

# An Introduction to .NET for MPE People

#### **Rich Trapp**

Senior Consultant Managed Business Solutions

- In the HPe3000 community, organizations have been running business critical applications on the HPe3000 over the last 20 years
- HP announced the sunset of the HPe3000 in 2001
- The organizations in the HPe3000 community have been developing a transition plan for migrating their applications off the HPe3000



#### The options for transition are:

- STAY: Remain on the existing HPe3000 platform with the existing application(s)
- **PORT:** Move the existing application(s) as is to a new platform
- -BUILD: Re-write or re-engineer the application(s) on a new platform, often enhancing the applications significantly
- **BUY:** Purchase an off-the-shelf application package to replace the functionality of the existing applications



#### • .NET may be involved if the organization chooses:

- PORT: Some Port tools convert the existing applications into .NET; further enhancements may involve .NET development
- BUILD: The application(s) may be re-written or reengineered in a .NET development environment
- BUY: Many off-the-shelf packages are now implemented in a .NET development environment; customizations and interfaces may best be written in .NET
- .NET is likely to be in your organization's future



#### In the HPe3000 community, the current IT staff is skilled in:

- COBOL
- 4gl's, such as COGNOS or Speedware
- Image
- MPE
- This staff will likely need to transition their skill set to .NET
  - VB.NET
  - C#
  - ADO.NET
  - SQL Server or other relational databases
  - BAT files and the Windows scheduler



#### Agenda

#### This presentation discusses

- The major differences between the HP3000 and .NET
- The benefits of .NET
- Getting started with .NET



#### Major Differences Between the HP3000 and .NET

- The major differences between the HP3000 and .NET development environment are
  - The IDE
  - Object Oriented Design (OOD)
  - Relational Databases
  - Jobs and Job Scheduling



- An Integrated Development Environment (IDE) is an application that allows for comprehensive development of application source code
- For .NET, this is Visual Studio.NET
- The IDE replaces the role of
  - -QEDIT
  - The COBOL compiler
  - The 4gl compiler / interpreters
  - Screen designers, such as VPlus
  - Debuggers



#### Features of Visual Studio.NET

- Visual Screen Layout: draw the screen instead of typing characters on a 24 x 80 screen
- Code Generation: the drawn screens automatically create code behind the scenes to support itself
- Objects Library:
  - File management
  - Date / Time
  - Array objects
  - Hash tables
  - Lots and lots of other routines

#### – Source Level Debugging:

- Call stack: keeps track of all of the method calls
- Step-by-step view of source code execution



#### Features of Visual Studio.NET

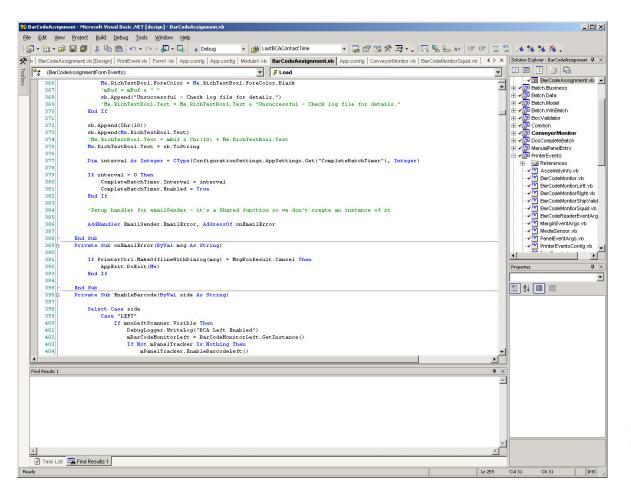
- Auto Formatting: automatically formats your source code
- -IntelliSense: if working with a reserved word or object, Auto Complete shows the valid options based on what's typed so far
- Integrated Help: highlight a reserved word and get a helpful description of it
- Go To Definition: right click on a word or procedure code and the IDE takes you to that source code
- Highlights: color coding of
  - Reserved words
  - Objects
  - Methods



- Visual Studio.NET Module Structure
  - Visual Studio.NET collects classes into projects
  - Visual Studio.NET collects projects into solutions
  - These items comprise:
    - Executables (.EXE)
    - Libraries (DLL's)
  - As an example, a typical project might have:
    - 5 DLL's
    - 5 FXF's
    - Established as 1 solution and 10 projects



 Sample Screen of Visual Studio.NET





- The primary difference between development in an HP3000 environment and a NET environment is:
  - The HP3000 is geared toward procedural design
  - NET is geared toward object oriented design
- Procedural Design:
  - Procedures or steps are established as a sequence of commands, acting on data structures
- Object Oriented Design:
  - Developers model real-world situations and business scenarios as objects that perform actions, have properties, and trigger events.



- Object Oriented Design contains the following concepts:
  - Classes: Definitions of common objects (i.e. a data type), to include the structure(s) of the data, and the procedures that act on that data
  - Methods: The procedures that act on the data
  - Objects: An instantiation of a class (i.e. a variable)
  - Properties: A member of a class that can implement "get" and "set" accessors and can be used like a variable
  - Shared Classes: Classes that do not require instantiation



- Classes provide inheritance
  - Sub-classes belong to classes
  - A sub-class inherits all data structures from its parent class
  - A sub-class inherits methods from its parent class
- Inheritance is important because
  - It promotes software re-usability
  - It standardizes coding approaches across the entire application environment
  - It reduces software development time



#### • Example: Class inheritance—a form

ờ Batch.WinBatch - Microsoft Visual Basic .NET [design] - PrintBatcherForm.vb [Design]	
Elle Edit View Project Build Debug Data Figmat Iools Window Help	
沼・浩・彦 🗑 🕼 🚴 🕫 🔍 🕫 - 💭 - 📮 - 📮 , Debug 🔹 🕫 LastBCAContactTime 💿 🗔 🗃 🕲 🛠 🦉	
其 [ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$	
🔆 b.config   ConveyorMonitor.vb   BarCodeMonitorSquid.vb   BarCodeAssignmentPage.vb   BatchListPage.vb   Design]   BatchListPage.vb   PrintBatcherForm.vb   Design]   D	
Cyclica Scaners	⊕ obj
	App.config
Batch Paper 🕨 II 🖌 🚳 🗖	AssemblyInfo.vb
	BarCodeAssignmentPage.vb     Batch.WinBatch.vbproj.vspsg
Batch Properties Batch Status	Batch./vinbatch.vbproj.vspst     BatchListPage.vb
Print Format: Single Sheet: Started:	Copy of BatchListPage.vb
Single Panel:	ErrorForm.vb
Left Roll Edge: 0 Right Roll Edge: 50 Paused:	
Left Roll Edge: C Right Roll Edge: C Finished	€ ✓ ∰ JobListPage.vb
옵 목 Left Sheet Job Tasks Right Sheet	
े g Left Sheet Job Tasks Right Sheet	PrintBatcherApp.vb
	PrintBatcherForm.vb     PrintQueuePage.vb
	Vssver.scc
	E - ✓ BoxValidator
β × ×	
BarCode	Properties 4 ×
	BatchListPage Circle.Production.Batch
	Design
× · · · · · · · · · · · · · · · · · · ·	(Name) BatchListPage
	Locked False Modifiers Friend
	E Focus
	CausesValidation True
	E Layout
	Anchor Top, Left, Right AutoScroll False
	AutoScrollMargin 0,0
	AutoScrollMinSize 0, 0
	Dock None
	E Location 0,0
	🗄 🗄 Size 760, 192
Pino results 1 ***	FormContainer Specified cast is not
	Misc
C Task List 🔤 Find Results 1	
Ready	

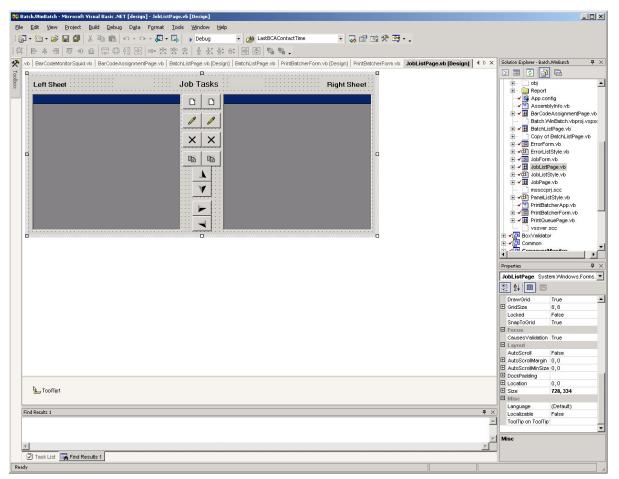


#### Example: Class inheritance—a form

😏 Batch.WinBatch - Microsoft Visual Basic .NET [design] - BatchListPage.vb [Design]										
Elle Edit View Project Build Debug Data Farmat Iools Window Help										
🏹 * 🛅 * 🝘 💭 🐉 🐁 🗈 🔃 い * < * - 4月 * 🖳 , Debug 🔹 😝 LastBCAContactTime 🔹 🖓 🖓 🚱 😨 🛠 🦉 * 🖕										
🔆 þ.config   ConveyorMontor.vb   BerCodeMontorSquid.vb   BarCodeAssignmentPage.vb   BatchListPage.vb   Design]   BatchListPage.vb   PrintBatcherForm.vb   Cesign]   PrintBatcherForm.vb   🔶 🗙	Solution Explorer - Batch.WinBatch 4 ×									
Batch Paper 🕨 II 🖌 🍘 🗅	⊕-      ⊖ obj									
	App.config									
Batch Properties Batch Status										
D Print Format: Single Sheet: Started	BarCodeAssignmentPage.vb     Batch.WinBatch.vbproj.vspsc									
Single Panel:	Batch.istPage.vb									
Left Roll Edge: 0 Right Roll Edge: 50 Paused	Copy of BatchListPage.vb									
Left Noil Coge	ErrorForm.vb									
	✓      JobListPage.vb									
	✓ <sup>™</sup> PrintBatcherApp.vb									
	PrintBatcherForm.vb     PrintQueuePage.vb									
	vssver.scc									
	Conversionitor									
	Properties 7 ×									
	BatchListPage Circle.Production.Batch									
	Design									
	(Name) BatchListPage									
	DrawGrid True									
	Locked False									
	SnapToGrid True									
	E Focus									
	CausesValidation True									
	AutoScroll False									
b ToolTip1 ☑ PrintPreview Ø CompleteBatchTimer	AutoScrollMargin 0,0									
	AutoScrollMinSize 0,0     DockPadding									
	E Location 0,0									
Find Results 1	⊞ Size 760, 192									
Find Nexults 1	Misc     Language (Default)									
	Localizable False									
×	Misc									
🖉 Task List 📴 Find Results 1										
Ready	<i>h</i> .									



#### • Example: Class inheritance—a form





- OOD includes the use of Events
  - Events are raised by a method
  - Events are handled by calling objects
  - Events allow objects to focus on their own task and to notify calling methods of issues
  - Events prevent called objects from having to handle issues that are beyond their scope
  - This eliminates direct calls from one procedure to another, when they are unrelated
  - This eliminates 'spaghetti code'



# Example: Event raising and handling

💔 P	rinte	rEvents -	Microsoft Visu	al Basic .NE	r [design	] - BarCo	deMonit	orSquid	.vb												
Eile	Ēc	lit ⊻iew	Project E	uild <u>D</u> ebu	g <u>T</u> ools	<u>W</u> indo	w <u>H</u> el	2													
1	] - 🏌	🗓 🕶 🗃	🖬 🕼   %	B 🛍	∞ + 0	a = 🚛	- 🖪	Det	bug	-	🔤 🗠	astBCACor	ntactTime		-	🌄 🖆	' 1 🕉	و ج	• • [ ]	2	∕ <b>≜</b>
	nment	.vb (Desig	gn]   PrintEvent	.vb   Form1	vb   App	.config	App.con	fig   Mo	dule1.vb	BarCo	odeAssig	inment.vb	App.confi	ig   Conv	eyorMo	nitor.vb	BarCo	deMor	nitorSqu	id.vb	$\P \triangleright \mathbf{X}$
Toolbox	BarCodeMonitorSquid 🔹 🕅 (Declarations)										•										
XOC		6 7 ⊟ N 8   9	amespace C	ircle.Pro	ductio:	n.Prin	Tools:														
	10 Public Class BarCodeMonitorSquid : Inherits BarCodeMonitor 11																				
		12 13											A 49.000 (1999)	, Була.	I e A	s Bart	odekea	derkv	encarg	5)	
	14         Public Event BarCodeReaderErrorEvent As BarCodeReaderEventHandler           15         Public Event BarCodeReaderReadEvent As BarCodeReaderEventHandler           16         16																				
		17	'Si	ngleton I	nstanc	e															
		18		vate Shar		-				deMon:	itorSqu	aid									
		19	Pri	vate Shar	<mark>ed</mark> mTe	stOk A	s Boole	an =	False												
		20 21 🖯	Pri	vate Sub	New (By	Val po:	tNumbe	r As	String	r)											
		22		MyBase.N	ew(por	tNumbe	c)														-
	•																				•
Rea	dy														Ln 20	)	Col 1	C	h 1		INS
																		HP	wo s and Techn		2004 erence & Expo

# Example: Event raising and handling

💔 P	PrinterEv	sts - Microsoft Visual Basic .NET [design] - BarCodeMonitorSquid.vb
Eile	e <u>E</u> dit	<u>v</u> iew <u>P</u> roject <u>B</u> uild <u>D</u> ebug <u>T</u> ools <u>W</u> indow <u>H</u> elp
1	] • ዀ	🚔 🔚 🕼 🐇 🛍 💼 🗠 - 여
*	nment.vb	Design]   PrintEvent.vb   Form1.vb   App.config   App.config   Module1.vb   BarCodeAssignment.vb   App.config   ConveyorMonitor.vb   BarCodeMonitorSquid.vb   🕨 🗡
Toolbox	প 🕻 Bar	odeMonitorSquid 🔽 🚺 (Declarations)
box	8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8	DebugLogger.WriteLog(barCodeValue.ToString)  If barCodeValue.ToString = "OK" Then  mTestOk = True  Blse sb.Append("Read Bar Code Value ") sb.Append(barCodeValue)
		<pre>barCodeValue.Length = 0 Else barCodeValue.Append(Chr(mCP.InputStream(0))) End If</pre>
	•	
Rea	dy	Ln 20 Col 1 Ch 1 INS
		HP WORLD 2004 Solutions and Technology Conference & Expo

# Example: Event raising and handling

<b>90</b> I	PrintingProcess - M	icrosoft Yisual Basic .NET [design] - ConveyorMonitor.vb						
Eile	e <u>E</u> dit <u>V</u> iew (	roject <u>B</u> uild <u>D</u> ebug <u>T</u> ools <u>W</u> indow <u>H</u> elp						
1	] • 🛅 • 🚔 🔒	🕼 🐰 🛍 🛍 🗠 - 🖙 - 💭 - 🖳 - Debug - 🍻 mBarCodeMonitor - 🕞 🐼 😤 🐼 🗉 🗟 🖕 Azt 🚍 😩 🔺						
*	ker.vb   BarCodeA	ssignment.vb [Design]   PrintEvent.vb   Form1.vb   App.config   App.config   Module1.vb   BarCodeAssignment.vb   App.config   ConveyorMonitor.vb   BarCodeMonitorSquid.vb   🔸 🗙						
Toolbox	ConveyorMonitor							
box	91	If (Me.mEnableBarCode) Then						
	92	'exception possible here						
_	93	While mBarCodeMonitor Is Nothing						
	94	Try						
	95	mBarCodeMonitor = BarCodeMonitorSquid.GetInstance()						
	96	Catch ex As Exception						
	97	sb.Append("Barcode Scanner open failed. Please run BLSetup.")						
	98	sb.Append(Chr(10))						
	99	sb. Append(ex. Message)						
	100 If MsgBox(sb.ToString, MsgBoxStyle.RetryCancel, "Open Barcode Scanner") = MsgBoxResult.Cancel Then							
	101 AppExit.DoExit(Me)							
	102 End If							
	103	End Try						
	104	End While						
	105							
	106	AddHandler mBarCodeMonitor.BarCodeReaderReadEvent, AddressOf OnBarCodeReadEvent						
	107							
	•							
Rea	ady	Ln 93 Col 38 Ch 38 INS 🦽						
		HP WORLD 2004 Solutions and Technology Conference & Expo						

- OOD includes the use of Exceptions
  - Exceptions are raised by a method
  - Exceptions are handled by calling objects
  - Exceptions allow objects to focus on their own task and to notify calling methods of issues
  - Exceptions prevent called objects from having to handle issues that are beyond their scope
  - This eliminates direct calls from one procedure to another, when they are unrelated
  - This eliminates 'spaghetti code'
  - Unlike events, exceptions use a Try, Catch, Clean Up code structure



#### • Example: Exception raising and handling

<mark>98</mark> I	PrintingProcess	Microsoft Yisual Basic .NET [design] - ConveyorMonitor.vb								
Eile	e <u>E</u> dit <u>V</u> iew	Project Build Debug Tools Window Help								
1	- 🛅 - 🚔	🔚 🗊 🕺 🛍 🛍 🗠 - 🖙 - 💭 - 🖳 - Debug 🛛 - 🍻 mBarCodeMonitor 💿 - 😡 😭 🐼 🛠 🏹 - 🛛 🖽 🌭 🗛 🙄 😩 🖉	•							
*	ker.vb   BarCod	eAssignment.vb [Design]   PrintEvent.vb   Form1.vb   App.config   App.config   Module1.vb   BarCodeAssignment.vb   App.config   ConveyorMonitor.vb   BarCodeMonitorSquid.vb	$\leftrightarrow \times$							
Toolbox	j ConveyorMonitor									
Ъх	91	If (Me.mEnableBarCode) Then								
	92	'exception possible here	-							
	93	While mBarCodeMonitor Is Nothing								
	94	Try								
	95	mBarCodeMonitor = BarCodeMonitorSquid.GetInstance()								
	96	Catch ex As Exception								
	97 sb.Append("Barcode Scanner open failed. Please run BLSetup.")									
	98 sb.Append(Chr(10))									
	99 sb.Append(ex.Message)									
	100 If MsgBox(sb.ToString, MsgBoxStyle.RetryCancel, "Open Barcode Scanner") = MsgBoxResult.Cancel Then									
	101	AppExit.DoExit(Me)								
	102	End If								
	103	End Try								
	104	End While								
	105									
	106	AddHandler mBarCodeMonitor.BarCodeReaderReadEvent, AddressOf OnBarCodeReadEvent								
	107		-							
	▲									
Rea	ady	Ln 93 Col 38 Ch 38	INS							



There are no pointers in .NET!

- (OK, there are, but you can't see them)
- Objects are instantiated with a name
- .NET is responsible for its own garbage collection



#### **Relational Databases**

- Image is not readily available in a .NET environment
- The databases used with .NET are typically relational:
  - SQL Server
  - Oracle



#### **Relational Databases**

Image is a Network Database

- Allows Master to Detail relationships
- Relational Databases allow
  - A relationship to be established between any two (or more) tables
    - Provides greater data modeling flexibility
    - Encourages normalization of the data
    - Improves the maintainability of the applications



#### **Relational Databases**

- .NET provides classes and methods that
  - Load database records into data structures
  - Associate screen fields directly to database records without having to write SQL
  - Handle transactions and rollbacks
- Database Administrators are required for relational database packages



#### Jobs and Job Scheduling

- Many kinds of job schedulers are available
  - Windows
  - SQL Server
  - Off-the-shelf
- These schedulers can invoke .NET programs directly, eliminating the need for 'jobs' in many cases
- Script files can be created for required system functions
- The biggest issue is tracking which schedulers are running which jobs

# Major Differences Between the HP3000 and .NET

- .NET is a very different environment from the HP3000
- The biggest difference is the object oriented environment instead of the procedural environment
- .NET is supported by very good tools
  - The IDE
  - Classes, methods, and objects
  - Event and exception handling
  - Databases
  - Job schedulers



- Learning .NET requires:
  - Training
  - Experience

Experience is crucial to the learning process



- Training: Good Books
  - Microsoft's "Visual Basic .NET Step by Step"
  - Microsoft's "Visual Basic .NET Core Reference"
- Books by MBS'er Kevin Hoffman:
  - "Professional .NET Framework, by Wrox Press
  - "Professional ADO.NET", by Wrox Press
  - "C# Programming Evolution", by SAMS Press
  - "Visual C#.NET 2003 Unleashed", by SAMS Press (forthcoming)
- Training: Pitfalls
  - The books are often geared to a VB6 audience, not an HP3000 audience
  - The books often contain simplistic examples that underplay real world complexity



- Training: Understanding The CLR
  - CLR stands for Common Language Runtime
  - The CLR functions in the background
  - The CLR does its job; programmers do not generally have to worry about it
  - All .NET books open with a discussion of the CLR



- To gain experience with .NET, follow these steps
  - Use the IDE
  - Use the debugger
  - Learn the libraries
  - Understand OOD
  - Get experience with .NET 'quirks'



- Experience: Use the IDE
  - Learning the editor and debugger is easy
  - The biggest issue is that the number of modules (classes) in an application is intimidating
- Experience: Use the debugger
  - Start with a sizable application, and walk through the execution of the source code using the Visual Studio.NET Debugger
  - The syntax is not too hard to understand
  - Biggest issue is understanding 'How did I get here?'



- Experience: Learn the libraries
  - There is a tremendous number of objects provided by **Microsoft**
  - Help and F1 are your friends
  - Internet sites, such as Google, can help programmers find ones that are needed



- Experience: Understand OOD
  - Becoming used to class inheritance
  - Keeping track of the call stack
  - Need to view classes and subclasses as 'superimposed' code
    - Sometimes a parent class will call a subclass's method
    - Sometimes a subclass will call a parent class method
  - Understanding that everything is an object
    - For example, strings now have methods associated with them
  - Understanding that events control the programs (e.g. user events), not the procedures
  - Hardest part of learning .NET



 Experience: Get experience with .NET 'quirks' - War Story: Literal over-written by garbage collection



- The greatest similarities between HPe3000 and .NET will be:
  - Project deadlines
  - Programming and logic skills
  - Problem solving skills
  - Figuring out why something doesn't work
- But otherwise, it's a completely different environment
- Plan on 3 6 months to become functional in .NET



#### Benefits of .NET

- The IDE provides efficient development, code generation and extensive on-line help
- OOD enforces excellent programming standards
- OOD allows for localized maintenance and enhancements in the future
  - Eliminates interdependence of procedures



#### Benefits of .NET

- There is an extraordinary amount of tools and libraries available
  - Microsoft provided objects and libraries
  - Free objects and libraries on the web
  - Interfacing mechanisms, such as XML
- Application integration is becoming easier
  - -Web services
  - Standardized interface formats
  - There is no longer a need for fixed format files



#### Conclusion

- .NET will be a major player in the development of applications well into the future
- There is a growing market share, resource base, and material available for .NET development
- This is occurring because .NET provides
  - Efficient development
  - Well structured applications
  - A large number of interfacing techniques and interfaces
  - A large quantity of existing, re-usable source code



#### "Wherever you go, there you are."

#### - Buckaroo Bonzai





Co-produced by:

