



## **Moving PowerBuilder® to EAServer—A Manager's Guide**

**How to look at existing PowerBuilder client/server applications, to determine when, why and how to move them to the web.**

## Introduction

The question of can and how do you move an existing PowerBuilder application to the Web has been asked many times over the past couple of years. I have personally been asked by every client that I have visited for the past 4 years how can they move their existing applications to the Web. One thing that I would like to make clear to all readers is that this is not a simple question with a yes or no answer or simply "just do this". I realize that this sounds like a typical consultant answer, but it really depends on the architecture of the application as well as where the code is written. The purpose of this white paper is to discuss the options that are available to move an existing PowerBuilder application to the Web along with the pros and cons of each approach. In addition some of the things to be aware of with each approach will be identified.

## Considerations

Before I address the various options that are available in moving an application to the Web let me reiterate that there is no simple answer to this question or is their a right or wrong answer. Each application is different as are the requirements to have the application web enabled. There are many considerations to be made when making this decision and some of these are listed below. These considerations are not listed in any particular order.

Consideration	Description
Is this a browser application?	
Number of expected concurrent users	
Expectations of the users	Do the users expect to see the same presentation styles as the client/server application? Do they want and expect a true Web application?
How large is the application in terms of windows, DataWindows, objects?	This will help determine the options that are available and how involved the migration will be.
Where is the code written?	Is the code placed behind controls such as buttons and windows or does it exist in non- visual objects?
What is the timeframe to complete the migration?	
Are large result sets of data presented to the users?	
Does the current presentation style lend itself well to a browser interface?	
Do your users understand that a browser interface is not a rich graphical user interface?	
What type of connections do the users have to the internet?	
What is the experience level of the development staff with PowerBuilder?	
What is the experience level of the development staff with Java?	

At this point I expect that many of you are thinking about your applications, the architecture, and where the code is placed. This is the first step in moving an application to the Web and a very important consideration. Depending on the architecture of an application, it may be easy to move to the Web or it may be faster to rewrite the entire application. If you have an application that is several years old and has grown significantly over the years it is difficult to throw it away and start from scratch. This paper will help you understand what options you have and get you started in the right direction.

## Options

There are several options to move a PowerBuilder application to the Web and each has its own merit. The considerations that are listed above are general and not specific to one option, however, many are related to browser applications. The term “Web-enable” implies a browser interface however this is not always the case. A distributed application may also have a need to be moved to the Web. There are several options that will be discussed on moving a PowerBuilder application to the Web and each is listed below.

Rewrite Using PowerBuilder

Rewrite Using Java™

Use Existing PowerBuilder Code

## Decision Making Process

Before discussing the rewriting and migrating options, a discussion on the decision making process is appropriate. My recommendation to clients is always to make the most informed decision possible. Do not take any one person’s opinion as absolute on the technological direction for your applications. Obtain several opinions and ask that each party making the recommendation provide a detailed explanation that your management and development staff can evaluate. This is recommended because if someone who has been writing PowerBuilder applications almost exclusively for 5 years makes a presentation, they will most likely recommend PowerBuilder. Likewise, someone that has been using Visual Basic or Java for 5 years will likely recommend Visual Basic or Java because that is what they are most comfortable with.

The purpose of this paper, as stated earlier, is to discuss the options available to moving an existing application to the Web. The intent is not to say that PowerBuilder is the only solution for application development, but to inform you about what can be done regarding PowerBuilder and the Web. If you are reading this paper then more than likely you have a large investment in PowerBuilder and you want to know if this investment can be leveraged, and if so then how. This paper will answer that question and give you an understanding of how to leverage your existing investment.

One point I would like to make is that with the technology currently available, one product or language does not have to be used for an entire application or for all applications within an organization. Organizations need to standardize on languages and tools, but there is nothing wrong with choosing several development tools. One mistake that I have seen many times is that an organization lets their languages and products be chosen based on information in trade magazines. The best example of this is Java, Java Server Pages™ (JSP), and Enterprise Java Beans™ (EJB). According to many hundreds and possibly thousands of articles over the past several years, every application needs to be written in Java, with JSP providing the interface and EJB providing the business logic. First of all, I am not saying that this design is bad. What I am saying is that not one architecture is correct for all applications. Each application needs to be evaluated to decide on the best architecture and programming. I have spoken with many people who make the statement “our application needs to be written in Java”. My question is “why”? Many people have done research and made a decision based on this research and their long-term direction. Unfortunately, the majority cannot answer my question of why does it have to be rewritten in Java. The correct answer is not “the xyz group has written their applications in Java and they work well”.

Make an informed decision, and one based on real-world research, not just hearsay. Also be cautious of anyone that immediately says “your application has to be rewritten” or “it is faster and cheaper to rewrite than salvaging your existing code”. These may be true statements, however, spend the time, and money, to do some investigating before choosing a potentially more expensive direction of completely rewriting your applications.

Now that the decision making process has been thoroughly discussed, let’s look at when it is appropriate to rewrite and what considerations should be made about rewriting.

## **Rewrite Using PowerBuilder**

Rewriting any application is always an option, and often the first choice given by a consultant. It is true that it may take more time to learn an existing application and then make modifications to Web-enable it than to rewrite it. However, there is an issue with taking existing software that works, and just throwing it away. A large application that has evolved over time has much business logic in the code that might be missed when a rewrite occurs. True this is part of the design process, however even the most detailed design specifications may miss some details.

I have been a consultant for over 12 years and have given the recommendation to rewrite on several occasions however, most applications I have reviewed can be migrated to the Web without being rewritten. If the decision to rewrite is chosen then there are considerations on whether to use PowerBuilder. This is not an easy decision, and while I know that PowerBuilder is an excellent product, a development team should investigate all options so that a well-informed decision can be reached. This is extremely difficult given the numbers of languages and products that exist. Some of the things to be considered include short-term goals, long-term goals, and the knowledge of the development staff in various products and languages.

One of the most compelling reasons to rewrite with PowerBuilder is to use all the experience of your development staff. If you have a large application and/or numerous applications that have been evolving over time, your development staff is very efficient in PowerBuilder and its techniques and that is not something that you want to throw away. You may have thousands of dollars invested in training and consulting costs that you do not want to throw away as well.

A second reason that goes together with the first is how quickly your staff can build applications with PowerBuilder. Any experienced developer builds applications more quickly with the tools he or she is most familiar with. In addition you may have an existing class library that saves incredible amounts of development time. The final reason to rewrite with PowerBuilder is that it is a great product!

If the decision to rewrite with PowerBuilder is chosen, there is one thing to be considered for your development staff. Unfortunately, many people consider a mature product such as PowerBuilder to be antiquated or out-dated. This can lead to turnover since many technical people want to be working on the latest technologies and not on mature products. Make sure that your developers are working with and are trained on EAServer as well as Java. When rewriting the application choose parts of the application that may make sense to write in Java and then write them in Java. This will give your developers real-world experience in Java as well as learning about the various things that EAServer supports. The developers will be immediately productive with the PowerBuilder portions of the application making end users and management happy with the results. At the same time the developers will be happy that they are learning EAServer and Java and do not feel they are only using a mature product.

## **Rewrite Using Java**

If the decision is made to rewrite the application and the long term direction of your company is Java, then using Java to rewrite is a good decision. Hopefully the tool you choose for your development will be PowerJ or JBuilder. Java is certainly the short term future of computing and only time will tell if it is the long term future as well.

Just because the decision has been made to rewrite the application in Java does not mean that all your PowerBuilder code has to be discarded. More than likely rewriting in Java indicates a Web or distributed application, in which case there will be an application server utilized. EAServer supports many different types of components including, but not limited to, Java and PowerBuilder. PowerBuilder business logic can be placed into EAServer components that are accessed from your new Java application! What this means is that your existing business logic can be utilized which will save development time and cost. The same recommendation that was made previously is made here which is to review existing PowerBuilder code in order to determine if and when the code should be rewritten in Java. Since Java is your long-term direction it will be achieved, however, faster results will occur by utilizing your existing business logic.

## Use Existing PowerBuilder Code

I saved this option as the last since it is what any PowerBuilder shop wants to know about, how to use their existing code. Management is especially interested in this option due to reasons of budget and time constraints. This decision, like all the other options, should be an informed decision and made for the right reasons. The most important reason is to leverage your existing PowerBuilder code and use that to get your applications to the Web quickly. Your development staff will have many new things to learn including HTML, JavaScript, Web Servers, and EA Server to name a few. This is no easy task and adding a completely new language such as Java just makes the learning curve that much steeper. PowerBuilder 8 can help with the learning curve with all the wizards and code that is generated for the developer. This includes PowerBuilder code as well as HTML and server-side PowerDynamo script. Version 9 of PowerBuilder will provide wizards that generate and support JavaServer Pages (JSP) which is an industry wide standard. A tool such as PowerBuilder creating code that can be reviewed by developers is a strong aid to learning Web development.

Your staff will need training and time to learn the new technologies mentioned above. This effort should not be underestimated and appropriate time and budget should be allocated for the initial project. Compare this move from client/server architecture to the move made from mainframe computing to client/server. There were significant learning curves for many mainframe developers to understand and become proficient in such as windows event driven model and object oriented programming. Expect the same type of transition and remember that there are more new technologies for the developers to learn and gain experience.

## Design Considerations

Now that the decision has been made to use PowerBuilder for your Web applications there are some things to be considered as far as the design of these applications. These considerations hold true whether this is a new application or an existing application being moved to the Web. This section of the document will discuss some common mistakes that occur when creating Web applications after years of client/server development. It will also identify some things that should be considered in the beginning of the development process. Many of these items are technical and deal with design but several are management issues. Both these areas need to be addressed in order to guarantee a successful migration to the Web.

## Common Mistakes

The same mistakes are made by many projects in many different companies. This section will identify these mistakes, discuss why they are mistakes, and offer possible solutions to help these be avoided.

### Unreasonable Timeframe

The first mistake made by many companies is to determine an unreasonable timeframe to migrate existing applications to the Web. There can be many reasons for this mistake and how to avoid these unreasonable timeframes is outside of the scope of this document. The key point to remember is that the timeframe must be based on the size of the application, expectations of the users, experience level of the developers, and the learning curves that will occur for that first project. The best approach to determining a reasonable timeframe is to spend the time to develop a good understanding of the tasks necessary for the project and how long each task will take. Obtaining help from experienced consultants who have successfully moved other applications to the Web can be valuable. It is nearly impossible for a developer to estimate how long each task will take without the experience of having done it several times before.

### Managing User Expectations

Many users expect that their application will look and work exactly as it did when it was a client/server application. This is more likely to be the case in a distributed application that is

not run in a browser. However, a browser interface is completely different than a Windows client/server application. The first difference is that a browser is not a rich graphical user interface (GUI) and it is limited in functionality. Graphics that appear on Web pages have to be downloaded from the server over the Web. Consider the connection speeds that the users that are accessing the application have and determine if these connections can handle large amounts of graphics.

The second difference is that a browser does not have a connection to the database. The main processing of data occurs on the server and not in the browser. Notice that many Web applications have a Submit button that is pressed to validate and process all the data at the same time. Many client/server applications validated each piece of data as it was entered. The reason a browser cannot process each piece of data as it is entered is that that requires that the HTML page be refreshed. The performance of each page refresh will not be acceptable if done as each field is entered.

Manage the expectations of your user community. This is absolutely necessary for a successful project. It involves working with the users on the design and educating them on how Web applications work. Once the users have an understanding of how Web applications work they will be helpful in the design and implementation phases of the project.

### **User Interface**

This section goes along with Managing User Expectations but is separate because of the technical discussion. Do not take your existing client/server GUI and duplicate it in a browser. As previously mentioned a browser is not a rich graphical user interface and is slow compared to a Windows application. In addition, many client/server applications implement Tab and Treeview interfaces. The Windows operating system has controls such as these as a native part of the operating system. Native means that these controls, and others, are easily created by an application. This is not true for browsers. Browsers do not have controls such as these internal. You can build a browser interface to look like a Tab or Treeview control, however it will not look and feel exactly the same. It will be similar, however it will not be exactly the same. Spend the time to learn how Web applications provide navigation and process business logic and then make your applications work in the same way.

### **Suggestions**

There are many considerations when designing any application whether it is a client/server, distributed, or Web application. This section will discuss some considerations for distributed and Web applications in order to help your development process get started in the right direction.

#### **Pilot Project**

Use a small application or part of an application as a pilot project to understand the development effort for all projects. Do not use something as simple as a reporting module, but rather use functionality that represents the overall functionality of your applications. Developing a small application that has several entry screens and reports can provide much knowledge of what can be done in a Web application and the best way to do it. Do not make the mistake of starting your first Web application with a very large application.

#### **Large Amounts of Data**

Since all the data and graphics will be sent over the Web as each screen is accessed, remember that screens need to be reasonably sized. This includes limiting the number of pictures on each screen as well as data. The rule of thumb is very simple. The more data that is presented requires more HTML be generated. The more HTML that is generated the larger the download that is required for the screen. The larger the download the longer the screen takes to load. This may take some design considerations to present screens to users in an efficient manner in order to provide the best performance.

## **Learning Techniques**

Allow your development staff ample time to learn Web application development techniques in order to design and develop the best application. This also includes learning how a browser processes pages and what settings will provide the best results. One example of this is JavaScript caching, which can significantly reduce the size of HTML downloads. This feature will store frequently used files on each computer the first time it is needed. These files contain common functionality that does not need to be downloaded every time the page is accessed. This is a feature that Sybase added with the Web DataWindow, which provides as much as 50-75% smaller HTML pages generated. Do not overlook the type of savings that a little bit of learning can help your application.

Developers can find many Web sites that contain tips and techniques in a variety of products and languages. Yes this means that they are browsing the Web while they are working, but what a better way to spend time working than learning new development techniques.

## **Conclusion**

This paper should give you some ideas on moving an existing PowerBuilder application to the Web or using PowerBuilder to build a new Web application. It does not answer every question about this process but should assist in the decision process to help make the correct decisions. Please feel free to contact Branick Consulting if you are interested in learning about our consulting and mentoring services.

## **About the Author**

Larry Cermak is the president of Branick Consulting, Inc. and has over 18 years industry experience, including working with PowerBuilder since version 2. He has been helping clients move PowerBuilder applications to the web for the past 4 years. Larry is a writer for the Sybase Developer Network, PowerBuilder Developer Journal, frequent speaker at conferences and user groups across the country, and the author of the Web DataWindow™ eBook.

Larry is the foremost authority on the Web DataWindow and has been working with the technology since it was first introduced during the PowerBuilder 7 beta and has worked with the Sybase DataWindow Engineering Team to introduce new features for the Web DataWindow. He has been training and mentoring clients all over the country on moving PowerBuilder applications to the web. Larry can be reached at [LCermak@branick-inc.com](mailto:LCermak@branick-inc.com) or (630)428-2650.

## International Contacts

<b>Argentina</b> +5411 4313 4488	<b>Korea</b> +82 2 3451 5200
<b>Australia</b> +612 9936 8800	<b>Malaysia</b> +603 2142 4218
<b>Austria</b> +43 1 504 8510	<b>Mexico</b> +52 5282 8000
<b>Belgium</b> +32 2 713 15 03	<b>Netherlands</b> +31 20 346 9290
<b>Brazil</b> +5511 3046 7388	<b>New Zealand</b> +64 4473 3661
<b>Bulgaria</b> +359 2 986 1287	<b>Nigeria</b> +234 12 62 5120
<b>Canada</b> +905 273 8500	<b>Norway</b> +47 231 621 45
<b>Central America</b> +506 204 7151	<b>Panama</b> +507 263 4349
<b>Chile</b> +56 2 330 6700	<b>Peru</b> +51 1 221 4190
<b>China</b> +8610 6856 8488	<b>Philippines</b> +632 750 2550
<b>Colombia</b> +57 1 218 8266	<b>Poland</b> +48 22 844 55 55
<b>Croatia</b> +385 42 33 1812	<b>Portugal</b> +351 21 424 6710
<b>Czech Republic</b> +420 2 24 31 08 08	<b>Puerto Rico</b> +787 289 7895
<b>Denmark</b> +45 3927 7913	<b>Romania</b> +40 1 231 08 70
<b>Ecuador</b> +59 322 508 593	<b>Russian Federation</b> +7 095 797 4774
<b>El Salvador</b> +503 245 1128	<b>Singapore</b> +65 370 5100
<b>Finland</b> +358 9 7250 200	<b>Slovak Republic</b> +421 26 478 2281
<b>France</b> +33 1 41 91 96 80	<b>Slovenia</b> +385 42 33 1812
<b>Germany</b> +49 69 9508 6182	<b>South Africa</b> +27 11 804 3740
<b>Greece</b> +30 1 98 89 300	<b>South Korea</b> +82 2 3451 5200
<b>Guatemala</b> +502 366 4348	<b>Spain</b> +34 91 749 7605
<b>Honduras</b> +504 239 5483	<b>Sweden</b> +46 8 587 70433
<b>Hong Kong</b> +852 2506 6000	<b>Switzerland</b> +41 1 800 9220
<b>Hungary</b> +36 22 517 631	<b>Taiwan</b> +886 2 2715 6000
<b>India</b> +91 22 655 0258	<b>Thailand</b> +662 618 8638
<b>Indonesia</b> +62 21 526 7690	<b>Turkey</b> +90 212 284 8339
<b>Israel</b> +972 3 548 3555	<b>Ukraine</b> +380 44 227 3230
<b>Italy</b> +39 02 696 820 64	<b>United Arab Emirates</b> +971 2 627 5911
<b>Ivory Coast</b> +225 22 43 73 73	<b>United Kingdom</b> +44 870 240 2255
<b>Japan</b> +81 3 5210 6000	<b>Venezuela</b> +58 212 267 5670
<b>Kazakstan</b> +7 3272 64 1566	<b>Asian Solutions Center</b> +852 2506 8700

For other Europe, Middle East, or Africa inquiries:  
+33 1 41 90 41 64 (Sybase Europe)

For other Asia Pacific inquiries:  
+852 2506 6000 (Hong Kong)

For other Latin America inquiries:  
+305 671 1020 (Miami)



### Sybase, Inc. Worldwide Headquarters

5000 Hacienda Drive  
Dublin, CA 94568-7902 USA  
Tel: +800 8 SYBASE  
www.sybase.com