time, TAL was the only language available for this platform.

Those days are long gone. Although TAL is still available if someone really needs it — typically for maintaining very old applications — no new development for HP NonStop needs to be done in TAL.

Today, all standard development languages, such as C, C++, COBOL, Java, Python, and Perl are available on and for the platform. We look at some in more detail in this chapter.

Programming languages and development environments

When developing for Windows computers, developers work on Windows computers. The same is true of Macintosh and Linux. But how about Android or iOS? No one develops on a phone, right? They all use a common IDE like Visual Studio or Eclipse, and a cross-compiler that builds, links, and deploys to the phone.

The same is true of HP NonStop. Compilers and linkers are available for TAL, C, C++, and COBOL that compile, link, and generate the objects needed to run the application on the big iron. All of these have support for embedded and dynamic SQL for SQL/MP and SQL/MX databases. The developer's IDE of choice (and often libraries of choice) can be used to develop, cross-compile, and debug applications.

In a nutshell, there are two options to create an executable for HP NonStop:

- Invoke the compiler right on HP NonStop. This has been the only option for decades and it stays a viable option.
- Use cross compilers on a PC. For a long time, that meant using a plug-in called ETK from HP for the Microsoft IDE. HP is currently transitioning its cross-compilers to run under the Eclipse IDE rather than under the Microsoft IDE.

Using Eclipse to develop for HP NonStop

The front-runner of Enterprise IDEs is Eclipse (see Figure 3-1 for a screenshot of a C++ project for HP NonStop being worked on in Eclipse), and the popular environment is about to replace

the Microsoft IDE as IDE of choice for development for HP NonStop. HP has a plug-in for Eclipse called NSDEE that will help the newer generation of developers feel right at home developing applications for HP NonStop.

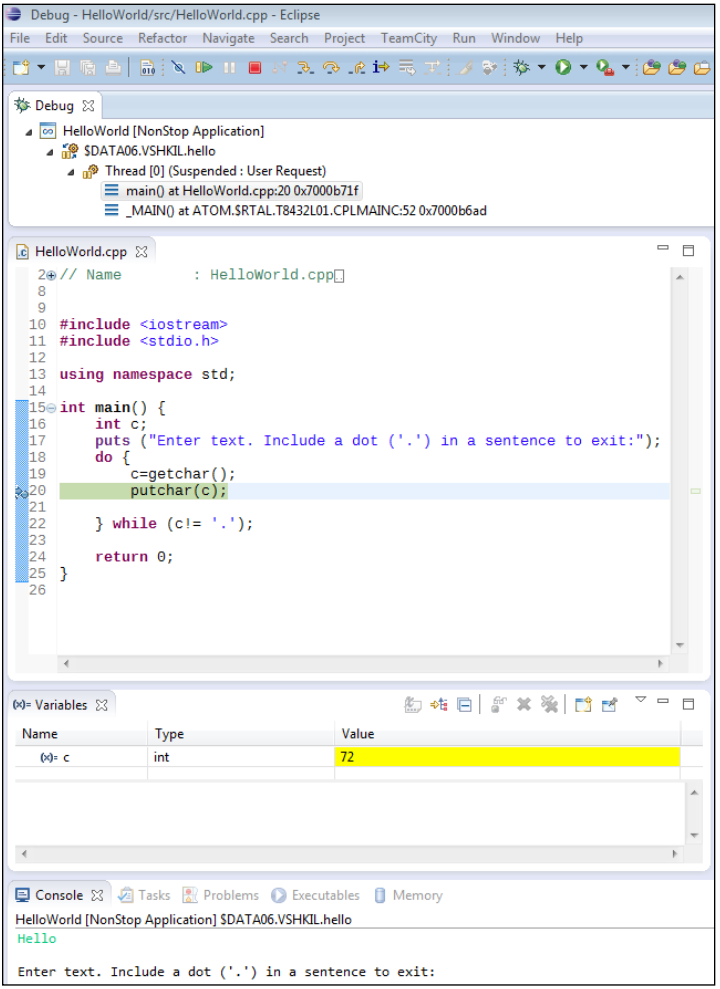

Figure 3-1: HP NonStop development environment for Eclipse.

Enter Java

The Java programming language has been supported on the HP NonStop platform for quite some time. Experienced Java developers might question the production speed of Java in a fault-tolerant and message-based environment like HP NonStop. Java has been clocked as fast on HP NonStop as it is on Unix, but your individual mileage will vary depending on many factors, such as how well your Java application follows design principles.

HP offers the SASH stack — consisting of Spring, Axis2, MyFaces, and Hibernate — for the platform. This offering is no different than what is found on any Unix box. If you have an application that doesn't require too many concurrent transactions, spends most of its time in complicated SQL queries, and needs sophisticated user interaction, Java using the SASH stack on HP NonStop may be absolutely the right choice for you.

Also, the product called HP NonStop Application Server for Java, or NSASJ, consists of a special version of JBoss, internally built with Pathway. There are JARs available that make integration of NSASJ into your application simple and reliable. This dramatically increases manageability of projects written in Java, and also shows HP's commitment to Java as a go-forward development language for HP NonStop. While a standard JBoss implementation on Unix/Linux uses a low-level interface called *sockets* to communicate between processes, the HP engineers replaced that layer with the ultrareliable and fast Pathway interfaces.

By doing this, they automatically also got TMF integration. This is invisible to the developer and makes your standard JBoss application take full advantage of NonStop's scalability and data integrity features. HP also replaced the need for shared memory within JBoss by a fault-tolerant and distributed context-keeping process, so that JBoss applications transparently can be distributed over several shared-nothing CPUs.

Debugging is in the mix

All these development tools are great if you write bug-free code. You probably know some people who don't write bug-free code, however, and therefore you can debug your code running on HP NonStop in a variety of ways:

- If you compile on the HP NonStop, you can debug from a shell prompt on the system.
- If you use NSDEE from Eclipse IDE, you can use the build-in debugger from Eclipse (see Figure 3-1).

Writing Good Software for HP NonStop

Although the HP NonStop OS does provide its core benefits out of the box, some design principles have to be taken into account for an actual application to fully leverage this.

Assembling a team of developers

Getting a team of developers together is easier than it must seem at first, because of the available tools for the platform and the fact that all modern development languages are supported. This is as it should be — there is a straightforward pattern to building a development team for a project of this scope.

What it comes down to is using the best developers in your chosen development environment and language(s). Then, find one or two HP NonStop specialists to engage at the architecture and configuration tier of the effort and make sure the code will perform at the level you expect.

The biggest factor to success on an HP NonStop project is no different than on any other platform: Let senior people who truly understand the requirements as well as the architectural options on HP NonStop run the show. System architecture and best practice can be described by a few specialists, even on a large project.

Writing efficient code

Writing the most efficient code possible improves the performance of HP NonStop applications just as efficient code does in any other environment. Efficiency comes down to throughput, response time, and resource consumption. Balancing

these requirements appropriately for the problem at hand leads to efficient systems that perform well.

Writing efficient code on HP NonStop comes down to understanding and making use of the platform fundamentals. It will not require you to use platform-specific features such as nonstop programming or no-waited programming (see next sections). We discuss this in more depth in Chapter 5.

Nonstop programming

If you deem it necessary, there is the possibility to write your very own code implementing a process as a process pair.

However, this is *never* required or even recommended for an application program — using development frameworks such as Pathway or NSASJ/JBoss will take care of fault tolerance for you without having to write any special code.

No-waited programming

One important — and somewhat unique — technology available on HP NonStop is called *no-waited* programming. No-waited programming appears on the surface to be a lot like so-called asynchronous programming on other platforms but some significant differences exist. Fortunately, this is only needed in rather rare circumstances of low-level systems programming (see Chapter 5 for more details).

Modernizing Existing Applications on HP NonStop

Many applications running on HP NonStop have been written years if not decades ago, many of them in TAL or COBOL. Although they're still providing the functionality they were written for, they often lack a modern GUI or contemporary security features, good integration with the Enterprise architecture, or simply the capability for new development in modern languages.

Fortunately, there are many options for modernizing legacy applications, which often provide a better ROI and less risk than replacing applications that contain hundreds if not thousands of years of stable code (for more on this, see Chapter 5).

Porting Existing Applications to HP NonStop

New development on HP NonStop has become more modern and approachable, but how about migrating existing applications to the platform? Applications that require new levels of reliability often need to change platforms to achieve total fault tolerance — wouldn't it be nice if the platform switch could be done without a lot of code changes?

The best way to look at this is through example. In 2011, HP performed a migration of a real-time fraud detection system. It was written about in detail within a case study you can find at `www.comforte.com/ns4d/migration.pdf`. Speed and fault tolerance were incredibly important because the service was used to check transactions *as they were processed* for fraud.

The original application was built on Oracle, developed in Java using the Spring and Hibernate frameworks. Because HP NonStop embraces these technologies in modern versions of the platform, migration was smooth. HP JToolkit and Pathway were used to improve overall reliability and manageability.

Database migration

Oracle migration to SQL/MX posed few challenges because the original architect of the application kept things simple — proving again that the old *keep it simple* rule holds under many different circumstances. Oracle data types map well to SQL/MX data types, as shown in Table 3-1.

JDBC was used in the reference application for data access to the Hibernate schema. For the OSS application, there is a Hibernate dialect jar, called hibernate3sqlmx, which provides nearly all the same features.

Table 3-1 Mapping Some Oracle Data Types to SQL/MX

Oracle	*SQL/MX*
Char	Char
VarChar2	VarChar
Date	TimeStamp(0)
Number (<5 digits)	Smallint
Number (<10 digits)	Integer
Number (<15 digits)	Largeint
Integer	Numeric(38)
Number(10,3)	Numeric(10,3)
BLOB	VarChar

Application migration

The reference application was built using the Model-View-Controller (MVC) pattern, greatly exposing separation between business logic and data. This design paradigm provided an ease of migration to the application layer similar to what was found in the data layer.

The port replaced standard communication methods (sockets) with HP NonStop-specific methods ($RECEIVE-based messaging and Pathway). Due to the modular architecture of the Java Spring framework, changes were required only at a few central places. The migrated code takes full advantage of the transactional architecture of HP NonStop.

Further considerations

In the past few years, several applications have been successfully ported onto the HP NonStop platform. HP as well as the independent software houses (ISVs) provide a wealth on expertise on how to do this smartly.

Breaking an application into multiple tiers and running these on multiple platforms can also eliminate or simplify the requirement to port the whole application: The parts of the application that remain running on their initial platform will need very little if any porting work.

Chapter 4

HP NonStop as Part of Enterprise IT

In This Chapter

- Securing HP NonStop systems properly
- Integrating with other Enterprise IT systems
- Examining the modernization of the platform and/or applications

Integrating HP NonStop hardware and software with the rest of your enterprise IT infrastructure requires careful planning, but is easier than ever before. HP NonStop has options for REST and SOAP service integration, for MQ messaging, and for other intersystem communication options.

Security is first and foremost in the system architecture, protecting the interactions with other entities. HP NonStop is a first-class citizen of the enterprise environment.

Securing HP NonStop Systems

In general, securing HP NonStop is no different than securing any other server at the core. When thinking about security for NonStop, you should remember a few things, though:

- NonStop applications are mission critical — a failure because of a security flaw could be a disaster.
- NonStop systems very often carry extremely sensitive data like credit card transactions or medical information. Cost must not be an issue when protecting this data.
- From a low-level perspective, NonStop systems are very different from the rest of IT. Security considerations must

consider this. For instance, a virus scanner makes absolutely no sense on NonStop. However, database content encryption can make a lot of sense.

To make NonStop secure, you must consider at least these three facts. But although this list is short, not many people really understand how to do this. Hire experts and let them do their work. Your sleep will be better.

Security has come a long way

In the last ten years or so, the built-in security of the HP NonStop platform has come a very long way by adding new capabilities, such as encryption of data in transit, strong authentication, or integrating them as the source for central event logs (SIEM systems). Depending on the specific implementation and configuration of an individual system, the common security attack vectors still exist, of course, including but not limited to:

- Using application security flaws — for instance, SQL Injection
- Network sniffing
- Attacks against media
- Breaking access control

General platform hardening

HP NonStop has a built-in security framework called Safeguard. Safeguard is a powerful component baked deeply into the NonStop operating system, which handles a lot of the on-platform security by providing user management and the capability to configure Access Control Lists (ACLs) for files and processes.

Then there is the list of details that aren't handled by Safeguard. They really aren't that different than the things you have to do to secure any other server:

- Replace the username/password authentication with SSH certificates or federate with Active Directory or LDAP.

- Let administrative users employ two-factor authentication.
- Encrypt data in transit with SSL or SSH.
- Encrypt data at rest at the application level or by using products that encrypt the underlying database without requiring application changes.
- Maintain file integrity and check system and application object files.
- Given the low number of HP NonStop installations compared to Windows, Unix, and Apple, there has not been a single instance of malware reported on the platform. For that reason, there is no malware scanning tool available.
- Apply reasonable security standards, especially PCI and HIPAA. Be sure to apply the spirit of the standard, not just the letters.

Specific hardening of the HP NonStop platform

The uniqueness of the platform results in having to secure components (such as Pathway) that aren't described in your typical server hardening document. For some resources specific to the platform, see Chapter 5.

IAM: Identity and access management

Large central servers are often in tightly controlled environments, with few users and even fewer administrators. They also last a long time, which means that as users come and go, user accounts often don't. IAM is something that needs to be considered as a process when dealing with HP NonStop.

Identity federation is the best overall solution for IAM on HP NonStop. Leveraging Windows authentication to access the HP NonStop system provides central management and can potentially also provide true single-sign-on. It is also strongly recommended to use two-factor authentication to get access to HP NonStop rather than relying on username and password alone.

Sharing usernames? Don't even think about it.

Another impact of the *set it and forget it* factor of large servers is that log files get lost in the ether. Safeguard, HP's logging system for the Guardian platform, does an excellent job of logging user events. It's highly recommended that those logs make it into a Security Information Event Management (SIEM) system like HP ArcSight. Other exception logs, such as application-specific log files and the EMS messages, should also be sent to a SIEM system.

Keys and certificates

Management of keys and certificates for SSL, SSH, and data-at-rest encryption is no different on HP NonStop than on any other system. As always, maintenance and protection of secret assets is the key, if you'll excuse the pun.

Existing corporate Public Key Infrastructure (PKI) systems should be extended to support HP NonStop applications, just as they should with any other encryption or messaging effort. Implementation of encryption is a one-off programming or installation task, but key management is an ongoing maintenance task. Make use of your existing talents and processes there.

Finally, the SSL and SSH protocols both only protect against sniffing attacks if at least server authentication is configured correctly on both the server (typically HP NonStop) and the client (typically a PC connecting to HP NonStop).

Integrating HP NonStop with Other Systems

For decades, HP NonStop systems were the beginning and the end of the chain of applications that resided there. The program ran on the NonStop and you accessed it with a dumb terminal — it did its job and you did yours. That's it.

Doesn't work that way today, though, does it? Applications and work flows span multiple systems, data has to be accessible from anywhere, and there are four different form factors for every user interface. Integration isn't just important, it's a requirement.

Client-server and multitier design

NonStop lends itself very nicely to multitier applications. Although traditional HP NonStop applications were built entirely on the one platform, today's applications are expected to have the user interface rendered by various other systems, or even handled by the client.

HP NonStop's core architecture is perfect for this. Unlike some monolithic applications where data, business logic, and user interface code are all batched together, HP NonStop applications have separated these tasks since the beginning.

The trend toward apps running on mobile devices is, from a technical viewpoint, just an extension of client/server. It can be accommodated by distributed computing where the NonStop typically will play the role of the transactional database backend.

Raw messaging

One of the reasons that business logic and data are separated in HP NonStop applications is the strong history of messaging on the platform. Raw binary encoded messaging has been implemented since the beginning, either as part of Pathway application design or using both custom developed endpoints and commercial products like the HP Remote Server Call (RSC) product.

In today's environment, binary encoded messaging isn't enough. New messaging protocols have made integration much easier.

Simple Object Access Protocol (SOAP)

SOAP is a rich XML-based message passing protocol using an envelope model for message metadata. SOAP supports many enterprise-required features out of the box, such as reliable messaging, privacy, and transaction support.

HP NonStop systems do come with the HP NonStop SOAP product; also, many Java toolkits have a rich set of SOAP

capabilities. And you can find a number of commercial products out there that provide additional features compared to HP NonStop SOAP.

Representational State Transfer (REST)

REST is the new buzzword and uses the well-established HTTP protocol for message passing. Usually using JavaScript Object Notation (JSON) for data transfer, REST takes advantage of HTTP verbs for CRUD operations:

- **PUT:** Create
- **GET:** Read
- **POST:** Update
- **DELETE:** Delete

REST is usually used for quicker, lighter applications than SOAP, because it doesn't have a thick infrastructure for reliable messaging or security — just what is provided by HTTP. For straightforward CRUD-style interaction with a system's backend, however, REST is an excellent messaging protocol.

REST has become the de facto standard for mobile applications. Support within HP NonStop is significant because of the strong presence of mobile and fault-tolerant systems in telecommunications and mobile payment systems.

REST-based frameworks are available for NonStop from several third parties, and they give application developers an awesome, reliable way to integrate HP NonStop servers into modern application designs.

Operations Management

Watching what a system does is an important step to prevent the unexpected and undesired. EMS is the main system event message hub for the HP NonStop (for more on this, see Chapter 2). However, several commercial products are available that allow you to monitor an HP NonStop system together with other systems, under a single operative umbrella.

To name a few:

- Nagios is an open source tool that has been successfully adapted to monitoring HP NonStop systems.
- HP itself offers several tools; visit the HP NonStop website (see Chapter 5 for this) and search for "HP NonStop manageability" products.
- The Tivoli suite from IBM happily interoperates with HP NonStop.
- There are several third-party ("ISV", see Chapter 5) products that provide powerful capabilities.

You can find a number of products that put a GUI on top of the command-line or block-mode screen-based tools HP NonStop offers out of the box. Refer to Figure 2-1 to see a screen shot of one of these.

File Transfer

Moving files around is arguably very important in enterprise integration. File Transfer Protocol (FTP) is the venerable protocol developed for Internet connected computers, and is, of course, supported on HP NonStop.

These days, FTP has been superseded by SFTP or SFTP/SSL, which are the secure file transfer protocols that are gradually replacing FTP. Fortunately, both are supported on HP NonStop.

If you need advanced features such as guaranteed file delivery, there are several commercial products available that implement what is then called *managed file transfer*.

Messaging

In Chapter 2, we discuss the fact that the HP NonStop OS is message-based at its core. In this section, we discuss messaging in a completely different context.

Asynchronous messaging means that applications send messages without waiting for an immediate reply. These messages are typically stored in a database, but only for a short period

of time until they can be delivered to the final destination (or to the next station if they have to travel a long way). Email systems work that way, but in the NonStop world, messages are typically sent from application to application — where applications can run on different platforms. These kinds of messaging applications are used for quite different purposes on NonStop:

- Manufacturing devices send status or work progress messages to central servers.
- Banking applications send messages to other banks.
- Credit card processing applications send messages to central billing systems.
- Many other examples are possible.

The common properties are that the messaging must be reliable, messages must be sent exactly once, the sequence of messages is typically important, and the application doesn't track the message. The underlying infrastructure where the application put the message must be so reliable that there is no need to care after the message has been put.

Messaging is very important in modern applications. For Java-based applications, there is the JMS standard defining a common, standardized interface how messages can be sent and received. As JMS is only an interface standard, there may be many different implementations, even on the same platform.

For NonStop systems, many industry-strength messaging applications are available right now:

- There is a JMS implementation based on SQL/MX available from HP (mainly for local messaging).
- Another JMS implementation based on Apache Active MQ is also available from HP (product name NSMQ).
- Websphere MQ, IBM's very popular messaging product, is available for NonStop from IBM.
- Finally, other third parties might also provide messaging solutions for NonStop.

Chapter 5

Ten Resources for More Information

In This Chapter

- Ten resources for HP NonStop information

There is more to the HP NonStop platform than is possible to fit into a little bitty book like this. The tome in your hands is a good primer, but you probably have more questions. In this chapter, you find links with a lot of additional information to help you find out even more. From developing to auditing to securing, these references are your next step to expanding your knowledge of HP NonStop.

NonStop Terminology and Entities: A Quick Reference

Throughout this book, we include terms that may be new to you, such as TACL, TAL, Pathway, Safeguard, SCF, and EMS. Although we could not squeeze a reference section into this book, you can find some extra information on these and other terms at `www.comforte.com/ns4d/quickref`.

TACL Fact Sheet

Do you know your way around the Windows CMD prompt and/or the Linux/Unix Shell and want to quickly find a command? Fear no more; the table at `www.comforte.com/ns4d/taclcheatsheet` will help you.

Open Source Tools Available on HP NonStop

We mention open source tools in several locations in this book, but we do not provide either potential sources or a list of available tools. If you are interested in these details, please see www.comforte.com/ns4d/opensource.

Tips for Developing for HP NonStop

Developing for HP NonStop uses the same languages, IDEs, and skills as contemporary applications (this is also discussed in Chapter 3). Developers will find that the life cycle that they're familiar with still applies in nearly every case. That said, some tips will make the platform feel more functional and reasonable. Find these tips at www.comforte.com/ns4d/devtips.

No-waited Programming

We briefly mention no-waited programming in Chapter 3. In a nutshell, this is required only for system-level programming where your application wants to handle multiple events in parallel. If you want to know all the gory details, find them at www.comforte.com/ns4d/nowaited.

Modernizing Legacy Applications

Although the HP NonStop platform is very modern, some applications would benefit from a little refresher. An abundance of options are discussed in more depth at www.comforte.com/ns4d/modernize.

Securing NonStop Systems

Many HP NonStop users run the platform to process credit and debit card data — either through connected point of sales (POS) systems or through ATMs. In both cases, the HP NonStop is at the core of the architecture, making it a prime target for cybercriminals. Because of this association with the financial industry, many HP NonStop customers fall under the Payment Card Industry Data Security Standard (PCI-DSS).

Although your security architecture should ideally not be driven by a single regulatory framework, the PCI-DSS is widely regarded as a well-worded guideline to make sure you have an appropriate security posture.

For more resources on the topic of PCI on NonStop as well as on securing NonStop systems in general, please go to `www.comforte.com/ns4d/security4d`.

The NonStop ISV Ecosystem

Old-timers in the NonStop (Tandem) world know what an ISV is. ISV stands for *independent software vendor* and simply means a software house that isn't part of HP and which develops and markets software specific for HP NonStop systems. comForte, the company behind this book, is an ISV.

There are plenty of these around, and many of them are listed on the HP website (see next section) under partners. Also, you will see them as sponsors of some of the User community meetings (see section "The User Community").

ISV applications and system software play an important role to make the HP NonStop platform attractive — without software and tools, the best hardware has only very limited use.

The HP Website and HP NonStop Computing

Hewlett Packard is a large organization with more than 300,000 employees working on a huge set of products and services. Designing a web page to inform your customers and to offer support for both humble printers for home usage and high-end systems like HP NonStop is a challenge.

However, you can find a lot of great resources about HP NonStop on the HP web page. The URL `www.hp.com/go/nonstop` is a helpful shortcut to get to these pages, which we recommend highly.

Also, the URL `www.hp.com/go/nonstop-docs` will take you right to the extensive documentation for HP NonStop systems.

The User Community

Most people dealing with HP NonStop systems are passionate about the platform. Ever since the start, there was always a strong community around the technology, supported by HP (Tandem) and the ISVs, but really driven by people who know that technology alone doesn't make projects succeed. Sharing knowledge about technology is a great help, however, and that's what this community is all about.

Today the community is strong as ever with online resources, a printed bimonthly magazine, and regular user group meetings from small local meetings to large gatherings. See `www.comforte.com/ns4d/community` for more.

Find out more about HP NonStop!

This book is for anyone who needs computers that never stop working. This book takes you through the best cases for integrating a fault-tolerant system into your information technology plans.

- *HP NonStop 101 — learn more about HP NonStop systems*
- *Examining the platform — find out why the platform is unmatched in the areas of high availability and linear scalability*
- *Working with HP NonStop — discover how to develop new applications on the platform and how to modernize existing ones*
- *Integrating HP NonStop — explore ways to better integrate your HP NonStop systems in the enterprise IT infrastructure*

Open the book and find:

- Sources for more information about HP NonStop
- The components of HP NonStop
- Key benefits of HP NonStop
- How to secure HP NonStop systems
- Info about what programming languages to use with HP NonStop

Go to Dummies.com®
for videos, step-by-step examples, how-to articles, or to shop!

FOR DUMMIES®
A Wiley Brand

ISBN 978-1-118-84993-4

WILEY END USER LICENSE AGREEMENT

Go to www.wiley.com/go/eula to access Wiley’s ebook EULA.