Successful Web Application Design

Integrating with the web

by Jason Southwell

Web enablement, internet enabled, e-enablement... Whatever the term you want to use to define it, the race to the internet is on. In the recent world of VCs, IPOs and other acronyms that translate into big bucks, everyone has been interested in being first to the web.

As you may have seen, recent months have not been the best environment for this 'new economy' of the internet. Investment interest in internet projects has indeed been slowing down; however, most analysts would agree that this slowdown is only temporary. Before too long some new internet euphoria will hit, or a former one will see a resurgence.

Most recently the euphoria was in Business to Business (B2B) e-commerce. This may in fact re-surge as more and more success is found in the arena, or some other new internet technology might overtake it. Regardless, you most likely will once again be pushed toward the web.

So when you are asked to web-enable your applications, what exactly will it mean? Are you best served by creating a complete web application, or simply integrating internet features into your current application? Unfortunately, too often 'the management' will push you down any available path to the web, before clearly working out a plan and choosing the right technology to implement that plan. This can be disastrous if you, as the developer, have to re-write or re-work your web application every time you hit a fundamental design flaw (or when the management change their minds again).

Thoroughly thinking through the internet potential of your application can simplify any problems you might run into at a later date. Also, carefully considering other factors, such as development tools, technologies and methodologies, will help assure a successful project.

This article is intended to get you thinking about various web development issues before you get started. Whilst it will touch on some specific problem areas, generally speaking you will need to do more research to find out the full depth of the issues represented.

Defining A Web App

Web applications mean different things to different people. At its simplest, a web application is a software solution that utilizes the internet. Moving on from that stage, a web application can be a fully hosted application or data driven website providing mission critical business services. At its most complex, a web application might be a distributed process application utilizing resources from multiple machines around the world to achieve computational power that would otherwise be unachievable.

With the vast diversity in internet applications, the question really shouldn't revolve around what a web application is. Rather, the questions that should be asked revolve around what specific needs require a resolution? Or what existing resolutions to business needs can be improved upon? Identification of the real internet needs of your application is key.

Careful examination of these needs will indicate what type of web application you should develop. Developers will sometimes assume that they need to use a tool like WebBroker to create a web browser interface to their application in HTML before thinking through the other options. In many cases, simply adding internet features directly to your current application can better solve your needs.

Delphi, right out of the box, affords you the flexibility to design your web integration to what best fits your circumstances. Add to that the thousands of third-party components available for Delphi, and the integration possibilities are virtually endless. Specifically, internet component sets such as Indy (formerly Winshoes) are invaluable add-ons and will greatly enhance the internet integration potential for your application (Indy will be shipping with Delphi 6 and Kylix under the name Internet Direct).

Application Or Integration?

We have been using the term 'web application' up to this point rather loosely to mean both browser apps and e-enabled business applications. I believe that the term really can apply to both if the level of integration warrants the term. Wall Street is beginning to agree. In fact, in the B2B world, the money people on Wall Street are giving more and more acceptance to fully e-integrated business applications. For more information, check out the publication *B2B Or Not B2B* published by Goldman Sachs (www.gs.com), a global investment banking and securities firm.

Fully e-integrated business applications concentrate on enhancing productivity of the business via the internet. For example, financial applications might benefit from directly calculating currency conversions for their users. Integrating to an online service such as Yahoo Finance (find it at http://quote.yahoo.com/m5) could do that. Perhaps your software helps companies develop their business plans; such an application might benefit from automatically retrieving demographic information from the US Census Bureau (http://quickfacts.census.gov/qfd/index.html).

There are several organisations that already provide easy access to their services and content from the web either in HTML or XML and are easily integrated into either a web page or application. It is important to note, however, that if a website does not explicitly give you permission to use its data as part of your application, you are not legally allowed to use it.

On the positive side, many companies in the 'new economy' are willing to grant you access to that data, many times for free, but usually in return for some monetary or advertising concession. In addition, there are many other dot-com companies and government agencies whose sole purpose is to provide information (for example, www.everyone.com and www.iSyndicate.com). Services like these are generally specifically designed to be integrated into a website but, when you are using tools such as Indy, they can easily be integrated into actual business applications too.

Another form of internet integration has to do with connectivity of applications via the web. For example, productivity in your work order application might be vastly improved if job tasks could be sent to different repair stations and status information were quickly and regularly sent back to the originating station. This type of internet integration is simply an extension of a business application across the internet. In essence, a business is using the internet as a WAN of sorts. Many different industries and types of applications can benefit from this level of application integration and it need not be difficult to set up.

Avoiding Common Problems

Once you have decided upon an internet strategy for your application, it is tempting to plunge into development without thinking through issues that will cause frequent headaches at a later date. The rest of this article discusses what I consider to be the best steps you can take to avoid the more frequent and problematic issues I have run into in web development.

1. Use The Right Technology

We have already discussed that Delphi will most likely give you the integration you need.

If you have decided that your situation is best suited by a fully functional server side web application, Delphi may be your first reaction as well, and it is a fine selection. But don't forget to consider that Delphi might not be the right solution in *every* instance (oops, did I just say that?).

If platform independence is a necessity and a browser interface won't handle the complexities of your requirements, then you may have to look to Java. Java does have a place in this internet thing and, in fact, some companies are doing some really exciting things with it and should be looked at closely. Altio (www.altio.com), for example, has an amazing new interface for web applications. It uses Java on the back end, but its new user interface is really interesting. Of course, you also could wait for the .NET initiative at Microsoft to be finished and released... (OK, now I'm scaring myself. Let's get back on track.) Since Delphi is everyone's favorite language, let's look closer at its potential web technologies.

Regardless of the type and level of internet integration you decide to do, or even if you are building a web application from the ground up, Delphi is certainly versatile enough to meet most, if not all, of your needs.

There are actually many different types of web application that can be built with Delphi. You may already know about WebBroker and the Delphi interface to ISAPI, CGI and WinCGI. Applications using these technologies perform 99.9% of their business logic on the server side. They are actually web server extensions that extend the functionality of a standard web server application, such as IIS, Apache or Netscape. These applications are activated from the web server as requests are made of them from a website. Applications built as web server extensions generally interface with the user via a presentation layer created in HTML or XML (often combined with JavaScript or VBScript to perform that other 0.1% of the business logic) and viewed from a web browser.

These solutions are great for simple and static web page design. They are often used in more complex applications as well. However, when performing more complex processes, such as connecting to databases or doing time-intensive calculations, these technologies don't always scale well. Later in this article, I will discuss scalability issues in more depth.

Other technologies accessible from Delphi that can work well for web development include DCOM and CORBA. These technologies can work well if you intend to design an application from the ground up. DCOM and CORBA are geared specifically for n-tier development and are designed to scale better with process-intensive applications. The presentation layer of an n-tier application can easily be distributed for use over the web. We won't touch much more on DCOM and CORBA in this article, but there are plenty of resources on the web for more information.

Also, when talking about technologies, we cannot forget simple TCP sockets. Sockets in particular are an interesting solution for internet-enabling portions of your application, as discussed earlier. These, in fact, are my favorite way to integrate internet functionality. Sockets can be used at a lower level to develop your own data interface, or on a higher level by accessing common protocols such as HTTP to grab data from web servers everywhere.

2. Use The Right Tools

We already touched on Delphi's suitability in most circumstances, but there are also other tools that you must consider.

In particular, when concentrating on full web applications, the proper HTML editor will save you hours in the end. It is important that the editor you choose creates HTML that is compatible across the widest range of browsers. There are many choices but only a few that I would recommend.

A solid WYSIWIG editor is Dreamweaver from Macromedia. Dreamweaver is a good tool for working out your basic web page and site layout. Once the basic design is finished, it is a good idea to switch over to an editor such as Homesite from Allaire (www.allaire.com) for the detailed HTML tweaking. A common mistake for new web developers is to use Microsoft's Frontpage for all of their HTML work: Frontpage works well for Microsoft browsers and not much else, it can also trap you into needing a web hosting setup that supports the Frontpage extensions.

3. Design For Scalability

Several decisions you make early on in the development cycle can severely affect the ability of your internet interface to scale to a high number of users.

First, if your application needs to scale up seamlessly, stay away from CGI. A web server interfaces with a CGI application by launching an application instance for every hit to the web page. This causes additional latency for each hit as the application is loaded. This is fine for very many applications, but will not be suitable for very busy sites expecting large numbers of users per minute.

ISAPI is a better choice in these situations, as it loads the ISAPI library once the first time it is accessed. With each additional web hit, a thread is launched to handle the request. Thread creation is of course much quicker than launching a new application process, so it will scale much higher. Unfortunately, using ISAPI means that you are tied to IIS or Netscape servers and cannot use web servers like Apache or other cheaper solutions. It may also mean that you need to host your own servers, as many hosting companies will not allow you to run ISAPI applications.

Delphi's implementation of ISAPI does not implement the pending response code for IIS that would allow an ISAPI library to scale even better. Therefore, if you are looking for maximum scalability in an ISAPI DLL, you may have to alter the WebBroker source files to expose this feature.

Other issues may apply when designing for scalability. Regardless of the server interface, be it a web broker application or some other socket-based server, it is important to avoid critical sections and implement all work via threads. Using critical sections will serialize the work in the server thread and therefore force unneeded latency in each request. Potentially, some level of serialization will be necessary to control the upper limits of threads depending upon the hardware specifications of the web server. Also, some speed improvements can be made via thread pooling. Pooling threads can decrease the latency of thread creation by reusing worker threads when their work is done.

Basically, the scalability of your project will be inversely proportional to the time it takes to respond to a hit. The quicker the response, the higher you can scale. Keep this in mind when working through your design.

4. Test Your Application

This may sound obvious, but can be a much harder process than you would imagine. Full web applications themselves are much more difficult to test than integrated applications, especially if intended for widespread release. Users have different browsers with different capabilities and your application must work with the greatest number possible. You must also take into account differences between scripting language implementations in browsers. For example, Netscape is much more picky regarding JavaScript case sensitivity than Microsoft.

5. Stress Test Your App

This step is imperative. In limited release applications, scalability is not nearly as important, but if you release your application to the masses, you must test for stress. There are many good (and sadly expensive) testing tools that can help you with this. If you are looking for some free help from Microsoft, check out the Web Application Stress Tool at http://webtool.rte.microsoft.com/. It is a free download and a fairly functional testing tool. Unfortunately, Microsoft disavows any knowledge of its existence and if you call for support they will not help you. It is an easy program to get to know and has a nice help file to get you started. We use this to test all of our web server applications.

If you have a serious web application it is a good idea to get third-party verification of your scalability. Companies such as Mercury Interactive, the makers of LoadRunner, help to establish a successful testing strategy with you and are invaluable if you have venture capitalists whom you need to impress. Take some advice from someone who has learned from his mistakes... don't spend the money on one of these companies until you have *thoroughly* tested your application using the freebie tool from Microsoft.

Final Thoughts

Whether you are looking to write a full-scale e-commerce site for your company, or simply trying to enhance your database application with some internet features, some common design issues need to be taken to mind. The information in this article comes from my own experiences in developing B2B e-commerce applications. Delphi makes web development so easy that sometimes we forget to think things through before we move forward.

Hopefully you can take enough from this article to avoid some of the same mistakes that I made early on in my own internet development experience.

Jason Southwell is Director of Internet Technologies for ComponentControl.com and has been developing in Delphi for five years, on both internet and database projects. Contact him at jason@southwell.net