PDSA.com

Solutions for the Real World

PDSA Special Report

Desktop or Web Application

Making the Choice Between Desktop and Web

When you begin any new development project, you need to decide which technologies you will use to develop the application. There are a wide variety of languages to choose from, various database engines you can employ, and you need to choose a style of user interface to use. In all of these areas, you need to weigh a lot of disparate criteria when deciding on the best technologies for the task at hand.

To decide which language to use, you need to consider which languages you know and which language will let you perform the tasks that need to be accomplished in the application. For example, you would not use a .NET language like Visual Basic or C# to develop a device driver. Instead you would choose a language like C++, as it allows you better access to the hardware than does Visual Basic or C#. You would choose .NET over C++ to create a standard business application because the RAD (Rapid Application Development) environment lets you create this type of application much faster than C++.

When choosing a database engine, you will also weigh certain factors. You need to consider what database engine your client might already have in-house. You might look at how much and what type of data needs to be stored. You also want to consider how easy it might be to integrate with other databases, or to be able to replicate data from one server to another. All of these factors are weighed before you choose a particular database engine.

Although this document will not help you decide which language or database engine you should choose, it will help you make some informed decisions about your user interface. You will go through the same type of decision making when choosing your user interface as you do when deciding the language or database engine to use.

Some choices about the user interface you choose will be very clear-cut. Some choices will not be so clear, and you will have to weigh many different factors before making the final decision. For example, if you are building any of the following types of applications, you will definitely be using a Web interface.

- E-commerce store
- Company Web site on the Internet
- Remote users with no VPN capabilities but access to the Internet

Likewise, if you are going to be building any of the following types of applications, you will definitely use a desktop type of interface:

- Graphical applications, like games or CAD/CAM
- Word Processors
- Spreadsheets

Or if

- You need to integrate one or more products, like Word and Excel, within your application
- You need to integrate with hardware like POS systems, scanners, cameras, or medical systems (however, you can even sometimes accomplish this on a web app)

Of course, there are always those applications that don't fall neatly into the above categories. For example, if you are building any of the following types of applications, you could use either type of interface.

- Look-up systems like product catalogs
- Data entry systems
- Order processing systems
- Accounting systems
- Work flow management systems
- Charting/graphing applications

In these cases you need to look at many other factors before deciding on which method will suit this application the best.

Advantages to a Web User Interface

When you wish to target a web browser, it's probably because of the advantages the web UI brings to an application. Developing for a browser allows you to use standard HTML interface elements. Most potential users are familiar with browsing the Internet, and are used to how web pages look and feel.

You get the benefit of centralized application deployment and upgrading. Instead of having to roll out a new software application to hundreds or thousands of desktops, you only need to place the new software on a central server, and all users can access the software in one place. Similarly, when it is time to make a change to the system, you only need to make that change in one place. You can accomplish this deployment scenario with desktop applications by using a technology such as Click-Once and many other deployment tools.

If you have to create an application that remote users can use, it is much easier to export data through corporate firewalls over an HTTP interface than any other proprietary system.

Some applications take a lot of processing power. Instead of having to update each individual workstation, you can increase the power of your central server. This will be a more cost-effective solution than upgrading hundreds or thousands of workstations.

Disadvantages of a Web User Interface

Of course, developing for the Web is not without its limitations. The same standard HTML elements that are an advantage are also a disadvantage, as they can be somewhat limiting depending on the browser. Elements can appear differently to different users, depending on the browser that is used. This may cause some confusion if users move from one workstation to another.

Another problem with standard HTML elements is that they have no standard way of providing an input mask. A lot of users like knowing the format into which they should enter data. For example, a phone number can be entered in many different ways. In a desktop application, you can specify an input mask very easily; when using standard web controls, it is a little more difficult, as you may have to resort to JavaScript or jQuery on the client browser.

You will also have limited control over where screen elements appear on the screen. Standard HTML is not great at letting you positioning elements as easily as on a desktop application. This might make your screen design less elegant than you would get in a desktop application.

Performance can be slower in a Web application, as you are sending both the data and the screen design each time you hit an HTML page across an HTTP interface. In a desktop application, the interface is rendered locally once, and just the data is brought across a fast network connection. However, with many modern web techniques (Web API for example) you can now get speed that is nearly as fast as a desktop application.

It is not easy, with HTML, to go beyond the boundaries of the browser. This means that any integration with other products is difficult to accomplish. With Internet Explorer, it is much easier to integrate than with other browsers, but even then it takes a lot more programming to make it happen. In fact, at this point you are almost creating a desktop application anyway.

Advantages to a Desktop User Interface

When you program using a desktop UI, you get a much richer set of UI controls than you do with standard HTML. You can make your UI look like almost anything you want. You can make nice Tab controls; you can use input masking controls, editable grids, and many other controls that make creating a rich UI very easy.

When creating a desktop application, you can take advantage of the hardware that is already on the machine. You will need to do this if you are thinking of creating a game. Games typically need to interact directly with the video card, which is very difficult (if not impossible) to do with a browser. If you are developing a Computer-Aided Drawing (CAD) or Computer-Aided Manufacturing (CAM) application, you will also need to interact directly with the machine's hardware.

Performance is generally quicker on a desktop because the screen is drawn only once on the desktop and only the data changes. This prevents a lot of screen data coming from the server to the client, which increases the time taken to display the data. Getting data back across a network, as opposed to sending data across an HTTP interface, is also much faster.

If you need to integrate with any special hardware, such as Point of Sale (POS) systems, numerical control machines, printers, and scanners, it is much easier to produce a tidy interface using a desktop application than it is from within a browser. Browsers are such a closed environment that it is difficult, if not impossible, to load the DLL needed to interact

with this special hardware. You can use SignalR to sometimes work with these external devices from within a browser, but this might not always work.

If you need to integrate with other desktop products, you will find it much easier to do from a desktop application than from a browser. Many times you might need to exchange data with Word or Excel. This data exchange is easy in a desktop application but not so easy with a browser-based interface. The integration process between applications is hard to make generic to work across all browsers.

Disadvantages of a Desktop User Interface

Although you can do a lot with the rich UI support that a desktop application provides, you can also do too much. One of the problems with desktop applications is that each one can "feel" different. This leads to user training issues, and may cause some confusion when users use a variety of applications.

Desktop applications are sometimes hard to use if the user works from a remote location. Connecting to the main network can sometimes be a problem, as users have to deal with VPN software, firewalls, and a host of other confusing issues.

One of the biggest disadvantages of a desktop application can be felt when you deploy that application to hundreds or thousands of users. Imagine having to burn CDs or setting up a "push" server to distribute an application to that many users! This can be quite a job. And each time you have to do an update, you have to do it all over again. With Click-Once technology this process is much easier, but there are still challenges.

If users move from one workstation to another, they have to worry about whether or not the application is installed. Also, if one user sets up their machine with different colors, different Start bar location, or other "preferences," another user trying to use that machine will have a more difficult time.

Summary

There are many factors to consider when choosing the best method of development for a particular project. Even after weighing all of the alternatives, you may still decide you could go with either one. If that is the case, you might just wish to choose the one with which your users are most comfortable.

Contact Information

If you would like to know more about the information in this special report, please contact either Paul D. Sheriff or Michael Krasowski at PDSA.

Paul Sheriff

(615) 675-4632

PSheriff@pdsa.com

Michael Krasowski

(714) 734-9792 x223

Michaelk@pdsa.com

Company Information

PDSA, Inc.
17852 17th Street
Suite 205
Tustin, CA 92780

Tel (714) 734-9792
Fax (714) 734-9793
www.pdsa.com

PDSA.com
Solutions for the Real World