Appeon Developer and User Productivity Study

An Appeon Whitepaper

Appeon® for PowerBuilder®

May 2004

LAST REVISED: May 25, 2004

Table of Contents

1 Introduction

1.1 What is Appeon Pet World?

The Appeon Pet World is an online pet store developed by Appeon to demonstrate Web RAD (Rapid Application Development) capabilities for using Appeon to build n-tier Web applications. In this study, we compare and contrast the Appeon Pet World to the .NET Pet Shop developed by Microsoft. The comparison focuses on architecture, developer productivity, and the end-user experience. You may download the Appeon Pet World source code and related studies from the following Websites:

- The Appeon Pet World PowerBuilder source code is available publicly at http://www.appeon.net/PetWorld/.
- The Microsoft .NET Pet Shop source code and study is available publicly at http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnbda/html/psimp.asp.
- The Sun J2EE Pet Store source code and study is available publicly at http://java.sun.com/j2ee/blueprints/.

1.2 Objectives of this study

Appeon took the Microsoft's .NET Pet Shop blueprint application version 2.0, the .NET Pet Shop, and implemented the same functionality using PowerBuilder. The result is Appeon Pet world. For this reason, the Appeon Web RAD paradigm can be compared to both ASP.NET and JSP. Specifically, this study focuses on comparing the architecture, developer productivity, and end-user experience between ASP.NET and JSP, and Appeon.

Because these productivity studies have caused a great deal of debate, Appeon took additional measures to ensure the accuracy of our productivity claims. There is no more accurate way to evaluate developer productivity other than to measure the *time* taken to completely build the application. As such, unlike the .NET Pet Shop study, Appeon measured the time taken to build the Appeon Pet World. However, this study still highlights the lines of code (LOC) comparison for a couple of reasons. First, it is the exact evaluation criteria used by .NET. Second, it can be a reasonable ballpark proxy for developer productivity, especially when it is necessary to estimate the productivity of a future project, such as a PowerBuilder Web migration project.

Appeon does not directly compare the Pet World to Sun's J2EE Pet Store based on delivery productivity since this is an unfair comparison. The comparison is unfair because the Sun J2EE Pet Store has not been optimized to reduce the LOC. As such, Appeon focuses exclusively on the .NET Pet Shop and for argument's sake, assumes that J2EE Web development is no more productive than .NET Web development. It is up to the reader to extrapolate from this study and the .NET Pet Shop study in arriving to his or her own conclusions about the benefits of Appeon vs. J2EE in terms of the architecture, developer productivity, and end-user experience.

1.3 How Appeon Pet World was developed

The full Sun J2EE Pet Store contains three sample applications:

1. Java Pet Store: The main J2EE Blueprints application.
2. Java Pet Store Administrator: The administrator module for the Java Pet Store.
3. Blueprints Mailer: A mini-application that presents some of the J2EE Blueprints design guidelines in a smaller package.

Using PowerBuilder, Appeon engineers set out to precisely replicate the UI layout, application functionality, and business rules of the J2EE Pet Store (the administration and mailer modules were excluded from this study), as well as the .NET Pet Shop. Since the .NET Pet Shop and J2EE Pet Store are functionally equivalent, Appeon chose to focus its efforts on one of these two applications - the .NET Pet Shop. Appeon engineers painstakingly examined every Web page and line of code of the .NET Pet Shop as well as documentation. This understanding was used to produce the Appeon Pet World.

The Appeon Pet World began life as a distributed PowerBuilder (DPB) application or 3-tier client/server application. Appeon engineers used PowerBuilder 8.0.4 to build the application. In building the application,

Appeon engineers refrained from using traditional client/server features that cannot run in a Web environment. In doing so, however, no compromises were made since the .NET Pet Shop and J2EE Pet Store are Web applications, and as such, do not utilize these client/server features.

For the application UI, Appeon engineers simulated the "page" experience of the .NET Pet Shop Web application within PowerBuilder to make the end-user comparison identical in every respect. However, Appeon is intended for building rich Web applications. These rich Web applications eliminate the cumbersome and counterproductive "page-by-page" application workflow that is legacy of the days where the Web was used for viewing documents. They incorporate a powerful event-driven programming model and advanced controls including the famous DataWindow, resulting in a rich Windows-style user interface.

The Appeon Pet World 3-tier client/server application was transformed automatically into an n-tier Web application using Appeon for PowerBuilder. Appeon engineers did not write a *single line* of HTML, JavaScript, and Java or .NET code. Furthermore, they did not require any knowledge of the Web architecture and programming model, such as developing cookies and managing the user session and state. Appeon engineers concentrated only on PowerBuilder client/server application development, and deployed the application automatically to the Web at the click of a button.

2 Pet World – A *Bona Fide* Web Application

The Appeon Pet World is a Web application that replicates the online pet store functionality of Microsoft's .NET Pet Shop, and inherently, the functionality of Sun's J2EE Pet Store. It deploys to the N-tier Web architecture with high scalability and security.

2.1 Functional walkthrough

The Appeon Pet World, like the .NET Pet Shop and J2EE Pet Store, is essentially a fully functioning online store from the shopper's perspective. It allows shoppers to browse and search the product catalog, add items to a shopping cart, view/modify the shopping cart, and finally checkout using their personal account.

2.1.1 Homepage

The main page loads when the user first starts the application. The user may return to the home page at any time, by clicking the Appeon Pet World logo in the upper-left-hand corner.

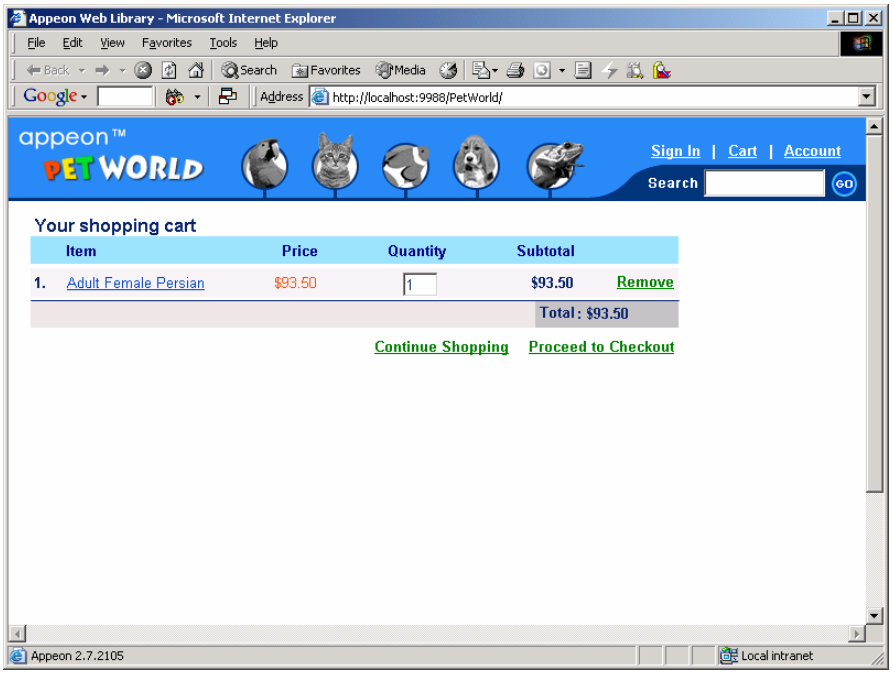
2.1.2 Browsing the product catalog

The product catalog allows users to browse in hierarchical order. Click the top-level category to open the sub-category page; click the sub-category; select the specific product and add it to the shopping cart.

- Top-level Category: There are five top-level categories, and each category has several associated products.
- Products: When a product is selected, all variants of the product are displayed.
- Product Details: Each product variant will have a detailed view.

2.1.3 Using the shopping cart

The shopping cart is full-featured, allowing the shopper to add, remove, and update any items. Once the shopper has added one or more items to the cart, the shopping cart lists the items and the cart's running total.



Quantity update:

Appeon for PowerBuilder is capable of adding a much more user-friendly "Update" functionality than the .NET Pet Shop, without any developer headaches. Microsoft's .NET Pet Shop provides an "Update" button for the user to update the price when quantity has been manually modified. As you can see above, Appeon Pet World is free of this unnecessary "Update" button, which clutters the user interface. Instead, the price is updated immediately and

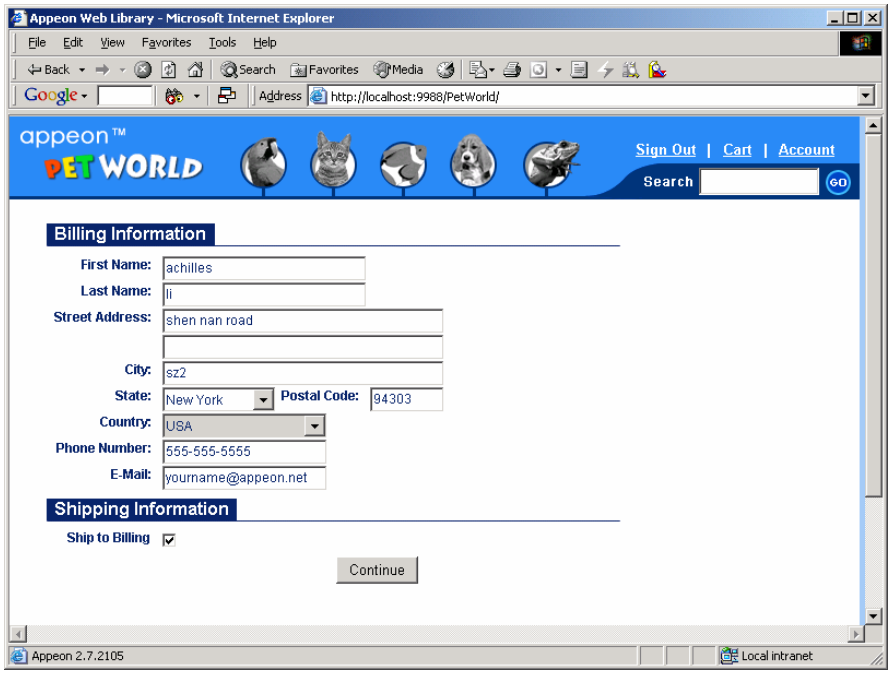
automatically when the user moves away from the shopping cart "Quantity" field: click anywhere else in the window, or press ENTER or TAB. In addition, the price update occurs without incurring a cumbersome page refresh.

Proceed to checkout:

Appeon has expanded the checkout functionality of the .NET Pet Shop. The .NET Pet Shop provides a "Proceed to Checkout" link that takes the user to the checkout from the shopping cart. In addition to this functionality, Appeon also provides a "Continue Shopping" feature. This feature enables the user to return to the catalog section for the most recent item added to the cart in the event the user is not ready to checkout.

2.1.4 Checking out

Checkout requires the user to be registered with the system and logged in. The process consists of two key steps. First, verify the billing address and provide shipping information if the default will not be used. Second, verify the order and provide payment information. When the order is submitted, the system will provide user with an on-screen order confirmation with a tracking number.



Login redirection:

When the user clicks "Proceed to Checkout," the user is redirected to the login page if the user has not signed in yet. After being authenticated, the user is automatically directed back to the checkout form.

Billing and shipping information validation:

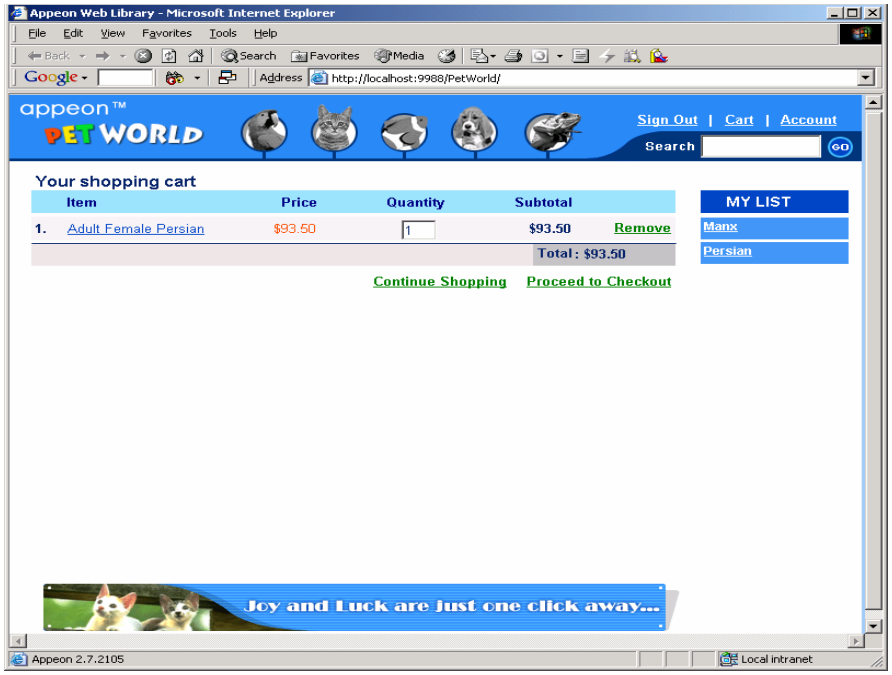
When checking out, the user will have the opportunity to verify the billing address information and change the shipping address information if necessary. All fields are required and will be validated. By default, the shipping information will be set to the same as the billing information.

Order verification and payment:

Before the order is submitted, the user verifies the order and enters the payment information. The "Submit" button will submit the order to the database and generate an order ID number for tracking purposes.

2.1.5 Personalizing the shopping experience

Appeon Pet World can be personalized based on your pet of choice. You must be signed in before your account profile can be retrieved and applied to personalize Appeon Pet World.



Personalizing the pet tips banner:

The pet tips banner feature will display, at the bottom of the shopping cart, a personalized banner based on your favorite pet category.

You can enable/disable the pet tips banner or change your favorite pet category by clicking the "Account" link and modifying the "Profile Information" section.

Personalizing MyList:

The MyList feature will display alongside the shopping cart a list of products for your favorite pet category that can be purchased from Appeon Pet World. Clicking on one of the items will take the user from the shopping cart to the product catalog section that was clicked.

You can enable/disable MyList or change your favorite pet category by clicking the "Account" link and modifying the "Profile Information" section.

2.2 Standard technology

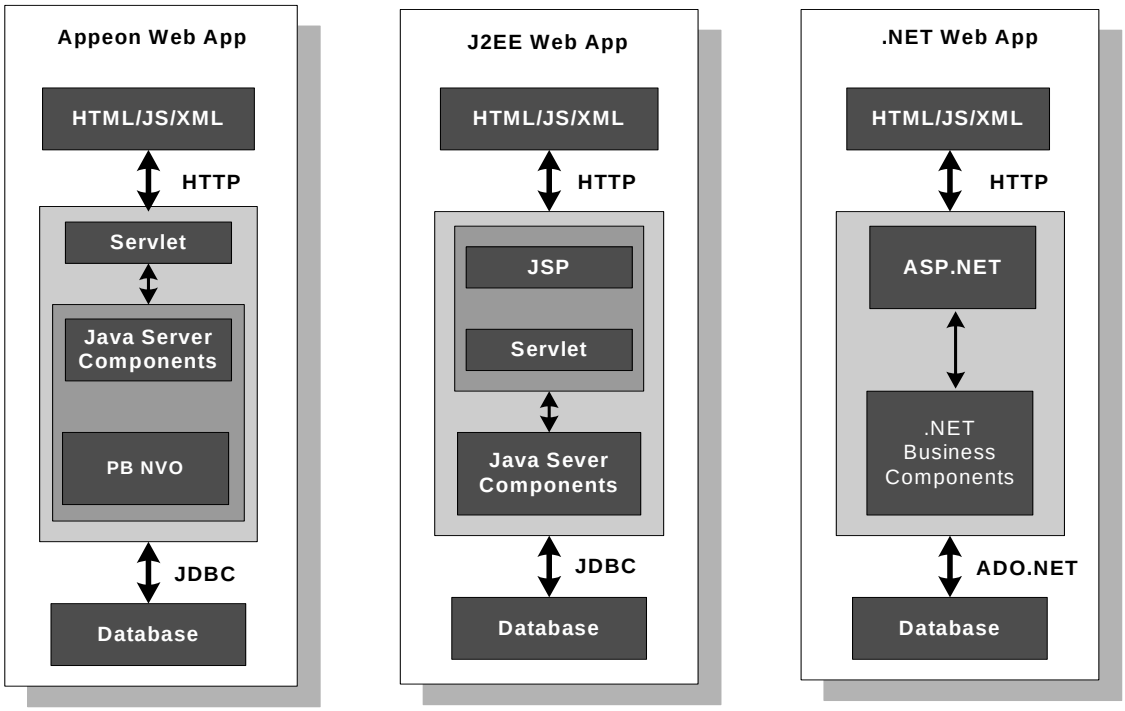
The Appeon Pet World is implemented using industry standard technologies that are open and proven.

The application presentation layer is implemented as an HTML think-client. The presentation layer uses HTML, JavaScript, and XML. As such, Appeon Pet World does not require the user to download browser plug-ins, Java Applets, ActiveX controls, Flash, or any other client software.

The middle-tier of the Appeon Pet World architecture is largely J2EE-based. Appeon Pet World deploys against the J2EE-based Appeon Server run-time framework, which provides runtime services for Appeon Pet World, such as database connectivity and DataWindow support, business logic execution, security, and integration. Only two of the many server components that make up the Appeon Server run-time framework are not J2EE CORBA server components. Two n-tier PowerBuilder n-tier NVOs provide Image DataWindow generation functionality and DataWindow data connectivity. The only other non-Java code running on the middle-tier are the PowerBuilder business logic contained in n-tier NVOs of the Appeon Pet World.

Appeon Pet World interacts with the server using only standard communication protocols. The browser sends requests and data to the Web tier using HTTP/HTTPS. Within the middle tier, J2EE CORBA components communicate with PowerBuilder CORBA components using IIOP. The middle tier queries the database tier using SQL and JDBC. An ODBC back-end can be supported as well using a production-quality JDBC-ODBC bridge driver.

Appeon Architecture Comparison



2.3 Browser-based n-tier architecture

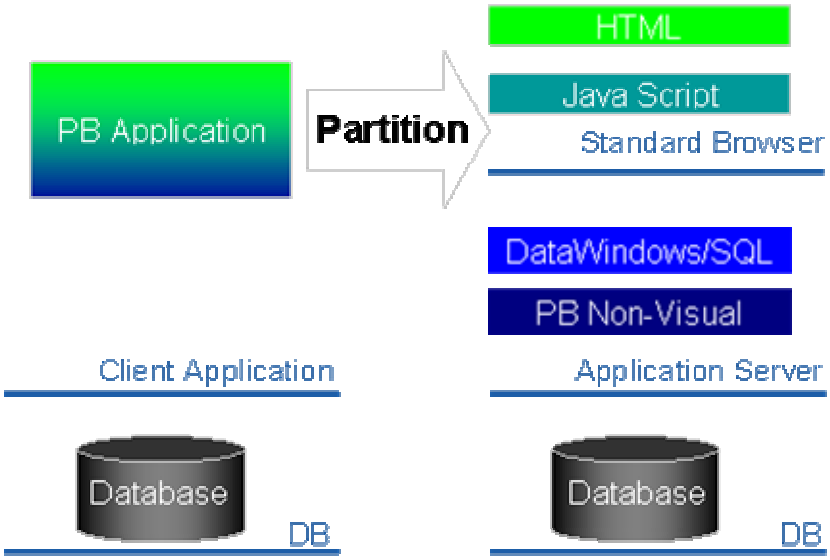
Appeon Pet World deploys the standard n-tier Web architecture – the presentation tier, middle tier, and data tier are logically separated.

- **Presentation Tier (Web Browser)** – The Web application presentation layer is implemented to deploy to the Web server and run at the Web browser. HTML pages define the layout of the Web application windows. JavaScript files contain the JavaScript equivalent of the PowerScript or UI logic coded in the Window object, user object, menu object, and any other visual objects.

- **Middle Tier (Application Server)** – The middle tier, which hosts the business logic, is implemented with J2EE CORBA components and PowerBuilder CORBA components. The J2EE CORBA components execute the DataWindows and any Embedded SQL coded in the presentation layer of the PowerBuilder

client/server application. The PowerBuilder CORBA components host the business logic of the Appeon Pet World, which ahs been coded as n-tier NVOs.

- **Data Tier (Database)** – The database stores the raw data for the application and stored procedures, if any were used. The existing database from the PowerBuilder application can be simply re-used without modification.



2.4 N-tier scalability

The Appeon Pet World n-tier architecture inherently offers superior server and database scalability that users expect today. The three key features of the architecture that makes this possible is server clustering with load balancing, stateless components, and database connection pooling.

- **Server clustering.** Appeon Pet World can be deployed to a cluster of Web servers and application servers with load balancing. This allows the system to handle a growing user base without re-architecting the application and system or over investing in a monolithic super computer.
- **Stateless components.** The business logic of the Appeon Pet World is deployed as a stateless component: the n-tier NVO named n_Pet. Stateless components are deactivated after each method call such that memory does not continue to be consumed. Implementing the business logic as stateless components will offer the best scalability.
- **Connection pooling.** Appeon Pet World connects to the database through the application server using connection pooling or connection cache. This technology does not require a unique database connection to be established for each application user. Rather, it shares and swaps the database connections to reduce the load on the database. This technology boosts database scalability dramatically, which is crucial since the database is the most common bottleneck in the system.

2.5 Strong Web security

Appeon supports the leading Web security standards and measures to ensure that all data transmissions are safe, secure, and authentic.

- Appeon Web applications are compatible with all corporate firewalls since Appeon communicates using HTTP over port 80 and only Web documents pass through the firewall (e.g. .HTML, .XML, .JS files).
- SSL encryption (HTTPS) up to 128-bits can be applied to all data transmissions to protect even the most sensitive data transmissions. This level of encryption is so secure that the USA government forbids

exporting Web browser software with 128-bit encryption to some overseas countries. For more information on the U.S. Export Administration Regulations (EAR), 15 C.F.R. Parts 730-774, and the Bureau of Export Administration ("BXA"), please see the BXA homepage at http://www.bxa.doc.gov/.

- Digital certificates may be used to ensure that the Appeon Web application and all data transmissions are authentic. That is, the application and data transmissions are in fact from the specified party/server, and that the application and data transmissions have not been altered or corrupted in any way.

Appeon conforms to the strict Web browser security sandbox to ensure that the client computer system security cannot be compromised.

- The Appeon client-side utilizes only non-invasive Web technologies that cannot bypass the Web browser security sandbox.
- It is implemented using only HTML, JavaScript, and XML. There are no Java Applets, ActiveX controls, Flash, browser plug-ins, or other client software required other than the standard Microsoft Web browser.
- Thus, Appeon Web applications cannot access the user's hard drive, the Windows registry, operating system files and settings, or, indeed, anything outside of the Web browser. There is virtually no higher level of client security available for the Web.

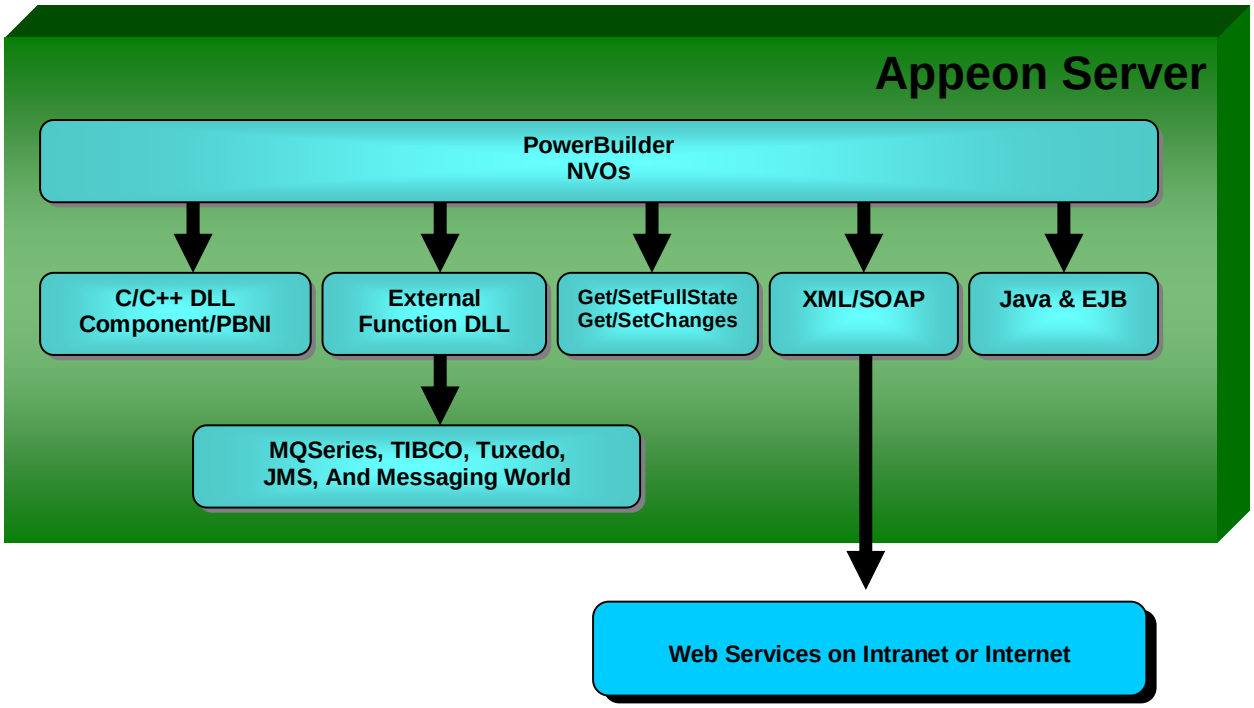
Appeon's built-in multilevel application security ensures that the system cannot be accessed by unauthorized users, even when deployed to many different users over public networks (Internet).

- Most existing PowerBuilder application security measures are automatically replicated in the Appeon Web application. This includes features such as specifying privileges for accessing particular menus, windows, functionalities within windows, and even DataWindow data (columns).
- Appeon adds a second layer of application-level security on top of the existing PowerBuilder application security. Application level security will authenticate users based on logon credentials (e.g. username/password and IP address) before allowing the user to logon to the application. The user access can be managed using an LDAP server or Appeon's on built-in system.
- Session timeouts can be easily applied to all Appeon Web applications by simply configuring a setting in the AEM (Appeon Enterprise Manager). This feature helps safeguard the application from unauthorized access when authorized users have stepped away momentarily or forgot to logout from the system.

2.6 Open and flexible J2EE/.NET integration

Appeon Pet World can be integrated with the following application types:

- .NET applications
- J2EE applications
- PowerBuilder applications
- Traditional Windows-based applications
- Other applications that support SOAP, IIOP, or PBNI protocols.



This flexible integration is made possible by Appeon's open middle tier that supports all leading industry standards for application integration.

- Appeon's middle tier supports SOAP and Web services. This relatively new but revolutionary standard enables application to be integrated with essentially any application type over the Internet and without having any knowledge of the internal workings of the application. PowerBuilder NVO Components as well as Java Components and C++ Components on the Appeon Server can be easily exposed as Web Services such that other applications can readily access the business logic inside of them. Conversely, Appeon Web applications can also consume Web services.
- Appeon's middle tier also supports IIOP and CORBA components based on the Java, C/C++, and PowerBuilder programming languages. As such, Appeon's middle tier can directly invoke methods of EJBs (Enterprise Java Beans), COM/ActiveX components, DLLs (Dynamic Linked Libraries), and PowerBuilder NVOs (PowerBuilder Non-Visual Objects) that are running on the application server locally. PowerBuilder NVOs can also easily access External DLL Functions of Windows applications to add in more integration possibilities.
- Appeon's middle tier also supports PBNI (PowerBuilder Native Interface), a set of C++ interfaces that enables C++ classes to harness the power of the PowerBuilder programming language and conversely enables C++ classes to be integrated into PowerBuilder/Appeon applications.
- Appeon's middle tier also supports passing DataWindow data and state between Server and Client including GetFullState/SetFullState/GetChanges/SetChanges for Composite, Crosstab, FreeForm, Graph, Grid, Group, Label, Nested, N-Up, and Tabular presentation styles. This allows Appeon Web applications to integrate with other visual and non-visual DataWindows external to the Appeon Web application.
- Appeon's middle tier also supports messaging through all standard messaging products including MQSeries, Tibco, Tuxedo, and JMS (Java Messaging Server).

3 Developer Productivity Comparison

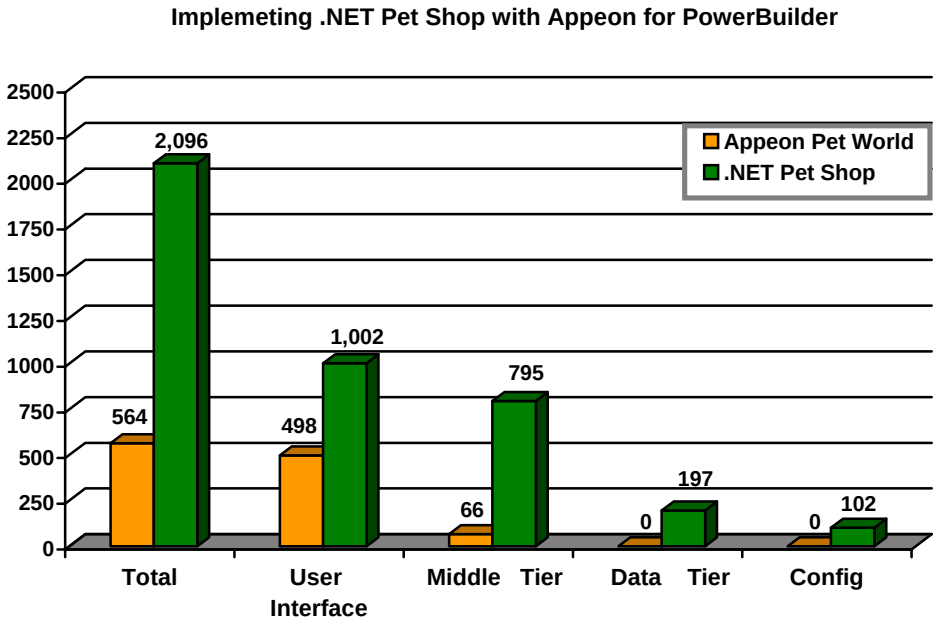
Appeon Pet World, which replicates all the functionality of the .NET Pet Shop, was implemented by an Appeon engineer in just 5 man-days and with 3.7 times fewer lines of code than .NET Pet Shop. The Appeon engineer only used standard PowerBuilder programming skills and Appeon for PowerBuilder to deploy the PowerBuilder application to the Web. In this section, we will compare the actual lines of code required of the .NET Pet Shop and Appeon Pet World implements. We will also analyze the Return on Investment (ROI) for these two solutions, .NET, and Appeon. We will leave it to the reader to extrapolate from this study and the .NET Pet Shop study in arriving to his or her own conclusions about how much higher the developer productivity and ROI is developing with Appeon instead of JSP and traditional J2EE tools.

3.1 Lines of code comparison

Appeon Pet World required developers to overall write 3.7 times fewer lines of code than .NET Pet Shop[1], with majority of the coding taking place for the presentation layer.

- The presentation layer required roughly 2 times fewer lines of code.
- The middle tier required roughly 12 times fewer lines of code.
- The data tier required no coding.
- The configuration tier required no coding.

[1] Appeon has based its analysis on Microsoft's .NET Pet Shop version 2.0: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnbda/html/psimp.asp



Please refer to Appedix1 for the methodology of counting lines of code in the Appeon Pet World and .NET Pet Shop.

3.2 Why is the Appeon code base so much smaller and more efficient?

There is a substantial difference in lines of code between the Appeon Pet World and .NET Pet Shop. This was realized by using PowerBuilder, the industry's leading open 4GL RAD IDE. Although ASP.NET has incorporated some RAD concepts, PowerBuilder naturally has much more. The key features of PowerBuilder that enabled Appeon engineers to minimize the Appeon Pet World code base was PowerBuilder's rich set of data-bound components including DataWindows, its event-driven programming model using OOP (e.g. encapsulation, polymorphism, inheritance) and class libraries (i.e. frameworks), and the 4GL abstraction with visual development. Appeon engineers focused exclusively on the workflow and business processes. Of course, this comparison between PowerBuilder and .NET only becomes fair and valid with the Appeon for PowerBuilder Web deployment option, which automatically deploys a PowerBuilder application to the n-tier Web architecture.

Aside from the productivity of a 4GL IDE and pre-built components, DataWindows played a very powerful role in reducing the size of the Appeon code base for the middle and data tiers. Most of the business logic and the user interface of the Appeon Pet World do information display, data query, and data update. This is an area where PowerBuilder offers great advantage over other tools, especially 3GL tools, through the patented DataWindow. The data-aware DataWindow object combines the visual presentation of the window with behind-the-scenes information required to retrieve/update the database into a single object-oriented control. DataWindows have an extensive set of built-in APIs (Application Programmable Interfaces), and can translate the data result into the latest technologies including HTML, Adobe PDF, and Microsoft Excel.

For example, by using powerful DataWindow functionality, Appeon engineers implemented the automatic total update for the shopping cart *without writing a single line of code*. The engineer defined the Subtotal price and Total price fields in the shopping cart as "computed fields" of a DataWindow. In the DataWindow painter, the developer defined Subtotal as "Price * Quantity" and Total as "…." When the user changes the quantity in the DataWindow object, that is, the shopping cart, the Subtotal field will be updated immediately when the focus is lost. The developer did not write a single line of code to implement this functionality!

Because of the extensive use of DataWindows throughout the Appeon PetWorld, the application does not require to developer to write many lines of code for the business logic and data access logic, which is why most of the lines of code is in the presentation layer. On the other hand, in the .NET environment, DataWindows do not exist, which explains there are so many more lines of code for the .NET middle tier.

3.3 The true measure of productivity

Measuring the lines of code does shed some light on the productivity advantages of Appeon over .NET; however, it does not provide the full picture. Building a client/server application is much simpler than building a Web application, even if the number of lines of code is the same. For example, there is no need to worry about breaking up the application workflow into static and stateless pages and managing the session/state information. Measuring the actual amount of time taken to build the Appeon Pet World and comparing to .NET Pet Shop is a more accurate measurement of productivity, revealing even greater effort reduction than the 3.7 times suggested by the lines of code comparison.

This productivity boost translates into superior ROI for projects, reducing the time, effort, and complexity of the project well beyond what is otherwise attainable. For sake of argument, the ROI calculation does not take into account the return associated with a shorter project cycle or lower complexity project. Instead, only takes into account the actual reduction in effort. As such, it represents the minimal ROI gains attainable with Appeon relative to .NET.

The referenced .NET study did not provide any hard details about the actual man hours saved. However, in the introduction, it mentioned that the entire project took 2 engineers and 4 weeks to complete. Assume a developer can write 100 lines of production code per day (8-hour workday). The roughly 2,100 lines of code of the .NET Pet Shop represent 168 man-hours of effort or roughly 4 man-weeks vs. the 8 man-weeks that was alluded to in the .NET study. The Appeon Pet World was built by one Appeon engineer working for one week, which represents 40 man-hours of effort. According to the comparison and assumption in the following table, Appeon would reduce the effort by 128 man-hours (168 man-hours – 40 man-hours), delivering an effort reduction of 128/40 or 320% vs. the 270% suggested by simply measuring productivity based on the reduction in lines of code.

Approach	Development Cycle	Effort in Man-hours	Effort Reduction
PowerBuilder/Appeon	1 developer @ 1 week	40	**320%**
Microsoft .NET	N/A	160	**128 man-hours or 16 man-days saved**

Assumptions:

1) A .NET developer can code 100 lines of production that is tested and free of bugs in a workday.

2) A workday is defined as 8 hours.

3) A week (workweek) is defined as 5 workdays.

The number of lines of code establishes the minimum productivity gains using Appeon. When the other productivity factors of a 4GL IDE come into play, there is even greater reduction in effort. Beyond productivity gains, however, Appeon enables you to do things that might not be possible at any level of investment (engineering effort), such as delivery on user expectations in a rapid time frame. Because of the rich client-server GUI supported by Appeon it becomes possible to keep promises that might not be feasible at all in a .NET or typical HTML based development environment.

3.4 Estimating the productivity gains for your project

3.4.1 Building a new Web application

The larger the application is, the more money, time, and effort customers will save using Appeon. For example, if we were to rebuild a small "real world" application with source code of 25 MB in size using .NET, it would require the .NET developer to write 26,085 lines of .NET code[2]. This is estimated by assuming each 1-Megabyte of PowerBuilder source code equals 282 line of PowerBuilder code, which corresponds to 1,043.4 lines of .NET code using the minimum 3.7 productivity factor. According to the comparison and assumption in the following table, Appeon would reduce the effort by 1,523 man-hours (2,087 man-hours – 564 man-hours) for this small "real world" project

Approach	Workload in LOC	**Effort in Man-hours**	**Effort Reduction**

[2] The 2.0-Megabyte Appeon Pet World application required the developer to write 564 lines of code using PowerBuilder. Using a linear extrapolation of 282 lines of code per Megabyte, it is assumed that a 25-Megabyte PowerBuilder application would require 7,050 lines of code. This study has shown that a .NET equivalent of a PowerBuilder application requires a minimum of 3.7 times more lines of code. Hence, the number of lines of code for the .NET equivalent of the 25-Megabyte PowerBuilder application is estimated by multiplying the 7,050 lines of PowerBuilder code by the factor of 3.7.

PowerBuilder/Appeon	7,050	564	
Microsoft .NET	26,085	2,087	**1,523 man-hours or 190 man-days saved**

Assumptions:

1) A developer can code 100 lines of production that is tested and free of bugs in a workday.

2) A workday is defined as 8 hours.

3) Each 1 Megabyte of PowerBuilder source code represents 282 lines of code.

4) Each 1 line of PowerBuilder code corresponds to 3.7 lines of .NET code.

How much effort would be saved for a 100-Megabyte PowerBuilder application? Based on the assumptions and calculations of the previous 25-Megabyte example, Appeon would reduce the effort by 6,091 man-hours (8,347 man-hours – 2,256 man-hours).

Approach	Workload in LOC	**Effort in Man-hours**	**Effort Reduction**
PowerBuilder/Appeon	28,200	2,256	
Microsoft .NET	104,340	8,347	**6,091 man-hours or 761 man-days saved**

Many factors go into calculating the ROI gains of using Appeon over .NET for your next Web development project. Some of the key factors include project schedule, effort, complexity, likelihood of failure, maintainability, and extensibility (e.g. deploying to a mobile platform). These examples should hopefully provide you with some basis for evaluating reductions in the project schedule and effort by using Appeon for your next Web development project. It is also important to note that PowerBuilder projects using Appeon can be instantly deployed to client/server architecture in addition to the Web architecture. These same projects can be seamlessly extended to PocketPC mobile device (via PocketPowerBuilder®) leveraging majority of the investments in the Web project and using only PowerBuilder programming skills. This study will leave it up to the user to assess the benefits of Appeon in regards to the other factors that improve ROI other than Appeon's productivity gains.

3.4.2 Webifying an existing PowerBuilder application

There is a greater productivity gain if Appeon is used to Webify an existing PowerBuilder application compared to rewriting the existing PowerBuilder application from the ground-up using .NET. A new Web development project does not leverage even one DataWindow, one line of PowerScript code, one control, one visual object, or non-visual object. Far greater savings can be realized by re-using parts or all of your existing application code. As such, if you intend to Webify an existing PowerBuilder client/server application, the reduction in effort and ultimately ROI will be even greater than that of a new Web development project using Appeon.

The reduction in effort will depend on the amount of rework that is required to deploy the existing PowerBuilder application to the Web. This rework is generally removing unsupported features or working around unsupported features using some other implementation. However, Appeon supports more PowerBuilder and DataWindow features than any PowerBuilder Web migration framework or Web development tool available. Appeon supports most PowerBuilder controls, DataWindow functionalities, syntax, and nearly the entire PFC framework. Even if you assume the worst-case scenario that you rewrite the entire application, you will still realize at least a 270-320% reduction in the effort relative to a .NET/J2EE rewrite.

4 Build Rich UI for your Web Applications

4.1 Rich UI = a better user experience

A poor user interface Web application is an excellent way to tell the world that you don't care about your existing customer base, partners, and employees. But most importantly, it says that you don't care about your company's ultimate success relative to the competition. It is no secret that the efficiency of a company's internal processes and the strength of its relationships with customers, partners, and employees are key factors that result in rise of one competing company and the demise of another. The company's applications, especially the user interface that the users must work with, plays a key role in all of this.

A common user experience in many applications: the user tries to purchase a product or service but enters mistaken data with good intentions. The customer clicks the "go," "enter," or "submit" button to trigger, let us say, the .NET code behind an ASP.NET Web page.

Rather than seeing a finished result the user is confronted, after a time-consuming "round trip" between his client machine and the server, with an error message, that complains that the user has made a mistake. In the worst case, the customer fixes the error but then the program code after the error finds **another** problem, and the customer has to re-enter more data, and proceed.

The user is also apt to be confused by an interrelated sequence of screens…to the extent that the user loses track of the "state" of his or her interaction with the site or application. In this case, especially where financial transactions and critical data is in play, the user may become confused enough to abandon the transaction or may not know whether a transaction has been accomplished. Duplicate transactions are very problematic and time-consuming to straighten out.

This poor user experience of the Web violates Rule Number One: The Customer (or user) is always right (Rule Number Two is, of course, that in cases where the customer is wrong, see Rule Number One). Many users may simply give up when a typical Web page repeatedly stops them from moving forward. At the least, the user will be very frustrated with the system. This may translate into a loss of a business transaction or opportunity with your customers and partners. If the Web application is internal to the company (used by employees), this translates into lower end-user productivity that increases labor costs and/or reduces capacity.

In Windows applications, the rich user interface provide a number of features which overcome the static, simple, single-functional, page-by-page workflow of Web pages and their forms:

- Event-driven user interface that is robust and dynamic
- Field level validation can take place when the field is changed, not later
- Rich data presentation styles, with a larger variety of shapes, colors, and patterns
- Powerful data entry mechanisms via advanced data-bound controls
- Data level refresh replaces the page-by-page refresh
- Overall, more functionality is available

The time-tested Windows user interface makes extensive use of rugged but complex controls such as listviews and treeviews, which allow the user to view and change structured data. The Windows user interface also saves "real estate" on the screen when a lot of information has to be displayed.

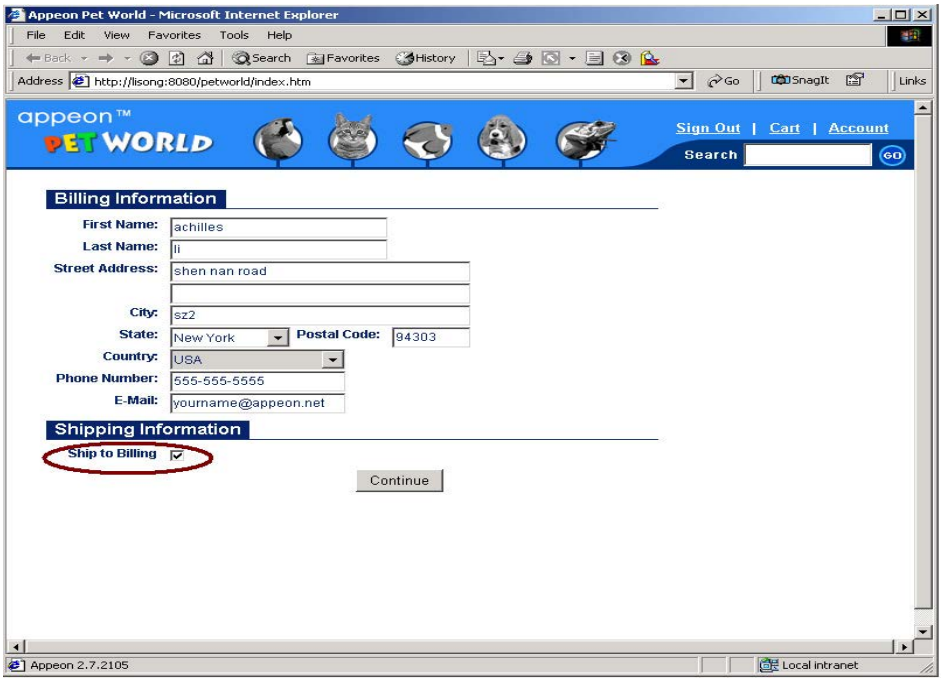
It is possible, usually with a huge investment in time, to "emulate" some of the rich functionality of Windows controls using .NET and using Java. However, after the massive amount of investment in time and other resources required, you would still be left with an inferior user interface that forces users to work page-by-page.

4.2 The Rich UI of Appeon Pet World

The Appeon Pet World illustrates a few of the many rich functionalities that can be deployed to the Web with few or no lines of coding and in a matter of minutes. In the Appeon Pet World, the examples are generally focused on data entry and presentation for Web forms (including shopping cart).

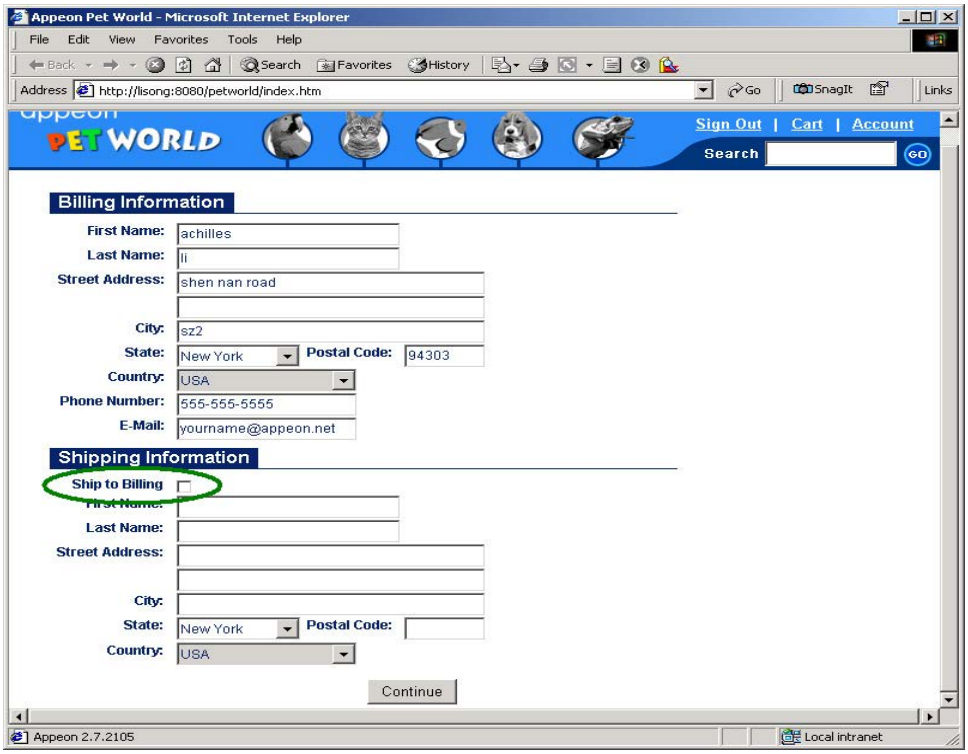
4.2.1 Dynamic forms

In Appeon Pet World, the forms can dynamically adapt based on the user input in a previous form without splitting the forms across multiple pages. In the billing/shipping information section, for instance, the shipping information is set as the same with the billing information when the default setting of the Ship to Billing check box (on) is selected.



If the user unchecks "Ship to Billing," the shipping information form will dynamically change to a blank fill form. This form is not a new page but a variant of the existing form that the user can enter new information.

In Microsoft's .NET Pet Shop, this feature is implemented as another separate page.



4.2.2 Automatic running total (update)

When the user works with the shopping cart, subtotal and total price of items added to the cart will automatically update if the user modifies the number in the quantity box and without a page refresh.

This cart is implemented as a PowerBuilder DataWindow. Therefore when the user changes any field in quantity column and presses Enter, presses Tab, or clicks outside of the Quantity field in the DataWindow, the subtotal and the total will be immediately refreshed with new values.

Compared with the Update button in Microsoft's .NET Pet Shop, the user operation in Appeon Pet World is much more flexible and user-friendly. The user is not required to click an "Update" button. The user is not forced to sit through a page refresh.

Appeon Pet World uses the processing capabilities of the client computer to perform calculations while the user is modifying information, and then saves the changes to the database at the server once they are finished. This avoids unpredictably slow round-trips to the server for recalculation.

4.2.3 EditMask functionality on the Web

True Edit Masks and Format Styles are rarely seen on web browsers because the JavaScript needed to implement them and the validation surrounding them is difficult to create and to run efficiently. However, PowerBuilder and Appeon make this once daunting web application development task simple and code free. PowerBuilder Edit Masks are properties; there is no coding involved. Appeon's patent-pending approach that provides a Rich GUI automatically takes dozens of PowerBuilder Edit Masks for Strings, Numbers, Dates, Times, and DateTimes and automatically moves them into a true web format and provides direct validation at the client-side so there is no need to round trip across the network to the server and back again.

4.2.4 Disabled and enabled controls

In PowerBuilder, developers can manage the state of controls in detail. Controls can be visible or invisible, or, alternatively, enabled or disabled (also known as "dimmed" or "grayed-out"). In the Appeon Pet World, you also have this same functionality. In the country selection, the DropDownListBox is set to USA and disabled. In some situations, it is preferred to disable a control but show its presence, instead of hiding it all together.

4.2.5 Page refreshes eliminated

When the user makes queries in the Appeon Pet World, the data queried will be retrieved and updated in Internet Explorer without reloading the displayed Web page. Reloading the Web page is necessary in traditional Web applications. In such a scenario, the client has to get all information in the new page from the server and display in Internet Explorer. The more data transmitted between the client PC and the Web server, the longer it will take for reloading and network/server loads will increase, sometimes to unacceptably high levels.

Appeon keeps the data transmissions to the absolute minimum. Only actual data displayed/entered on the screen is transmitted between the Web browser and server. In addition, there are no redundant transmissions of data. In other words, the user is not forced to wait to download a bunch of data that is already displayed on the screen just because one small number needs to change, such as the shopping cart total. These waits are one of the major sources of frustration for users and rob them of productivity.

4.3 Enriching Appeon Pet World

Appeon is capable of offering a far richer UI than that demonstrated in Appeon Pet World. With Appeon for PowerBuilder, engineers can take advantage of the rich palette of controls and functionalities in the PowerBuilder painter. This palette exposes an unprecedented rich graphical, intuitive, and friendly user interface on the Web. For example, Appeon Web applications can take advantage of real windows rather than pages, multi-level menus and toolbars, advanced Web controls like EditMask, Tab, TreeView, ListView, and robust DataWindow support. In short, Appeon enables the Web application to faithfully replicate the user interface of the corresponding PowerBuilder application using standard HTML and JavaScript.

Most of these features were not shown in Appeon Pet World because of the nature of the study. Using an online PetStore does not allow us to show off the rich UI that is demanded for enterprise applications that run a company's daily and mission-critical business processes. Another restriction was the developer productivity aspect of the study. Adding a richer UI would not have accurately demonstrated how much more productive Appeon is compared to .NET in building the same application. In the real world, since most of us are not building Pet Stores, the richness of the UI is up to our creativity and imagination.

5 Conclusion

Appeon brings the best of the client/server world and the Web world, running on a J2EE-based enterprise-class architecture. Enterprises enjoy the QoS (Quality of Service) of the n-tier architecture, standard technology, flexible integration, and strong security. Appeon Web applications can be readily integrated with J2EE, .NET and Web Services-enabled applications. 4GL client/server developers are instantly transformed into 4GL Web developers. End-users enjoy a rich Windows-style GUI that keeps them happy and productive.

Appeon future-proofs your project investments. An Appeon Web project can be instantly deployed to client/server at the click of a button. There is no recoding required. Using PocketPowerBuilder, the same project can be seamlessly extended to the Web. There is no need to have three separate projects. For example, a WinForm project for client/server, WebForm project for the Web, and another solution for mobile. One PowerBuilder project deploys to all devices (desktop, Web, and mobile) using essentially one IDE (PowerBuilder), and one skill set (4GL PowerBuilder programming with DataWindows).

The larger your project, the greater the benefit of Appeon. We encourage you to find out more about how much Appeon can save your organization.

Appendix: Counting the Lines of Code

All LOC (lines of code) statistics about .NET Pet Shop are referenced from the Microsoft's .NET Pet Shop Version 2.0 Whitepaper, using .NET to implement Sun Microsystems' Java Pet Store J2EE Blueprint Application.

Microsoft's .NET Pet Shop Whitepaper is available at http://msdn.microsoft.com/library/en-us/dnbda/html/psimp.asp.

Sun's Java Blueprints including J2EE Pet Store is available at http://java.sun.com/blueprints/.

Counting the lines of code in Microsoft's .NET Pet Shop

The code counted is code actually written by the developer and excludes the following:

1) Comments and empty lines

2) The user interface HTML code in the .aspx and .ascx files, since the code can be obtained by the drag-and-drop technology

3) The user interface . css files

Counting the lines of code in Appeon Pet World

The code counted is code actually written by the developer and excludes comments and empty lines.

Appeon Corporation
1/F, Shell Industrial Building
12 Lee Chung Street
Chai Wan District, Hong Kong
www.appeon.net