Focus: Secure Your Passwords
Berndt Hamboeck

Use cryptography to provide functionality in your apps

PBDJ Feature: The PowerBuilder Integrated Development Environment: Understanding the IDE
Bill Green

Book Excerpt: Dynamically Creating DataWindow Objects: It's easier than you think
Bob Hendry

Nonstandard Data Model: Hierarchical Data: Reduce coding changes
Glynn Naughton

PART 1
Berndt Hamboeck

Components: NVOs and NVCs: What's the difference?
Jerry Neppl

From Sybase: Got Code? Want Code?: Come to the source
Jessica Bronesky
Amyuni Technologies, Inc.
www.amyuni.com
PB9 En Route

PBNI tears down the barriers

The availability of PowerBuilder 9 was announced March 4 and the release was made available March 24. If you have an upgrade subscription or have purchased a copy, then you’ve probably already received it by the time you read this. I have the beta version, but haven’t yet received the official final copy, though I’m currently watching my mailbox in hopeful anticipation of an imminent delivery. When I receive it, I’ll put it to good use right away. I don’t know what you’re doing with PowerBuilder, but for what I’m doing, the new PB9 features are great!

Allow me to put on my client/server hat and tell you what I like about PB9. PBNI!! The second best thing is PBNI, and third, PBNI!! In my opinion, for client/server development the PowerBuilder Native Interface is the single biggest advancement to PowerBuilder since the NVO was introduced with version 4 back in 1994. Until PB9, PowerBuilder could call functions in external DLLs, but that functionality was limited by issues such as the inability to execute callbacks. That severely restricted the interoperability of PB with C/C++ and meant the external functions could be called, but as resources without full two-way communication capabilities.

To achieve interaction with externals, OLE was used, but it has its own limitations. In a similar way, other languages were unable to fully interact with PB objects because they could only call PB via OLE. There was no ability to directly call PB objects in the same way a function in a C DLL would call a function in another C DLL. Of course, that meant it was impossible for a C++ class to be inherited from a PB class. Those inadequate capabilities severely limited the use of PowerBuilder in hybrid systems, which most systems are.

With PB9 and PBNI the gates are flung wide open. The limitations and restrictions mentioned above have been eliminated. Here’s what Roy Kiesler, the expert on PBNI, has to say about it in the book PowerBuilder: Advanced Client/Server Development, edited by Bruce Armstrong and Millard Brown: “Since its inception, PBNI has gained remarkable popularity within Sybase; PowerBuilder 9.0 ships with many new features, three of which – EJB client, Web services client, and the PBDOM XML parser – are PBNI implementations. Those, however, are merely the tip of an iceberg; the promise of PBNI can lead PowerBuilder programmers to new frontiers like Microsoft .NET, and has the potential to establish PowerBuilder as a new component model, alongside Microsoft’s COM and Borland’s VCL.”

PBNI is a C++ type programming interface modeled after the Java Native Interface (JNI), and allows other languages to interact with PB like they do with C++. So PB now has the interoperability power of C++! Certainly this is a great addition to client/server development, but Web/n-tier developers also have a lot to gain, as is obvious since the EJB client, Web services client, and PBDOM XML parser are all based on PBNI.

Bruce Armstrong says, “This has the potential to do for PowerBuilder what VBX controls did for Framework.”

PBNI tears down the few barriers that have kept PowerBuilder from being fully integrated as an open tool in hybrid systems. It brings PowerBuilder into the mainstream and puts it on an equal technical footing with C/C++, while increasing its 4GL advantage over the other tools. As with NVOs, the full capabilities of PBNI and the way it will be used won’t be fully known until developers start using it. Maybe it will result in a whole new generation of application architectures, like the implementation of NVOs did.
Being an inquisitive teacher I'm always searching the Web looking for educational offerings. For the past 10 years I've made my living teaching at a university.

When class is not in session, I take my show on the road and visit training rooms throughout corporate America.

Like most paranoid teachers, I'm always making sure my course material is relevant and fresh; don't want to be teaching classes such as “XML and Punch Cards” or “The UNIVAC and You,” so I visit other schools and training centers to see what they're up to. Over the past year, I have found an alarming trend in how courses are marketed.

One morning I was on the Web looking at all the XML offerings at various training centers. After about an hour, a clear pattern started to emerge. I call it “tier” marketing. Some training centers were marketing the same courses packaged as “good,” “better,” and “best” versions. Of course, the “best” courses were more expensive but, wow, do you get more material — or so they say. I called one training center and asked for a more detailed syllabus of their “best” XML training course. Five minutes later I received an e-mail that contained so many topics I almost fell off my chair. By my estimation, covering all those topics would take weeks, not 40 hours.

Shocked, I called the center back for more info. I was told that all topics are explained “thoroughly and comprehensively” and that the course was “accelerated and not for the faint of heart.” Her answers flew in the face of logic — but are what potential students want to hear.

‘Be an Expert in One Week!’

Come on. How can anyone be an expert in anything in one week? I just started taking piano lessons. So far I'm six weeks into it and still have to practice while my family is out of earshot. I'm bad, really bad. However, that doesn't mean I won't be good some day. With instruction and practice (yes, I said practice) I could be playing Carnegie Hall some day — okay, maybe a VFW hall.

Courses are starting to be marketed like amusement park thrill rides. The “good” class is the merry-go-round, while the “best” class is the roller coaster — very clever marketing. Which ride do you want to go on?

This approach is doomed from the start. I can only imagine the folks who fork out thousands to sit in a class waiting for the magic to happen. Maybe the instructor passes out a pill on the first day (stay away from the Kool-Aid)? With such unrealistic expectations, give your money to a faith healer or a television evangelist.

The problem is that more realistic training centers and schools have to compete with this. More and more teachers and curriculum developers are pressured to fit more material into less time. That's what people are paying for. After all, who wants to buy “Seven Minute Abs” when you can buy “Six Minute Abs”? It's become almost absurd.

Corporate vs Academic Training

Once you decide on a realistic course, be aware of the differences between the format of a corporate training center and an academic institution. A corporate training course usually takes place during regular business hours. The course material is usually a binder rather than a book, and the course is pretty much scripted from start to finish. After completion of the course, you will receive a “certificate” rather than a grade.

On the other hand, academic institutions meet on a schedule that includes evenings and weekends. The course material is usually a book complemented with class handouts. Since there is time between classes, the course content can be modified to fit the student body. At the end of the course you're given a grade.

Which one suits you? It depends. In most cases, if your employer will reimburse the cost of the course, you go where they tell you. If the location is up to you, use the following guidelines.

If you're a quick learner and need little or no time to absorb content between classes, corporate training is for you. On the flip side, if you need time to absorb the lessons and like a time cushion between classes — go back to college. Keep in mind that corporate training is usually more expensive than a college course.

Is one better than the other? Not really. Corporate training centers probably don't want you to know that many of their instructors trade their suit for a pair of jeans and a T-shirt at 5:00, then drive off to a local community college to teach the same course.

Final Thoughts

Where is the magic in training? Believe it or not, it's in you the student. Take all the variables that go into learning a skill: teacher, course material, faculty, and curriculum. To the surprise of many, chances of success or failure rest squarely on the shoulders of the student. All other factors are the supporting infrastructure in the learning world. Taking a class at a training center or local university is a great self-investment. Keep in mind that knowledge is instruction and practice over a period of time. Education is a process, not a pill.
Sybase, Inc

www.sybase.com/powerbuilder
As architects of enterprise systems, we know that security should always be a concern when we transfer or store sensitive data. In the past, you might have cut corners because you assumed the corporate LAN provided some implicit level of protection. You might have rationalized that because only people with access to the LAN could see data being transferred across the wire – and surely no one at your company would try to steal sensitive data – your data was secure. Likewise, your internal applications might not have been passing sensitive information and you leveraged password protection on the operating system. Although corporate LANs, corporate WANs, and intranets might provide adequate security for some applications, many applications are now deployed across wide-area public networks such as the Internet. The danger is that along with the ease of accessibility to these public networks comes, potentially, lots of prying eyes. Protecting sensitive data in this environment isn’t an option – it’s a requirement. If you expose someone’s password or credit card information to an unauthorized source, you could incur legal action and lose customers and business.

The solution is for us to use cryptography. Cryptography, the science of information security, can address many of the issues associated with applications that handle sensitive data. Namely, it can prevent unauthorized persons from reading the data. There are many topics related to cryptography – enough to fill several books.

In this article, I’ll examine three useful cryptographic techniques: hashing, encryption, and decryption. I’ll discuss how each technique can improve application security and how to implement each technique in PowerBuilder.

The Simple Password Model
The simple password model illustrates the fundamental concepts behind password-based security. A password entered by a user is compared with a password stored on a system or in an executable file. If the passwords match, a secured operation is permitted.

Implementing the most popular approach to secured functionality – requiring the user to enter a password and enabling functionality only if the password is valid – is straightforward in PowerBuilder. I called this function comparepwd, and it takes an argument that’s the password entered by the user. Depending on the value of the parameter, we could enable different functionality in our application (see Listing 1).

Although it’s obviously a poor approach from a security perspective and very easy to break (you might have used the IDA Pro disassembler already to look at string references from dlls or executables, but here even Notepad [see Figure 1] would be enough to break the “security”), you might have seen this approach in older applications like shareware or demo software; you also might have seen that passwords have been read from a file or database (a scenario common in applications that support multiple users, such as databases or e-mail clients) and have been sent back to the client.

This brings us to the next problem: transmitting a password in clear text. Eavesdropping on a TCP/IP network can be carried out very easily and effectively against protocols that transmit passwords in the clear. To defeat this simple sniffing attack (sometimes I use my iPaq with CENiffer running and check around our network to see if there’s anything wrong with our security in-house), we have to look at something different.

The Crypto API
The Microsoft Crypto API (CAPI) is a core OS service that’s been shipping with Windows since Windows 95 OSR2. It’s the API used by Microsoft to develop applications using cryp-
Algorithm independence: An application can change cryptographic algorithms without having to rewrite the code.

Device independence: Applications written to CAPI can use software- or hardware-based crypto tokens without having to rewrite code.

Single store for certificates and software keys: This allows a user to use the same certificate and key in a variety of applications without forcing the user to export and import them between applications or toolkits.

Built-in support for revocation checking and complex certificate chain building: All the logic is built into the API; applications need only request revocation services or chain building and the OS does the work.

The Microsoft Crypto API can be accessed as a C language API, which makes it easy for us to access it. We simply have to declare the constants and external functions we’d like to use (see Listing 2).

START HASHING YOUR DATA

One of the first things we have to understand if we want to use the crypto API is hashing. A hash is simply a “summary,” or “tag,” that’s generated from a digital document (a binary string) using a mathematical rule or algorithm. It’s designed so that a small change in the document would produce a big change in the hash. As an example I created a hash for two different strings:

Hash for the String, tests
5616C51D2C563E8B5F56B8E47B334
Hash for the String, tests
F20C24485F0D5FABE71B53C4E34B46C

Keep in mind that you could create a hash value from a string or document, but that you cannot get back (in a reasonable amount of time) the string or document from a hash value. A hash is often smaller than the original string (an exception is a small word like the one we used in our example) and, as we can see, it’s generally unreadable by humans. Hashes are also used to check the integrity of files and documents, and are also often used in digital signature algorithms. We could use the hashed value to compare it against the value the user entered (which is hashed through the same algorithm). You might think of writing your own hashing algorithm, but before you start consider what a good hashing algorithm should provide:

• A small change in the document should produce a large change in the hash; notice in our previous example where a tiny change in the string produced two totally different hashes.
• Hashes should not be predictable. In other words, I should not be able to guess that changing the binary string will have a specific effect on the hash.
• Hashes should not collide, and it should be computationally difficult to find collisions. In other words, two binary strings should have an infinitesimal chance of having the same hash, and it should be virtually impossible to find a document that has the same hash as a known document.
• Hashing should be fast. Even if we enhance our application to use hashing, there shouldn’t be a negative impact on our application.

Why not use something that already exists? There are already some good hashing algorithms out there: MD2, SHA, SHA-1, and MD5 are well known (just to name a few). We’ll probably move to SHA-1, which is the U.S. government-approved hashing algorithm, over the next year or two.

At first we call the CryptAcquireContext function, which is used to acquire a handle to a particular key container within a particular cryptographic service provider (CSP) (see Listing 3). This returned handle is used in calls to CryptoAPI functions that use the selected CSP.

The CryptCreateHash function initiates the hashing of a stream of data. We pass as a parameter the hashing algorithm we’d like to use. In Listing 3 we use MD5, but feel free to give another algorithm a try. For example, you might use SHA-1. The function creates a handle to a CSP hash object and returns it to the calling application. This handle is used in calls to CryptHashData to hash session keys and other streams of data.

The CryptGetHashParam function retrieves data that governs the operations of a hash object. We use it to read back the hash value. At the end we convert the hashed password to a hex string, which we display in the MessageBox. We could use this string to verify an entered password.

REMOTE AUTHENTICATION

The two examples you’ve seen so far involve local management and verification of passwords. This is fine, if only your application is involved in checking the password, but this is not often the case in the enterprise. Think of a bigger application where a lot of different back-end systems are involved, every one having its own authentication. Even if we move to more EAServer development in our enterprise where we integrate a lot of different back-end systems, we have to authenticate ourselves against the EAServer components. Even if we use the hashed password approach I described earlier, the remote password model suffers from several new security flaws, unless you can somehow guarantee the security of the connection between the local and remote systems. These flaws are that it transmits the password and the username to the remote object in plain text, and/or it returns the authentication result in plain text.

Let’s think a bit differently. How would it be if clients could authenticate without sending a password? I think it’s certainly hard to steal a password that’s never transmitted, isn’t it?

How do you go about verifying a password without seeing it?

ENCRIPT/DECRYPT YOUR DATA

To tell you the truth, this is nothing new. The approach we’ll be looking at is the same one Windows uses to authenticate users on domains, or Sybase Enterprise Portal uses to authenticate its users. The CryptoAPI CryptDeriveKey function from the cryptographic API generates cryptographic session keys derived from a base data. This function guarantees that when the same cryptographic service provider and algorithms are used, the keys generated from the same base data are identical. The base data can be a password or any other (random) user data. This means that if we use this function with exactly the same base data, we would get exactly the same hashed value on the server side and the client side. On the server side this hashed value could be stored in a database, a password file, or any other system (LDAP).

If a user enters a password, we generate the hashed version and then both the local and remote systems would have exactly the same hashed version of the password.

Now we need a way to compare the two passwords without actually sending the password between the two systems: the client generates a random block of data that it encrypts using a key derived from the hashed password (see Figure 2). The client sends the random
block, along with user identification, to the remote verification computer. That machine retrieves the known hashed password for the specified user and encrypts the random block of data (the same random data that the client used) using a key derived from that hashed password. It then sends the encrypted data back to the client application.

The client application now has two encrypted blocks of data. If the two passwords (the one the client entered and the one that was stored on the server) match, the two encrypted data blocks will match as well, meaning the password is correct and the operation can proceed. Keep in mind that the used random data has to be the same on both the client and the server side. You could use EAServer to generate this random set of data and, just before you start hashing, get this value from a component.

The source code to encrypt a string would look like:

```c
ls_encrypted = lnv_crypt.EncryptData("my sensitive data", "SecretKey")
```

where `ls_encrypted` is the encrypted data we would pass to the server.

To decrypt a string on the remote computer we would use:

```c
ls_decrypted = lnv_crypt.DecryptData(is_crypted, "SecretKey")
```

where `ls_decrypted` holds the original data.

The functions could be implemented in one function (see Listing 4). The only difference is, if we want to encrypt or decrypt:

```c
of_encryptdecrypt(String data, String password, boolean encrypt)
```

Unless you're using an application that requires a lot of bandwidth, most computers can easily handle encrypting all data transferred over the network without causing major performance problems. If you're using a high-bandwidth application, you might want to consider purchasing a crypto accelerator card for your machine to allow it to encrypt all of its data, for example.


**Conclusion**

In its simplest form, password protection is highly insecure because it introduces fairly easy ways for unauthorized users to obtain valid passwords. Cryptography is the extra ingredient you need to secure functionality in your applications. We've seen how to add basic password protection to PowerBuilder applications and how to use Microsoft's Crypto API to make that protection secure, whether you implement local or remote password authentication.

The challenge/response password model improves security dramatically by eliminating the need to transmit the password in any form between the local and remote systems. Instead, your program uses the password to derive an encryption key that's used to encrypt a random block of data. If both the client and server obtain the same result when encrypting the random data with the password they have, the passwords must match.

---

**Listing 1**

```c
Boolean lb_PasswordValid
Choose Case as_password
Case "Password1"  
  lb_PasswordValid = TRUE
Case "Password2"  
  lb_PasswordValid = TRUE
Case else  
  lb_PasswordValid = FALSE
End Choose
IF lb_PasswordValid THEN  
  /* enable functionality */  
ELSE  
  MessageBox("Error", "Invalid Password")
END IF
return lb_PasswordValid
```

**Listing 2**

```c
Function ulong CryptAcquireContext &
  (Ref ulong phProv, &
   String pszContainer, &
   String pszProvider, &
   ulong dwProvType, &
   ulong dwFlags) Library "advapi32.dll" Alias For
"CryptAcquireContextA"
```

```c
Function ulong CryptGetProvParam &
  (ulong hProv, &
   ulong dwPara, &
   Ref Blob pbData, &
   Ref ulong pdwDataLen, &
   ulong dwFlags) Library "advapi32.dll" Alias For
"CryptGetProvParam"
```

```c
Function ulong CryptCreateHash &
  (ulong hProv, &
   ulong Algid, &
   ulong hKey, &
   ulong dwFlags, &
   Ref ulong phHash) Library "advapi32.dll" Alias For
"CryptCreateHash"
```

```c
Function ulong CryptHashData &
  (ulong hHash, &
   ref String pbData, &
   ulong dwDataLen, &
   ulong dwFlags) Library "advapi32.dll" Alias For
"CryptHashData"
```

```c
Function ulong CryptDeriveKey &
  (ulong hProv, &
   ulong Algid, &
   ulong hBaseData, &
   ulong dwFlags, &
   Ref ulong phKey) Library "advapi32.dll" Alias For
"CryptDeriveKey"
```

```c
Function ulong CryptDestroyHash &
  (ulong hHash) Library "advapi32.dll" Alias For
"CryptDestroyHash"
```

---

**AUTHOR BIO**

Berndt Hamboeck is a senior consultant for BHITCON. He's a CSI, SCAPC8, EASAC, and SCJP2 and started his Sybase development using PBS.
Sybase, Inc

www.sybase.com/pbextension
Function ulong CryptEncrypt &
(ulong hKey, &
ulong hHash, &
ulong Final, &
ulong dwFlags, &
Ref string pbData, &
Ref ulong pdwDataLen, &
ulong dwBufLen) Library "advapi32.dll" Alias For
"CryptEncrypt"

Function ulong CryptDestroyKey &
(ulong hKey) Library "advapi32.dll" Alias For
"CryptDestroyKey"

Function ulong CryptReleaseContext &
(ulong hProv, &
ulong dwFlags) Library "advapi32.dll" Alias For
"CryptReleaseContext"

Function ulong CryptDecrypt &
(ulong hKey, &
ulong hHash, &
ulong Final, &
ulong dwFlags, &
Ref string pbData, &
Ref ulong pdwDataLen) Library "advapi32.dll" Alias For
"CryptDecrypt"

Function Long CryptGetHashParam &
{ulong hhash, &
ref Blob pbData, &
ref ulong pdwDataLen, &
ulong dwFlags} Library "advapi32.dll" Alias For
"CryptGetHashParam"

Listing 3

string Password = "testpwd"
Blob lblb_hash
ulong lul_bloblen = Len(lblb_hash)

String ls_hashed
Char lc_hash[]
Long ll_from = 1 to Long(lul_bloblen)

ls_hashed = Left(String(lblb_hash), lul_bloblen)

Listing 4

ulong hCryptProv
ulong iLength
String sTemp
ulong hHash
ulong hKey
String ls_ret

if Encrypt Then
lLength = Len(Data)
sTemp = Data
End If

MessageBox("DEBUG", "Error during CryptCreateHash!")
End If
Sybase

www.sybase.com/powerbuilder
The PowerBuilder Integrated Development Environment

UNDERSTANDING THE IDE

For the new book PowerBuilder: Advanced Client/Server Development, Bill Green has contributed a chapter dedicated to the PB9 IDE. The IDE has not changed much from PB8 so he not only covers the new features but also describes important historical details about the IDE and provides tips on the best ways to use it. The following is a small portion of the chapter that I think you'll find educational.

In this chapter I highlight the changes in the IDE from version 8 to 9, and also dig into the IDE and detail some of the productivity improvements that are available, but not generally used.

The IDE is a powerful development tool as it puts all kinds of information at your fingertips, but sometimes the amount of information exceeds its value and becomes clutter that results in developers trimming the IDE capabilities to an inferior level. When you're done with this chapter, you should be able to take advantage of the many capabilities that are hidden just beneath the surface.

IDE History

PowerBuilder helped coin the term Integrated Development Environment (IDE). An IDE means you don't have to go outside the environment to perform tasks related to the tool's function. As an example, if you wrote some Java code using Notepad, you would have to switch to something else to compile the Java code. An IDE integrates the compiler and it's a function of the tool to compile the code written using that tool. When developing client/server applications with PowerBuilder, you don't have to go outside the PowerBuilder environment to compile, set up data access, etc. It's not that you can't go outside the tool, rather you don't have to.

PowerBuilder's IDE did not have any major changes through PB6. Sure there were some changes in menu options, menu icons, and additional tools and capabilities, but the IDE remained pretty static, except for the debugger, which underwent a major facelift with PB6. In PB6, the debugger incorporated changes that allowed developers to see more data at one time. Distinct sets of data were separated into views that could be combined into a viewing area using a tabbed metaphor. This gave developers quick access to all the information available to the debugger. It was also very flexible, allowing developers to set up the debugger appearance any way they saw fit. The combination of maximum flexibility and maximum data at the same time was quite revolutionary then.

THE PB6 DEBUGGER

Many things were going on at the time PB6 was released. The PB6 debugger was receiving positive feedback regarding the data and flexibility it provided. Another product, PowerSite, was also getting some good press with its interface, which was similar in style and concept. There was also some demand to add Web-building capabilities to PowerBuilder and surrounding all of that was the requirement for the PowerBuilder IDE to change from a modal, single-threaded development environment. PB7 was intended to resolve these requirements. It began with the molding of the development environment to match the debugger environment – display maximum data with maximum flexibility. The main purposes were threefold:

1. Eliminate modal dialogs.
2. Give developers access to as much data as they want.
3. Establish a highly customizable development environment.

Another targeted feature was building a foundation for combined IDEs. At the time, several different IDEs were available to perform different tasks, such as HTML and Java development. Each product had its own IDE, and each IDE had its own set of features. The problem was that having multiple “Integrated Development Environments” defeated the purpose, so the effort was made to standardize an IDE feature set that would be common to the three in order to move toward having one truly “integrated” development environment. What we have today is the result of that effort: developers can truly develop PowerBuilder, HTML, or Java code (at least JSP code) within the one environment.

The incorporation of common IDE features began in PB7, and the IDE began to take on its current form. The modal windows were replaced with a flexible docked window display. This approach meant that developers could view multiple sets of data relating to the object/context they were working in, but it also meant that the more data you wanted to see at one time, the less you could see of any particular set of data; there was simply a limited amount of screen real estate to go around. The IDE and the debugger now supported the same feature set, and the end result was that developers either liked it or hated it.
PowerBuilder 8 was the next step forward in the IDE's technology. The main elements brought into the PB8 environment were workspaces and targets; support for non-PowerBuilder code, and enhanced Source Code Control integration. Let's quickly cover the highlights.

Source Code Control, or SCC integration, was improved to the point that the SCC implementation in PowerBuilder 8 was now identical to that of other development products. Instead of multiple copies of PowerBuilder Libraries (PBLs) in both the SCC Repository location and the developer's workspace, the PBLs resided only on the developer's workspace. The SCC Repository contained only the objects within the applications and in a flat-file format. This opened the door for more extended use of standard SCC features such as branching and building configuration management. This was a big step forward, but was still missing some important SCC capabilities from within PowerBuilder, such as the ability to rebuild the developer workspace based on the SCC Repository, or intelligently manage a nightly build process.

Workspaces and targets introduced PowerBuilder developers to the concept of working on more than one application at a time, and even more than one type of application at a time. An application that comprised a Web site, some components for a middle-tier application server, and some standard client/server executables could now easily be configured in a developer's workspace giving the developer instant access to all the objects.

The PB8 IDE

PowerBuilder 8 has continued the process, and while it does not incorporate major changes to the IDE, it does address some additional needs, including enhanced SCC integration and the addition of some new targets, such as the JSP target. These features do not affect the IDE specifically.

Let's take a closer look at the IDE and its various components and see where all the elements are and how you can take advantage of the IDE's flexibility to enhance your productivity rather than detract from it.

Workspaces and Targets

As previously mentioned, PB8 introduced the concept of workspaces and targets to the PowerBuilder developer. Simply put, a workspace is a collective area where you can maintain and manipulate several targets. For example, a PowerBuilder application, a Web site, or a collection of EServer components might be considered targets, but if they all pertain to the same overall system, they can be combined within a single workspace, giving developers a more coherent and complete picture of their system.

Workspaces

The workspace concept eliminated the single application limitation. Developers can have multiple application targets within their workspace and direct access to all objects across all targets within the given workspace. Application switching is now a thing of the past.

At the workspace level there are a set of commands available to the developer. To access the Workspace Menu, position your pointer to the workspace, click the right-mouse button (RMB), and a set of options is presented.

Workspace Menu

The menu options are context-sensitive, so some options are enabled or disabled depending on the situation. For example, SCC options are not shown if you're not using some sort of SCC. Let's look at the menu options available and see what each one means.

New

This command advances you to the Create New Workspace capabilities within PowerBuilder. The workspace file is a simple text file and is fairly irrelevant by itself. Where you place the workspace file, however, can have an impact on your development environment. All paths within the workspace, be they target paths, PBL paths, or other files, are stored relative to the workspace location, so it's recommended that the workspace be located at the root level of the System Development Environment. I've gone so far as to redesign my system structure, in terms of the directories and subdirectories I have where the code is stored, to more comfortably support the workspace I'm working in. My development
environment structure looks something like Figure 2.

- **Sample Development Environment**
  The structure is pretty simple – the top level (work environment) is the root. Beneath that is a company identifier – I am a consultant and work with several clients, therefore this separation is needed – followed by an application identifier, then a version structure containing the elements of the application. My workspace is set up at the root level as it encompasses all the applications. Each target is created at the application/development level. Note that there may be multiple targets within an application as there may be a client/server application, middle-tier components, and/or Web components. There may also be several versions of the same application, and I give each of these a target. I can then assemble a workspace to include however many targets and versions of targets that I need to accomplish any given task.

**Add Target**
This menu option allows you to add an existing target to your one, perhaps one that’s already been created. The target file is also a text file and is identified by a .PBT extension.

**Open Workspace**
This option allows you to open another workspace, replacing the workspace you currently have open. This indicates that you can only be working in one workspace at a time. Another thing to remember is that you have access to all objects within your workspace, but you also don’t have access to objects outside your workspace.

**Incremental Build**
We all know what an incremental build is – the process of regenerating any objects that have changed since the last build, including referenced objects. At the workspace level, this means that each target within the workspace will execute an incremental build instruction in sequence.

**Full Build**
Like the incremental build, the full build at the workspace level will simply execute a full build instruction for each target in the workspace, consecutively. This is an excellent option for performing a full rebuild on everything in your workspace prior to shipping the code to your QA group.

**Deploy**
The deploy option is a new feature added in PB8. It allows you to execute all project objects in your workspace in the sequence you’ve defined. The sequence is defined by the sequence of targets and the sequence of projects within targets. At the target level, as we’ll see later, you can modify the sequence of the projects to be executed. In this manner, you can control exactly how the objects in your workspace should be deployed, for example:
1. Deploy a component to EAServer.
2. Create proxies for the component in EAServer.
3. Deploy another component using the generated proxy.
4. Create proxies for that component.
5. Deploy a client/server executable that calls the components in EAServer using the proxies generated.

In this manner, you have complete control over the deployment sequence and can deploy everything in the workspace with a single mouse click.

**Debug**
The debug option at the workspace level will run the debugger on the currently selected execution target. If a target is not currently selected, it will offer a dialog window where you can set the target to debug.

**Run**
Likewise, the Run command will run the currently selected target. Note that both the Run and Debug options will only work if the target is a component that can be run, for example, a GUI application. Also note that the Current Run/Debug target is not necessarily the target you currently have open. More on this later in the Targets section.

**Close**
This will close the current workspace.

**Show**
This option allows you to specify which types of objects can be shown, such as targets, PBLs, and applications.

**Properties**
This option provides access to workspace properties. Upon selection, a dialog window is opened that provides access to three areas of workspace properties: the Target Sequence List, a Deployment Preview presentation, and Source Code Control setup options.

**THE WORKSPACE PROPERTIES DIALOG**
Under targets, you’ll see a list of the targets in the workspace. The sequence of targets can be altered here using the up/down arrows. The checkbox indicates whether the target is included in deployment instructions.

**THE DEPLOY PREVIEW DIALOG**
In the Source Code Control tab, you can set various options for Source Code Control management. The first option is the type of SCC you’re using and the options here will generally include PBNative and whatever third-party SCC providers you have installed. The UserID specifies your identifier ID as it relates to Source Code Control. The Project option is expandable and the expanded options will vary by SCC Provider, but its main purpose is to establish where the SCC Repository is located. The Local Root Directory points to where your workspace object can be found. Then, several options allow you to control how you want SCC to operate.

**Source Code Control Options**

**TARGETS**
A target is defined as the type of deployment you are planning for a particular set of data, for example, if you have a set of code you plan to deploy as a PowerBuilder executable, which would be considered a PowerBuilder target. Similarly, a collection of Web components designated for building a Web site would be considered a Web target. It’s simply a way to separate the different deployment characteristics of your overall application. PowerBuilder now supports multiple targets within a workspace so you could have a complete system including multiple PB executables, EAServer components, and Web components all within the same workspace. This simplifies both the development and maintenance efforts required.

To quickly view which targets you can create, choose the menu option File | New and then the Targets tab. This will depict the various types of targets available (see Figure 4).

**PowerBuilder Targets**
The target options are as follows:

**Application**
The Application target creates a new application in a PBL. It does not create any additional PB objects.

**TemplateApplication**
For more sophisticated application generation, the template application allows you to create the prerequisite elements for one of three types of applications:
1. **SDI**: Single Document Interface applications are based on a single controlling window.
TARGET OPTIONS FOR POWERBUILDER TARGETS

New...

This option takes you to the New Target panel discussed earlier so you can create a new target in your workspace.

Search

This option allows you to perform a search across all objects within the target.

Incremental Build

Like the workspace option, the incremental build option allows you to incrementally rebuild all objects within the specified target.

Full Build

Like the incremental build, the full build at the target level will simply execute a full build instruction for all objects in the target consecutively. This is an excellent option for performing a full rebuild on everything in your workspace prior to shipping the code to your QA group.

Migrate

This menu option allows you to migrate a single target. While, generally, migration is done when creating a target, and PowerBuilder takes care of informing you when a migration is needed/will happen, it’s useful to know that you can perform the migration yourself when necessary.

Deploy

The deploy option is a new feature added in PB8. This option allows you to execute all project objects in your target in the sequence you have defined. The sequence is defined as the sequence of projects within targets. At the target level, you can modify the sequence of the projects to be executed.

Debug

The debug option at the target level will run the debugger on the currently selected execution target. If a target is not currently selected, it will offer a dialog window where you can set the target to debug.

Run

Likewise, the Run command will run the currently selected target. Note that both the Run and Debug options will work only if the target is a component that can be run, for example, a GUI application. Also note that the Current Run/Debug target is not necessarily the target you currently have open.

Remove Target

This will remove the currently selected target from the workspace. It does not delete any files.

Show

This option allows you to specify what types of objects can be shown, such as targets, PBLs, applications, etc.

Properties

This option provides access to target properties. Upon selection, a dialog window is opened that provides access to a dialog window with two tabs (for a PowerBuilder target). The first of these is your application library list and the second is the deployment sequence. Figure 5 shows the library list options. It’s interesting to note that the library list is now located here instead of in the application object.

THE TARGET PROPERTIES DIALOG SHOWING THE APPLICATION LIBRARY LIST

The second portion of this window is the Deploy tab. This tab allows you to control which projects within the target should be run and in what sequence. Any changes here will affect the deployment sequence of both the target, when deploying just the target, and the workspace.

Target Deployment Options

This may not seem a big deal in a typical client/server application, but it can be a major contributor in multitarget settings such as an EAServer component-based development effort.

Conclusion

There are a lot of features in the PowerBuilder IDE that often go unused and sometimes even undiscovered. In Bill’s chapter he points out those features and describes how to use them. The IDE is something you really need to explore in order to exploit its capabilities. The chapter will give you a head start into understanding the IDE, its philosophies and also some of its idiosyncrasies. With this understanding, you will be better prepared to take advantage of an environment that is actually full of possibilities, and positioned well for future expansion.

AUTHOR BIO

Bill Green is a member of TeamSybase and coauthor of several PowerBuilder books including the forthcoming PowerBuilder 9 series of tools.

wgreen@power3.com
Dynamically creating (or destroying) objects within a DataWindow has many advantages such as:

- Dynamically changing the content
- If a printed DataWindow varies in appearance from its visual presentation

The syntax for creating objects within DataWindows can be daunting; no wonder it’s not used that often. Before I go into more detail, it’s important to know how objects are contained in a DataWindow in the first place. Once you understand this, you will find that dynamically creating objects is easy.

**The Naked DataWindow**

**BACKGROUND**

It should come as no surprise that a DataWindow is really just a collection of objects, each with its own properties. When a programmer is creating a DataWindow via the DataWindow painter, he or she is actually just using a graphical IDE to create and set the properties of the objects that make up the DataWindow. Actually, the naked DataWindow is not graphical at all but exists in text format. It’s just that most programmers tend to create and edit DataWindows via the DataWindow Painter.

Prior to PowerBuilder 8, if you wanted to take a peek at what a DataWindow looked like in its text format, it would have to be exported via the library painter, then opened up within a text editor. At that point, changes could be made and the text file could be imported back into PowerBuilder. As the file was being imported, PowerBuilder would regenerate it, making sure your hack was syntactically correct.

Since PowerBuilder 8, Sybase has allowed programmers to directly modify objects via a Source Editor, effectively putting an end to the enjoyable, unsupported DataWindow source hacking days. By viewing the source code of an existing DataWindow, you’ll appreciate the syntax of creating objects dynamically. A basic understanding of the DataWindow syntax can help a lot. Let’s start with the DataWindow illustrated in Figure 1.

**Understanding the Syntax**

If this simple DataWindow is opened up in the Source Editor, you’ll see all the objects that it’s comprised of. The complete syntax is extremely lengthy and cannot be listed in its entirety here. At first glance, the syntax looks foreboding, but after further inspection, it becomes more familiar. If you think about it, it looks almost identical to the syntax for the Describe and Modify function. That’s because it is the syntax for Describe and Modify.

The source code can be broken down into six categories:

1. Version information
2. DataWindow properties
3. Band properties
4. Source definition
5. Object definitions
6. DataWindow HTML/XML properties

**VERSION INFORMATION**

Let’s take a closer look at our Naked DataWindow:

```
release 9;
```

The first line of the syntax comprises only one statement indicating the PowerBuilder release with which this DataWindow object was constructed. This line will contain only major release numbers (you won’t see 9.01). The release number is important as it tells the DataWindow Engine how to handle the rest of the syntax. Obviously, more recent DataWindow versions contain added features. If you’re in PowerBuilder 8 and try to open a DataWindow that was built in PowerBuilder 9, PowerBuilder gives the error message “DataWindow Syntax has incorrect release number.” On the other hand, a more recent version of PowerBuilder will happily import a DataWindow created in an earlier release. When an earlier version of a DataWindow is saved or regenerated, it’s migrated to the current version.

If you’re resourceful, the DataWindow may be migrated backward by changing its release number. It may take a bit of trial and error to remove any of the source code that may not be understood by previous DataWindow Engines.

**DATAWINDOW PROPERTIES**

```
datawindow(units=0 timer_interval=0
color=12632256 processing=0 HTMLDW=no
print.printername="" print.document-name="" print.orientation = 0
print.margin.left = 110 print.margin.right = 110 print.margin.top = 96
print.margin.bottom = 96
print.paper.source = 0
print.paper.size = 0 print.cansusedefaultprinter=yes print.prompt=no
```

**FIGURE 1** Simple tabular DataWindow
Millions of Linux Users
One Magazine

There is no escaping the penetration of Linux into the corporate world. Traditional models are being turned on their head as the open-for-everyone Linux bandwagon rolls forward.

Linux is an operating system that is traditionally held in the highest esteem by the hardcore or geek developers of the world. With its roots firmly seeded in the open-source model, Linux is very much born from the "if it's broke, then fix it yourself" attitude.

Major corporations including IBM, Oracle, Sun, and Dell have all committed significant resources and money to ensure their strategy for the future involves Linux. Linux has arrived at the boardroom.

Yet until now, no title has existed that explicitly addresses this new hunger for information from the corporate arena. Linux Business & Technology is aimed squarely at providing this group with the knowledge and background that will allow them to make decisions to utilize the Linux operating system.

Look for all the strategic information required to better inform the community on how powerful an alternative Linux can be. Linux Business & Technology will not feature low-level code snippets but will focus instead on the higher logistical level, providing advice on hardware, software, through to the recruiting of trained personnel required to successfully deploy a Linux-based solution. Each month will see a different focus, allowing a detailed analysis of all the components that make up the greater Linux landscape.

Regular features will include:
- Advice on Linux Infrastructure
- Detailed Software Reviews
- Migration Advice
- Hardware Advice
- CEO Guest Editorials
- Recruiting/Certification Advice
- Latest News That Matters
- Case Studies

Linux Business and Technology

FOR ADVERTISING INFORMATION:
CALL 201 802.3020 OR
VISIT WWW.SYS-CON.COM

Regular annual cover price $71.76
YOU PAY ONLY $49.99
12 ISSUES/YR
*OFFER SUBJECT TO CHANGE WITHOUT NOTICE

SUBSCRIBE TODAY!
WWW.SYS-CON.COM
OR CALL 1-888-303-5282

SAVE 30% OFF!
After the PowerBuilder release number comes the DataWindow properties, such as the color and print information. Note the new DataWindow features for 9.0, such as hidegrayline, in the above source code. Another DataWindow property that’s worth pointing out is the processing property. This specifies the DataWindow’s Presentation Style:

- 0 - (Default) Form, Group, Query, Tabular, N-UR Label
- 1 - Grid
- 2 - Label
- 3 - Graph
- 4 - Crosstab
- 5 - Composite
- 7 - RichText

The next time you want to change the style of a DataWindow, there’s no need to re-create the entire DataWindow. Just change the processing property in the source code. Also at runtime, using dot notation or Describe, the “processing” attribute can be used to determine the DataWindow style. You’re not allowed to change the DataWindow Presentation Style at runtime.

**BAND PROPERTIES**

```plaintext
header(height=256 color="536870912")
summary(height=92 color="536870912")
footer(height=0 color="536870912")
detail(height=68 color="536870912"
      height.autosize=yes)
```

The Band Properties section consists of one statement for each band in the DataWindow. It describes the properties of each band; for example, its height, color, and any expressions it may have. Actually this section is not mandatory, as PowerBuilder will create these four bands even if you don’t specify that it do so. If the band properties are not supplied, they’ll be created with a height of zero. If your DataWindow contains groups, they won’t be specified here. Group “bands” in PowerBuilder are specified elsewhere in the source code.

**SOURCE DEFINITION**

The source code in Listing 1 has been cosmetically aligned for readability purposes. It’s divided into two sections. The first section describes the result set, specifically:

- Data types
- Update characteristics
- Database column names
- Default values

The type property defines the PowerBuilder data type for the column. This property can be changed whenever PowerBuilder fails to correctly determine the data type of a database column. This often happens when PowerBuilder is working with less common data types and time stamps.

The second portion of Listing 1 specifies the SQL source, including any PowerBuilder-defined retrieval argument. This section also describes the SQL that will generate the result set. The SQL source is actually stored internally in a generic PowerBuilder dialect called PBSELECT. This is how the SQL gets generated when the SQL statement is “Painted”. If you choose the “Convert to Syntax” option and type in the SQL statement, PowerBuilder stores the statement and standard SQL:

```plaintext
retrievel ~"employee~",~"emp_lname~",~"emp_fname~",~"emp_address~",~"emp_city~",~"emp_state~",~"emp_zip~",~"emp_phone~",
FROM ~"employee~"
WHERE ~"employee~"."state~" = 'TX'
ORDER BY ~"employee~"."emp_lname~" ASC
*
)
```

**OBJECT DEFINITIONS**

This section contains all the other objects in the DataWindow. It contains important information as to which band each object belongs to. Objects such as columns, text objects, computed fields, and drawing objects are found here. Listing 2 provides the object definitions for a column, text, and line object.

Notice that the code in the listing is literally a help file to see which properties belong to which objects.

**DATAWINDOW HTML/XML PROPERTIES**

This final section contains all the HTML/XML properties that are associated with the DataWindow. Many of these are new to PowerBuilder 9.0.

```plaintext
htmltable(border="1")
htmlgen(clientevent="1" clientvalidation="1" clientcomputedfields="1"
    clientformatting="0" scriptable="0" generatejavascript="1"
    encodeselflinkargs="1" netscapestyle="0"
    export.xml(headgroups="1" includewhitespace="0" metadatastype=0
    savemetadata="0")
```

**Creating Dynamic Objects**

Creating a dynamic DataWindow looks like a daunting task. But as you can see, if you know what a naked DataWindow looks like, the job becomes much easier.

**Summary**

Creating dynamic DataWindows is much easier. After the PowerBuilder release number comes the DataWindow properties, such as the color and print information. Note the new DataWindow features for 9.0, such as hidegrayline, in the above source code. Another DataWindow property that’s worth pointing out is the processing property. This specifies the DataWindow’s Presentation Style:

- 0 - (Default) Form, Group, Query, Tabular, N-UR Label
- 1 - Grid
- 2 - Label
- 3 - Graph
- 4 - Crosstab
- 5 - Composite
- 7 - RichText
LEARN WEB SERVICES. GET A NEW JOB!

Get Up to Speed with the Fourth Wave in Software Development

Only $69.99 for 1 year (12 issues)*
* Newsstand price $83.88 for 1 year

Subscribe online at www.wsj2.com or call 888 303-5252
*Offer subject to change without notice

SYS-CON Media, the world’s leading i-technology publisher of developer magazines and journals, brings you the most comprehensive coverage of Web services.

SUBSCRIBE TODAY TO THE WORLD’S LEADING WEB SERVICES RESOURCE

• Real-World Web Services: XML’s Killer App!
• How to Use SOAP in the Enterprise
• Demystifying ebXML for success
• Authentication, Authorization, and Auditing
• BPM - Business Process Management
• Latest Information on Evolving Standards
• Vital technology insights from the nation’s leading Technologists
• Industry Case Studies and Success Stories
• Making the Most of .NET
• Web Services Security
• How to Develop and Market Your Web Services
• EAI and Application Integration Tips
• The Marketplace: Tools, Engines, and Servers
• Integrating XML in a Web Services Environment
• Wireless: Enable Your WAP Projects and Build Wireless Applications with Web Services!
• Real-World UDDI
• Swing-Compliant Web Services
• and much, much more!

www.PowerBuilderJournal.com

Listing 1:

table {
    column=(type=char(20) updatewhereclause=yes
    name=emp_lname dbname="employee.emp_lname"
} column=(type=char(20) updatewhereclause=yes
    name=emp_fname dbname="employee.emp_fname"
} column=(type=decimal(3) updatewhereclause=yes
    name=salary dbname="employee.salary"
}

retrieve="PRSELECT( VERSION(400) TABLE(NAME="employee"
) COLUMN(NAME="employee.emp_lname")
COLUMN(NAME="employee.emp_fname")
COLUMN(NAME="employee.salary")
WHERE EXP1 =~ "employee."---"state"== OP
  EXP2 =~ "TX"-)
ORDER(NAME="employee.emp_lname" ASC=yes ) * }

Listing 2:

text(band=header alignment="0" text="Emp FName" border="0" color="33554432" width="576" html.valueishtml="0" name="emp_fname_t visible="1" font.height="8" font.weight="600" font.family="Arial" font.charset="0" background.color="536870912"

column(band=detail id=1 alignment="0" tabsequence=32766 border="0" color="33554432" width="576" format="#0" html.valueishtml="0" name=emp_lname visible="1" edit.autoselect=yes edit.limit=20 edit.case=any edit.autoscroll=yes font.family="2" font.weight="400" font.charset="0" background.mode="1" background.color="536870912"

line(band=header x1="73" y1="248" x2="1559" y2="248" name=l_1 visible="1" pen.color="33554432" background.mode="2"

compute(band=header expression="'Page ' + page() + ' of ' + pageCount()"
border="0" color="33554432" x="1157" y="24" width="466" html.valueishtml="0" name=page_1 visible="1" font.family="2" font.weight="10" font.charset="0" background.color="1073741824"

Listing 3:

dw_1.Modify("create compute(band=header alignment="1" expression="'Page ' +
    page() + ' of ' + pageCount()" *border="0" color="33554432" x="1157" y="24" width="466" format="#0" html.valueishtml="0" name=page_1 visible="1"
    font.family="2" font.weight="10" font.charset="0" background.color="1073741824"

Listing 4:

dw_1.Modify("create compute(band=header alignment="1" expression="'Page ' +
    page() + ' of ' + pageCount()" *border="0" color="33554432" x="1157" y="24" width="466" format="#0" html.valueishtml="0" name=page_1 visible="1"
    font.family="2" font.weight="10" font.charset="0" background.color="1073741824"

Listing 5:

dw_1.Modify("destroy compute(band=header alignment="1" expression="'Page ' +
    page() + ' of ' + pageCount()" *border="0" color="33554432" x="1157" y="24" width="466" format="#0" html.valueishtml="0" name=page_1 visible="1"
    font.family="2" font.weight="10" font.charset="0" background.color="1073741824"

The code listing for this article can also be found at www.sys-con.com/pbdj
The problem with this approach is that it's too hard coded; changes to the hierarchy (for example, adding a new level between two existing levels) involve significant changes to the database and to TreeView controls. In this article I'll look at a more generic approach that generates considerably fewer modifications when a hierarchy changes, as well as making complex queries against a hierarchy much simpler.

**Geographical Hierarchy**

I was part of a team that used this approach while developing an application for high street bank. The bank had a network of branches organized hierarchically. At the top was the bank and beneath it were regions, then cities, and then the branches. Figure 1 shows the hierarchy displayed in a TreeView.

Traditionally, you would store this hierarchy in a number of tables: BANK, REGION, CITY, and BRANCH, with each child table linked to its parent via a foreign key. The code in your TreeView has to determine which are the appropriate tables depending on the level the user is currently exploring. This approach works adequately in the short term, but what if, in the future, the bank decides to introduce a new level into the hierarchy, for example, a DISTRICT level between REGION and CITY? You need to redesign the data model by adding a DISTRICT table and replumbing the referential integrity between REGION and DISTRICT, and DISTRICT and BRANCH. Then you need to modify the code in the TreeView to accommodate the new structure. This is a lot of work. How can you store a hierarchy in a more generic way that considerably reduces the amount of database and application changes when the hierarchy changes?

**A Generic Solution**

The answer is to store the hierarchy items in one table (we'll call it ITEM in this example) and the hierarchy structure in another table linked to it (ITEM_HIERARCHY). ITEM_HIERARCHY is a self-referring table that relates child items to their parent using a column (parent_id) that refers back to the table's primary key (item_id). Listing 1 shows the bank hierarchy represented in such a table. The code in the TreeView becomes much simpler when written against this table. Previously, the SQL to retrieve the child items of an expanded item depended on the expanded item's level (for example, the child items come from the REGION table if the expanded item is at level 1 or the CITY table if it's at level 2). Now, the SQL:

```
SELECT item_id, item_name
FROM   item_hierarchy
WHERE  parent_id = :parent_id
```

(where :parent_id is the primary key of the expanded item) retrieves the child items irrespective of the expanded item's level.

It's now much easier to alter the structure of the hierarchy. Introducing a DISTRICT level between REGION and CITY is simply a matter of adding and updating data in the ITEM and ITEM_HIERARCHY tables.

**Storing Additional Information**

The ITEM table stores the bare minimum of information about an item: its name. This is often sufficient; for example, when an item is at a hierarchy level, that's simply a way of grouping the items on the level below. (For example, the bank might not be interested in regions in their own right, but going straight from the bank level to the city level in the TreeView would fill the display with too many items. For convenience, cities are grouped into regions.) If, on the other hand, you need to store additional information about an item, add an additional table. For example, to store cities' populations, add a CITY_ATTRIBUTE table with the following structure:

```plaintext
<table>
<thead>
<tr>
<th>city_id</th>
<th>population</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>2000000</td>
</tr>
<tr>
<td>6</td>
<td>3000000</td>
</tr>
</tbody>
</table>
```

The city_id column joins to the item_id column in ITEM. Those readers with data warehouse experience will note that this data model vaguely resem-
bles a snowflake schema, with ITEM_HIERARCHY as the “fact” table and “dimensions” linked to it. Figure 2 shows an example of such a schema.

Navigating the Hierarchy

What if you need to query the hierarchy? For example, the bank TreeView had an “Expand All” option that expanded a TreeView item and all its child items at all levels. Many items had enough child items to cause a delay between the user expanding the item and all its expanded children appearing in the TreeView. To give the user a visual cue that something was happening, we added a progress bar. But to prime a progress bar, you need to know the number that represents “task complete” (in this case the total number of child items). Calculating this number using the ITEM_HIERARCHY table would involve “walking” the hierarchy. Oracle has the CONNECT BY clause to do this, but other DBMSs do not.

However, you can greatly simplify traversal of the hierarchy by creating another table (we’ll call it HIERARCHY_ACCESS) that stores the hierarchy in a different format. Listing 2 shows the HIERARCHY_ACCESS table for the bank hierarchy. For example, this table tells us that the Piccadilly branch has three parents: Manchester at level 3, North West at level 2, and High Street Bank at level 1. Using this table, a query such as “return the total number of child items beneath North West” is simple:

```
SELECT count (*)
FROM   hierarchy_access
WHERE  parent_id = 2
```

You can also answer more complicated questions with simple and quick-executing queries written against this table. What if you want to identify all the branches that are in the same region as Liverpool, or the number of cities that are in the same region as the Portswood branch? Listing 3 shows the SQL for these queries.

Maintaining the HIERARCHY_ACCESS Table

Clearly, the data in HIERARCHY_ACCESS is derived from the data in HIERARCHY_ITEM. Synchronize the two tables using triggers. Whenever a user deletes an item from HIERARCHY_ITEM, the delete trigger simply cascades the delete down to HIERARCHY_ACCESS. The insert trigger is slightly more complex and is shown (written for SQL Server 2000) in Listing 4.

Note: You can quickly check that the tables are in synch by running the queries:

```
SELECT SUM (item_level - 1)
FROM   item_hierarchy
and
SELECT COUNT (*)
FROM   hierarchy_access
```

Because each item in the hierarchy has (item_level - 1) parents, the two queries should always return the same number.

More Flexibility for the User

As well as making a hierarchy easier to maintain and change for developers, the generic hierarchy approach potentially gives users more flexibility. For example, I’ve written my own simple contact management program using a generic hierarchy. I did this because I was frustrated by the limitations of commercial programs. Often you add a contact to a category (for example, Friends, Colleagues) giving you a simple, fixed two-level hierarchy. But this is not how my mind works. I like to be able to create a hierarchy with an arbitrary number of levels, for example, Friends/London or Paris/Tradesmen/Plumbers. In the application TreeView I can add any number of groups before adding a contact.

In addition, the generic hierarchy allows an item to have more than one parent. In my contact management application I can include the same contact in several different places – Paris/Tradesmen/Plumbers or Tradesmen/Paris/Plumbers, or Apartment/Contacts. I find that this considerably reduces “Where did I put that?” frustration and allows me to avoid using the application’s search function, because I’ve covered all the possible paths through the hierarchy that my untidy mind is likely to take.

All of this is simply programmed against a generic hierarchy; it would clearly be impossible against a standard hierarchical data model.

Summary

This article showed how to store a hierarchy in a relational database in a way that significantly reduces coding changes when the configuration of the hierarchy changes, and makes complex queries against the hierarchy simple and quick to execute. The approach also allows for applications to provide richer, more flexible functionality to users. The solution involves a slightly nonstandard data model, but the benefits of this over the traditional hierarchical data model are clear.

“I like to be able to create a hierarchy with an arbitrary number of levels, for example, Friends/London or Paris/Tradesmen/Plumbers”
Listing 1:

ITEM table

<table>
<thead>
<tr>
<th>item_id</th>
<th>item_name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>High Street Bank</td>
</tr>
<tr>
<td>2</td>
<td>North West</td>
</tr>
<tr>
<td>3</td>
<td>North East</td>
</tr>
<tr>
<td>4</td>
<td>South East</td>
</tr>
<tr>
<td>5</td>
<td>South West</td>
</tr>
<tr>
<td>6</td>
<td>Manchester</td>
</tr>
<tr>
<td>7</td>
<td>Liverpool</td>
</tr>
<tr>
<td>8</td>
<td>Newcastle</td>
</tr>
<tr>
<td>9</td>
<td>Sunderland</td>
</tr>
<tr>
<td>10</td>
<td>London</td>
</tr>
<tr>
<td>11</td>
<td>Southampton</td>
</tr>
<tr>
<td>12</td>
<td>Bristol</td>
</tr>
<tr>
<td>13</td>
<td>Piccadilly</td>
</tr>
<tr>
<td>14</td>
<td>Deansgate</td>
</tr>
<tr>
<td>15</td>
<td>Cheetham Hill</td>
</tr>
<tr>
<td>16</td>
<td>Portswood</td>
</tr>
<tr>
<td>17</td>
<td>Shirley</td>
</tr>
<tr>
<td>18</td>
<td>Highfield</td>
</tr>
</tbody>
</table>

ITEM_HIERARCHY table

<table>
<thead>
<tr>
<th>item_id</th>
<th>parent_id</th>
<th>item_level</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>10</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>11</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>12</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>13</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>14</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>15</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>16</td>
<td>11</td>
<td>4</td>
</tr>
<tr>
<td>17</td>
<td>11</td>
<td>4</td>
</tr>
<tr>
<td>18</td>
<td>11</td>
<td>4</td>
</tr>
</tbody>
</table>

HIERARCHY_ACCESS table:

<table>
<thead>
<tr>
<th>item_id</th>
<th>item_level</th>
<th>parent_id</th>
<th>parent_level</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>13</td>
<td>4</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>14</td>
<td>4</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>15</td>
<td>4</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>17</td>
<td>4</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>18</td>
<td>4</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>22</td>
<td>4</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Listing 2:

Listing 3:

All branches in the same region as Liverpool:

```
select i.item_name
from item i,
     hierarchy_access h
where i.item_id = h.item_id
and h.item_level = 4
and h.parent_id =
  (select parent_id
   from hierarchy_access h,i
    where i.item_id = h.item_id
   and h.parent_level = 2
   and i.item_name = 'Liverpool'
  )
Number of cities in same region as Portswood branch:

select count (*)
from hierarchy_access h
where h.item_level = 3
and h.parent_id =
  (select h.parent_id
   from hierarchy_access h,i
    where i.item_id = h.item_id
   and i.item_name = 'Portswood'
   and h.parent_level = 2 --Region level
  )
```

Listing 4:

```
CREATE TRIGGER trg_hierarchy_insert
ON item_hierarchy
FOR INSERT
AS
DECLARE @ItemID integer,
         @ItemLevel integer,
         @ParentID integer
DECLARE hierarchy_cursor CURSOR FOR
SELECT item_id
     ,item_level
     ,parent_id
FROM inserted
OPEN hierarchy_cursor
FETCH NEXT FROM hierarchy_cursor
INTO @ItemID
     ,@ItemLevel
     ,@ParentID
WHILE @@FETCH_STATUS = 0
BEGIN
WHILE @ParentID > 0
BEGIN
    INSERT INTO hierarchy_access
    SELECT @ItemID
         ,@ItemLevel
         ,@ParentID
    FROM item_hierarchy
    WHERE item_id = @ParentID
    SELECT @ParentID = parent_id
    FROM item_hierarchy
    WHERE item_id = @ParentID
END
FETCH NEXT FROM hierarchy_cursor
INTO @ItemID
     ,@ItemLevel
     ,@ParentID
END
CLOSE hierarchy_cursor
DEALLOCATE hierarchy_cursor
GO
```
This one-day intensive workshop is designed for developers who wish to increase the efficiency and reliability of their code development.

The day will begin by looking at the various hints and tips you can utilize at the code level to improve the quality and reduce the number of bugs you have to contend with.

The next part will look at Apache’s Ant and how you can use this freely available tool for your own development, irrespective of your IDE.

Last, and most important, as the old saying goes: “You can never do enough testing.” This session will look at JUnit and show you how to start building test harnesses for your code so you can begin your testing strategy.

>Performance
Java is a powerful language. While it offers a rich array of tools, the fundamentals mustn’t be overlooked. Improving your code at the core layer will result in great improvements in efficiency and produce (hopefully) less bugs. We’ll look at the do’s and don’ts of programming and learn many hints and tips that will accelerate your Java coding.

>Efficiency with Ant
Apache’s Ant is a powerful scripting tool that enables developers to define and execute routine software development tasks using the simplicity and extensibility of XML. Ant provides a comprehensive mechanism for managing software development projects, including compilation, deployment, testing, and execution. In addition, it is compatible with any IDE or operating system.

>Reliability with JUnit
A critical measure of the success of software is whether or not it executes properly. Equally important, however, is whether that software does what it was intended to do. JUnit is an open-source testing framework that provides a simple way for developers to define how their software should work. JUnit then provides test runners that process your intentions and verify that your code performs as intended. The result is software that not only works, but works in the correct way.

What you will receive...

- INTENSIVE ONE-DAY SESSION
- DETAILED COURSE NOTES AND EXCLUSIVE ONLINE RESOURCES
- JDJ CD ARCHIVE

ONLY $295*

*JDJ subscribers ($395 non-subscribers)

*GROUP DISCOUNTS AVAILABLE

To Register
www.sys-con.com/education
Call 201 802-3058
Sybase Enterprise Portal Evaluation Version

Provides a single, secure, browser-based interface

Part 1 of 2

The Sybase Enterprise Portal is ideal for organizations that are looking for a solution that allows developers to integrate existing enterprise content and applications as well as incorporate the tools that will enable them to access, manage, share, and understand them better.

Parts 1 and 2 of this article will take a close look at the Sybase Enterprise Portal. I’ll start by describing why you might need an enterprise portal and then examine the evaluation version of the Sybase EP.

We’ll look at:
• The architecture of the Sybase Enterprise Portal
• Installing Enterprise Portal 5.01
• Creating pages and using portlets using the portal interface
• Creating pages and portlets using the portal studio
• Using the security features in Sybase EP
• Using the search service
• Creating applications for the enterprise portal

Why Do We Need a Portal?

According to Sybase, “In this period of economic uncertainty, corporations are looking for solutions that will reduce costs and improve productivity. They also need to take full advantage of their information resources and maximize the value of their IT investments. One such solution is to Web-enable business-critical information and applications and to consolidate them within a portal.”

By providing a unified view, portals give employees, partners, and customers immediate access to the information and services they need to perform their jobs. By providing a unified view of information sources, enterprise portals are seen as the “hub” of the convergence of multiple, complementary information management solutions, including document and content management, information search and retrieval, knowledge management, team collaboration, workflow, and business intelligence.

With an Enterprise Portal, you’re able to:
• Integrate all types of enterprise data and take control of heterogeneous IT environments.
• Take advantage of open standards to leverage existing investments.
• Convert unstructured information into usable knowledge.
• Provide content to users according to their specific roles.
• Share information and collaborate across the extended enterprise.

Essentially, the portal will be a secure, personalized, and centralized access point to all enterprise content and applications, as well as the tools users need to find, share, manage, publish, and analyze that content.

Why Sybase Enterprise Portal?

Sybase offers a lot of tools for developers, and you can choose between different databases to fit your needs. They offer enterprise and mobile development tools as well as enterprise integration tools. These tools come together to help you provide high-quality service and/or products to clients at an affordable price (the Enterprise Portal gives both employees and customers fast, secure access to information, which leads to better service and better products). As we’ll see in Part 2, Sybase Enterprise Portal is built around
The Portal Interface:

- An access control database: Sybase Enterprise Portal 5.01 (Sybase EP Evaluation Version) with tools you might already know.
- Point-and-click portal maintenance leading application servers and offers without programming and deploying to users and developers to build portlets without having to learn any portlet application interfaces (API) (see Figure 1). Developers can create HTML, JSP, XML, database, and Web content portlets without programming by using step-by-step wizards or the Portal Studio's Portlet Builder (see Figure 2). Portal Studio is a Web application, so log in using Internet Explorer (or Netscape Communicator in the Enterprise edition). It also provides a collaborative environment for developers and administrators by offering workflow, versioning, a search filter, detail view, and roles-based access.

Sybase EP Evaluation Version

The current version of Sybase EP is 5.01 (we'll start to look at this version in the next article, but there is also an evaluation version). You can download it from http://crm.sybase.com/sybase/www/eB D/EP5EYalDnd.jsp. It's a zip file with about 37MB.

This evaluation version is designed for a single machine install on Windows NT or 2000.

The following browsers are supported: IE 5.0, 5.5, and 6.0. There are known stability issues with Netscape because of its support of JavaScript.

What's included:

- Tomcat version 4.04: Tomcat is the servlet container that's used in the official Reference Implementation for the Java servlet and JavaServer Pages technologies developed by Sun.
- An access control database: Sybase EP provides a page where you can register for the portal. Enter your first and last name, username, password, and e-mail and, after submitting this information, you can log in to the portal by entering your username and password. This information is matched against the information found in the Access Control Database (ACDB). The database used is Sybase Adaptive Server Anywhere.
- The Portal Interface: An enterprise portal that end users view and interact with to access predefined content or to create their own. It consists of site areas and tools that help you create and manage your pages.
  - Page tabs let you name your pages so you can navigate between them. The number of page tabs in a page group varies, depending on the length of the page tab names you choose. If you run out of space, Portal Interface activates the page groups feature.
  - Page groups group your pages in sets of pages. This allows you to add more pages to your account than you can view in a single browser window.
  - The toolbar, located below your page tabs, contains some of the major functions necessary to create and manage your pages. Additional functions display as icons above the page tabs.
  - Portlets display the content you've selected from the Portal Interface or the Web. Portlets are visual, dynamic components that make up a Web page residing in the enterprise portal. Typically, when an end user requests a personalized page, multiple portlets are invoked when that page is created. Take care when choosing a portal since portlets rely on APIs to access various types of information, such as a user profile. Until now, the lack of standards has led portal server platform vendors to define proprietary APIs for local portal components and for invocation of remote components. This creates interoperability problems for portal customers, application vendors, content providers, and portal software vendors. To overcome these problems, the Java Portlet API was defined by the biggest enterprise portal vendors, including Sybase. It will clearly separate portlets from the surrounding portal server infrastructure so that the portlets can run on different portal servers, just as servlets can run on different application servers. The API is now specified (see JSR-168 Portlet Specification at www.jcp.org/en/jsr/detail?id=168) and will be a great enhancement for enterprise portals.
  - Account access within the portal interface is controlled by Enterprise Security. We'll see how this works in Part 2.
- Sybase Portal Studio: The application for portal development and administration, Portal Studio allows developers to build a variety of portlets, templates, pages, and catalogs without having to learn any portlet application interfaces (API) (see Figure 1). Developers can create HTML, JSP, XML, database, and Web content portlets without programming by using step-by-step wizards or the Portal Studio's Portlet Builder (see Figure 2). Portal Studio is a Web application, so log in using Internet Explorer (or Netscape Communicator in the Enterprise edition). It also provides a collaborative environment for developers and administrators by offering workflow, versioning, a search filter, detail view, and roles-based access.

All user access within Portal Studio to Studio objects and Portal Interface pages and portlets is controlled through the Portal Studio Account Manager. Developers create content in Portal Studio and deploy it for display to end users in portlets in the Portal Interface. Administrators use both Portal Studio and Portal Interface to perform portal administrative tasks. Pages and portlets within a Portal Interface user account are controlled by either an administrator or by the end user. Portlets created in the Portal Studio can be arranged into pages that are automatically added to portal
Installation

When installing an Enterprise Portal for the first time, I locked my door and turned off the phone for the entire day. It should just be the machine, some CDs, and me. Yes, I was afraid of doing the installation, but I found that the installation is so easy that it can’t really be called an installation. However, decide for yourself if that’s true. The following is a list of the steps involved:

1. Extract the file to your hard drive (let’s say this is c:\).
2. The evaluation version uses the name demo.sybase.com. To make your machine recognize it, edit your hosts file. You’ll find it at \winnt\system32\drivers\etc\hosts. Add a line:
   
   127.0.0.1 demo.sybase.com

3. Go to the directory where you unzipped the files and run the file startdb.bat. You’ll initially see a few messages with “Ping server failed – Database server not found.” This is okay, because the script pings the database during startup. Wait until you see the message “Ping server successful.” The icon of the “SQL-ASA” database will appear in your taskbar.

4. Within the same directory, run the file starttomcat.bat. A new window will be launched and you’ll see several messages starting with “Starting service Tomcat-Standalone” and “Tomcat connecting to our ASA database.” Wait until you see the message “Apache Tomcat/4.0.4,” then minimize this window.

5. To run the evaluation version of the Sybase Enterprise Portal use the following URL: http://demo.sybase.com:4040/onepage/index.jsp (see Figure 3).

6. That’s it. Click on the “Join now” link and register for the portal (see Figure 4). To run the evaluation version of the Sybase Portal Studio use the following URL: http://demo.sybase.com:4040/onepage/index.html.

   Use the existing login account “opsuper/opsuper” (Note that the first character of the password is a zero.)

To stop the portal:
• Close the window containing Tomcat.
• Click on the ASA icon in the taskbar and then on “Shutdown”.

To login again in future sessions, simply perform steps 3 and 4 to start the database and Tomcat, and launch the portal with steps 5 and 6. That’s it. Quite easy, isn’t it?

I suggest you get your hands on the EP light version as soon as possible, install it, and build your first portlets. Next month we’re going to install the “big brother,” the Sybase Enterprise Portal 5.01, and then we can start building our own portlets to provide company information to our employees or customers from a single point of entry.

Conclusion

The Sybase Enterprise Portal is aimed at providing a single, secure browser-based interface to all corporate applications and data. Its design and scope appeal to companies because it removes the heavy structure of complex, 32-bit applications. Only the most frequently used functions of each application are exposed, giving users a single point to access their information, whether inside or outside the office. With an Enterprise Portal, professionals are then able to use technology in the way it was intended, and can quickly and efficiently get on with their work.
Introducing PowerBuilder 8! The award-winning, industry-leading application development environment just got better. Release 8 of PowerBuilder adds exciting new features and capabilities that will make your development of Web, client/server, and distributed applications easier, faster, and more cost-effective. New features focusing on improved productivity, tighter integration with EAServer, and Web application development make this release indispensable to your development efforts.

“FULCRUM” is a Java development tool that uses a proprietary concept of code templates that can be used as “building blocks” to construct efficient Java objects and applications. Unlike code generators that impose their coding techniques and run-time environments on the users, FULCRUM allows users to build their own coding standards in the templates and also lets users switch to manual coding at any point in time.

Simplicity Enterprise Lite provides the ability to build server-side, enterprise class Web applications. Using a palette of graphical modules, the developer links together their desired functionality and interactively tests their application in one seamless development environment. Application targets include Web sites, WAP-enabled mobile devices, n-tier application services, and enterprise Servlets.

JBuilder is the leading cross-platform environment for learning Java programming and personal application development. JBuilder 6 Personal includes an integrated editor, debugger, compiler, visual designers, wizards, and tutorials.

WebLogic Developers Journal.com SYS-CON Media, the world’s leading publisher of i-technology magazines for developers, software architects, and e-commerce professionals, brings you the most comprehensive coverage of WebLogic.
NVOs and NVCs

What's the difference?

Adaptations...as software developers we thrive on adaptations. A large part of our present and future work is and will be simply morphing or adapting our previous efforts or the efforts of others before us.

Sometimes we need to reassemble pieces when an application needs maintenance and other times we bring pieces from our past and get to enjoy the creative thrill of new application development.

Many things in our environment constantly change. When PowerBuilder first came into being, many of the TLAs (three-letter acronyms) we constantly toss around today didn't even exist. While it's important to utilize good practices, it's even more important to reevaluate, improve, and refine the rules we live with everyday. When Sybase added the concept of a component to the world of PowerBuilder, it was logically placed in the realm of the nonvisual object (NVO). While components are unmistakably NVOs, they are a specific NVO subclass and invariably require special handling whenever they're used. In order to have meaningful conversation and clear usage practices regarding these nonvisual components, they must be distinguishable from the ancestor NVO - thus the idea of an NVC.

Nonvisual Component

NVCs should be viewed as separate mini applications. The scope of NVC functionality can be greater than a single application. Logic built into an NVC can be shared across multiple applications and utilized as part of a multi-phased transaction, even transactions spanning multiple databases. When using an NVC, the execution of logic is transparent to the caller.

NVCs are a great way to share resources (see Figure 1); however, overuse can also cause problems. If a Web application uses a large number of NVCs, be sure the code is not doing intercomponent calls, such as checking for an empty string. To illustrate, if there's a form that's been submitted with 30 different fields, you don't want to make 30 separate calls to a component. In this case, if you're running clustered application servers, you could even be calling between different boxes. In this situation it would be better to use an NVO that's locally instantiated. Be sure the NVC calls are warranted.

Nonvisual Object

A nonvisual object is an object that exists entirely within a single application scope. NVO classes contain business rules that are frequently reused. NVOs will always be an extremely important tool for PowerBuilder developers. Their role is critical for making robust and adaptable applications, even mini applications. When creating an NVO, keep the class size down; this will help your performance. It will take less time to load the NVO into memory and use fewer resources while helping execute the rest of your scripts.

Good candidate scripts for NVOs will be utilitarian and widely reused bits of code that help you get basic tasks done, such as special string handling or date and time functionality. You'll also find that time-tested code that's not likely to change and that enforces your everyday business rules is a good choice.

If you have existing NVOs, why not use them inside the components you create? Say you have a few killer NVOs, the only problem is they pass DataStores; don't be disappointed because the CORBA interface won't allow you to pass DataStores. These existing NVOs make great helper or inner classes and will give your components that extra piece of functionality you need to make an awesome component.

The Differences Between NVCs and NVOs

To begin there are extra events found in NVCs, such as the activate and deactivate events, that add additional complexity not found in the NVO (see Table 1). These added events make NVCs more powerful and flexible.

Having activate and deactivate events implies NVCs are meant to be pooled in memory to save the overhead of creating and destroying an NVC at every use. Conversely, NVOs are usually destroyed; these single-use objects indicate that NVOs should only be allowed to take up resources when required. Keep in mind that the single use may include many...
method invocations, perhaps for a validation routine, but when the routine is finished with the NVO, it should no longer be kept around; the destructor will get called and the resources will be released.

Each object is used and called differently. NVCs use proxies and execute remotely to the caller, while NVOs are loaded into memory within the caller’s scope and execute locally.

The preparations required to use each of the objects is quite different. An NVO Create statement will pull into memory the definition of the NVO created. Depending on the nature of the NVO, this may be a significant use of resources and is likely something you don’t want to leave hanging around in memory. Because you would have to call the init() for any subsequent uses, be sure it’s ready for the caller. Sometimes it’s required to pass in the NVO’s parent or the transaction the NVO will participate in. NVOs will frequently require that you call one or more methods to set the NVO to a ready state for method execution.

NVCs require the caller to use a remote interface definition, a proxy, and a connection to the server, which returns an IOR pointer. This pointer generated by a transactionserver.createinstance() call is simply a string of instructions on where to send a remote call and how to package a call to the Object Request Broker (ORB).

NVCs are self-initializing; code placed in the activate and deactivate events should complete the required setup and cleanup for each use. If there are memory-intensive objects like DataStores, it’s more efficient to place those in the activate and deactivate events. However, to ensure there are no large result sets hanging around in memory, call a reset on your DataStores.

NVOs only have constructor and destructor events. Frequently this will require an Initialization method to allow for the reuse of the object. Passing in a Parent object or a Transaction object adds extra steps to your code, important steps that you shouldn’t miss, depending on how the object is designed.

Holding an instance of an NVC remote interface, a.k.a., an IOR pointer, will always be less costly from a resource point of view than holding the entire definition of an NVO.

NVOs require holding a Power-Object variable reference, and the library holding the NVO source must be held in memory.

NVOs are internal to PowerBuilder. Even though you may use a particular NVO in many places within an application, it’s not meant for consumption outside of PowerBuilder.

NVCs can be published for the world at large using CORBA-compliant data types, which means you can’t expose the same data types as the NVO. An NVO’s logic and rules can be reused, but a copy for each application that uses that logic can eat up a lot of resources. An NVC can share the same instance across multiple applications. Which one is better - sharing or the always-in memory cost - depends on your usage and needs.

While a component shares many attributes with a standard NVO, there are enough important differences when we discuss and use regular NVOs and NVCs in development that a distinction should be made. A service object is different from a service component. Why waste any time inferring by context how to handle the initialization, storage, and cleanup of a given object? Spend the time talking about the logic that should be involved.

If the points I attempted to make sound reasonable, whether or not you agree, keep it in mind for the next conversation about NVOs and NVOs or rather NVOs and NVCs.

Give the new distinction a whirl; I believe you will find it lends clarity and efficiency to your discussions.

TABLE 1 The differences between NVOs and NVCs

<table>
<thead>
<tr>
<th></th>
<th>NVO</th>
<th>NVC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Events</td>
<td>Constructor, Destructor</td>
<td>Constructor, Destructor, Activate, and Deactivate</td>
</tr>
<tr>
<td>Designed for pooling</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Execution related to caller</td>
<td>Locally</td>
<td>Remotely</td>
</tr>
<tr>
<td>Self-initializing</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>CORBA-compliant public interface required</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Meant for public consumption</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

AUTHOR BIO

Jerry Neppl, previously CEO of a small liquidation company, has been working for PowerTeam, Inc., in Minneapolis for the past year and a half, having fun and getting paid to play everyday.

www.POWERBUILDERJOURNAL.com  PBDJ volume 10 issue 4 29
Many Web sites provide some of this, but they're spread all around the world. What if you could find everything in one place? You, the Sybase techies, have been asking us to provide this site as well as a mechanism that will enable developers around the world to collaborate. Well, we've done it! In early April, Sybase launched a new resource expressly for its techie community. SDN CodeXchange, from the Sybase Developer Network, allows for the free exchange of code samples, utilities, scripts, and other tools. Content in CodeXchange is not supported by Sybase, but can come from any Sybase user, customer, or employee—all of whom will benefit from the contributions of others.

CodeXchange also hosts collaborative development projects where users can work on a joint project involving a general technology or a specific aspect of a Sybase product, including open source. Sybase is sponsoring CodeXchange solely for the benefit of SDN members—it's all technical stuff, for techies, by techies. Now it's your turn to do your part. We need you to participate and contribute samples, utilities, and snippets. The success of this project depends on you!

How It Works

CodeXchange is organized around projects, each of which contains specific tools like file sharing (for code samples) and Concurrent Versioning System (CVS) for larger pieces of code, utilities, or open-source projects that need to track changes between versions. Project “communities” address specific products such as PowerBuilder, ASE, and EAServer. Other communities focus on more general technologies, such as Sybase Open Source for eBXML Messaging. In each project, CodeXchange provides the following features to facilitate communication and involvement:

- File sharing
- Code revision control (CVS)
- Mailing lists
- Issue and bug tracking
- Customizable project pages
- Web-based, secure environment

Any SDN member can access CodeXchange by signing on to MySybase or SDN from the Sybase.com site. You can then browse or join existing projects, and suggest new ones. You can download pieces of code you find in a particular project and, more important, submit code or tools that you’ve created.

Need a utility that works with a new version of PowerBuilder? Find one on CodeXchange, download it, change it, and submit it back.

Have a code sample you think others might find useful? Submit it to the appropriate project and make a name for yourself in the community.

Your level of involvement is up to you; you can simply observe, or you can assign or fix defects in the code you find, communicate with other members via mailing lists, and submit your own work. The more people who contribute, the more valuable the community will be for its members, so we’re looking for eager participants.

CodeXchange will be of particular interest to PowerBuilder users with the release of PowerBuilder 9. We have already aggregated our existing Sybase-produced samples and utilities, including new PB 9 sample code. We have a number of contributions from TeamSybase, including several PBNI-related open-source utilities (you’ll see some articles about these utilities in upcoming issues of PBDJ). Come download the source for these, upload your own, and start collaborating!

CodeXchange is similar to a library that contains only donations so it will start off small, but we expect it to grow rapidly as the Sybase world centralizes all their collaborative efforts there.

You can access CodeXchange via SDN at www.sybase.com.developer. If you’re not an SDN member, you can sign up at the Web site. Everyone is welcome. See you there!
SAVE $31 Off* the annual newsstand rate

Receive 12 issues of PBDJ for only $149.00. That’s a savings of $31 off the annual newsstand rate. Visit our site at www.powerbuilderjournal.com or call 1-888-303-5282 and subscribe today!

Here’s what you’ll find in every issue of PBDJ

- New PowerBuilder Features
- Tips, Tricks and Techniques in Server-Side Programming
- DB Programming Techniques
- Tips on Creating Live PowerBuilder Sites
- Product Reviews
- Display Ads of the Best Add-on Products
- PowerBuilder User Group Info

*Offer subject to change without notice

The world’s leading i-technology publisher
SUBSCRIBE TODAY TO MULTI-

Go To www.sys-con.com/suboffer.cfm

and receive your FREE CD Gift Package VIA Priority Mail

Each CD is an invaluable developer resource packed with important articles and useful source code!

List Price $198

More than 1,400 Web services and Java articles on one CD! Edited by well-known editors-in-chief Sean Rhody and Alan Williamson, these articles are organized into more than 50 chapters on UDDI, distributed computing, e-business, applets, SOA and many other topics. Plus, editorials, interviews, tips and techniques!

Web Services Resource CD

List Price $198

The most complete library of exclusive JDDJ articles compiled on one CD! Assembled by JDDJ Editor-in-Chief Ajit Sagar, more than 1,400 exclusive articles are organized into over 50 chapters, including fundamentals, applets, advanced Java topics, Swing, security, wireless Java... and much more!

Java Resource CD

Pick one CD with your 3-Pack order

Pick two CDs with your 6-Pack order

Pick three CDs with your 9-Pack order

List Price $198

The most complete library of exclusive WLDJ articles ever assembled! More than 200 articles provide invaluable information on “everything WebLogic”, including WebLogic Server, WebLogic Portal, WebLogic Platform, WebLogic Workshop, Web services, security, migration, integration, performance, training...

WebLogic Resource CD

List Price $198

The most complete library of exclusive CFDJ articles on one CD! This CD, edited by CFDJ Editor-in-Chief Robert Diamond, is organized into more than 30 chapters with more than 400 exclusive articles on CF applications, custom tags, database, e-commerce, Spectra, enterprise CF, error handling, WDDX... and more!

ColdFusion Resource CD

List Price $198

The largest and most complete library of exclusive XML articles on one CD! Edited by well-known Editors-in-Chief Ali Sagar and John Evdemon, these articles are organized into more than 30 chapters containing more than 1,150 articles on Java, XML, XML A, XSLT, c-BiXML, data transition... and more!

XML Resource CD

List Price $198

The most up-to-date collection of exclusive WSDJ articles! More than 200 articles offer insights into all areas of WebSphere, including Portal, components, integration, tools, hardware, management, sites, wireless programming, best practices, migration...

WebSphere Resource CD

List Price $198

An up-to-the-minute collection of exclusive Linux Business & Technology articles plus Maureen D.Gara’s weekly LinuxGram, which keeps a finger on the pulse of the Linux business community. Edited by LBT’s Editor-in-Chief Alan Williamson, these articles cover migration, hardware, certification, and the latest Linux-related products. Available June 2003!

Linux Resource CD

List Price $198

Your order will be processed the same day!
OFFER FROM SYS-CON Media

AND SAVE UP TO $400 AND RECEIVE UP TO 3 FREE CDs!

TO ORDER
- Choose the Multi-Pack you want to order by checking next to it below. • Check the number of years you want to order. • Indicate your location by checking either U.S., Canada/Mexico or International. • Then choose which magazines you want to include with your Multi-Pack order.

3-PACK
Pick any 3 of our magazines and save up to $275.00
Pay only $175 for a 1 year subscription plus 1 FREE CD
• 2 Year – $299.00
• Canada/Mexico – $245.00
• International – $315.00

6-PACK
Pick any 6 of our magazines and save up to $350.00
Pay only $395 for a 1 year subscription plus 2 FREE CDs
• 2 Year – $669.00
• Canada/Mexico – $555.00
• International – $710.00

9-PACK
Pick 9 of our magazines and save up to $400.00
Pay only $495 for a 1 year subscription plus 3 FREE CDs
• 2 Year – $339.00
• Canada/Mexico – $295.00
• International – $390.00

www.sys-con.com/2001/sub.cfm
PowerBuilder News

All things of interest to the PB community

BY BRUCE ARMSTRONG

EA Server

EAServer 4.2

2/11 - Sybase announced the availability of EAServer 4.2, an open application server that has been selected by a growing number of state and regional governments for its uncompromising performance and support for secure Web services within an "open-choice" development and deployment environment. As evidenced by forward-thinking customers including Hennepin County, Minnesota; the Texas Workforce Commission; and Wellington City Council, New Zealand, government agencies and enterprises alike are increasingly turning to Sybase EAServer for the technology infrastructure to help them unlock the value of their existing data and to quickly deploy and develop new services and applications.

IANYWHERE

Nec

2/12 - iAnywhere Solutions, a subsidiary of Sybase, announced an agreement with NEC Corporation to stimulate the development of mobile, wireless, and remote database-powered solutions in Japan. As part of this relationship, NEC will resell SQL Anywhere Studio and Manage Anywhere Studio 5, providing a full life-cycle management solution for enterprise applications and hardware including PCs, laptops, and mobile devices. This new version introduces new OS migration capabilities, enhanced mobile security, and new specialized software modules - enabling companies to select just the features they require today and easily add new functionality in the future. Manage Anywhere Studio 5 features targeted software modules to address specific business needs including asset management, live support, mobile and remote software deployment, and OS migration.

2/03 - iAnywhere Solutions, a subsidiary of Sybase, announced that PDS Research selected SQL Anywhere Studio and Manage Anywhere Studio for use in its cardiopulmonary diagnostic equipment. PDS Research will leverage technology from iAnywhere Solutions to power its KoKo Link solution, which is provided to laboratories to test patient breathing capacity as part of clinical drug trials. www.pulmonarydata.com

POWERBUILDER

PowerBuilder 9.0

3/04 - Sybase announced that PowerBuilder 9.0 is currently scheduled for general availability the week of March 24. This version of Sybase's industry-leading rapid application development tool enables developers to continue building rich-client applications for their mission-critical business requirements, and at the same time propel them further into Web and n-tier development with significant new capabilities. PowerBuilder 9.0 also lays the foundation for "4GLplus," the next RAD environment from Sybase that will bring an even higher level of developer productivity through tight integration of design, modeling, development, deployment, and management.

Adaptive Server

ENTREPRISE

DBXray

3/03 - Sybase announced the availability of ASE Performance Monitoring Option - DBXray for Sybase, a part of the SmartDBA solution for comprehensive enterprise data management from BMC Software Inc., a leader in enterprise management. DBXray for Sybase simplifies database management through a performance "dashboard," allowing database administrators (DBAs) to easily monitor the health of an overall system, anticipate and solve potential problems, and achieve a 24-hour view of activity for database performance optimization. The tool will be available as an option in Sybase Adaptive Server Enterprise 12.5.0.3, Sybase's enterprise-class relational database management system, and will be supported by Sybase.

CORPORATE

AvantGo

2/26 - Sybase announced the completion of its acquisition of AvantGo, leading provider of mobile enterprise software. Sybase will operate AvantGo in conjunction with its iAnywhere Solutions subsidiary, strengthening the company's leadership position in the mobile middleware market. AvantGo stockholders approved the acquisition by Sybase of all outstanding shares of AvantGo for cash consideration of approximately $1.029 per share. AvantGo common stock will no longer be publicly traded. The majority of AvantGo employees are joining iAnywhere Solutions and will be working in the Dublin, California, headquarters. www.avantgo.com

iAnywhere Solutions Subsidiary

2/11 - iAnywhere Solutions, a subsidiary of Sybase, announced the formation of iAnywhere Solutions K.K. in Japan. This new subsidiary of iAnywhere Solutions will work to develop and expand the company's presence in the Japanese mobile, remote, embedded, and workgroup database and mobile middleware markets. Noriyuki Hayakawa, who has directed the company's mobile and embedded operations in Japan as vice president of Sybase K.K. and country manager for iAnywhere Solutions, Inc., will become the president of the new subsidiary.

EVENTS

The Fifth Annual EAServer 4.2 and Sybase Tools Seminar

Minneapolis, MN

April 7, 2003

www.powerobjects.com/seminar

TechWave 2003

Orlando, FL

August 4-8, 2003

www.sybase.com/techwave

2003

bruce.armstrong@teamsybase.com
E.Crane Computing
www.ecrane.com
PowerObjects
www.powerobjects.com