Introducing Microsoft Silverlight

Dragos-Paul POP
Faculty of Computer Science for Business Management,
Romanian – American University, Bucharest, Romania

ABSTRACT

Despite all the wonderful things you can say about HTML, CSS, and JavaScript, they form a pretty poor environment for developing modern sites and applications. If you care about your content working on most web browsers (or even just Internet Explorer and Firefox), accommodating their differences can be maddening. Many techniques and JavaScript libraries have been developed and shared over the years that can reduce this frustration, but none of them are silver bullets. In addition to browser differences, the graphical capabilities of HTML are too limiting for many user experiences that people want to create. Drawing a simple line, incorporating video, and a number of other things are extremely difficult or impossible with HTML alone. It's not that these technologies were poorly designed, but simply that they were designed for hyperlinked documents rather than the extremely rich presentations that most people want to create on the Web these days.

1. INTRODUCTION

Although the Web is easily the most popular environment for business software, there are some things that web applications just can't do, or can't do very well. Even if you outfit your web pages with the latest cutting-edge JavaScript, you won't be able to duplicate many of the capabilities that desktop applications take for granted, such as animation, sound and video playback, and 3D graphics. And although you can use JavaScript to respond on the client to focus changes, mouse movements, and other "real-time" events, you still can't build a complex interface that's anywhere near as responsive as a window in a rich client application.

Considering these issues, it's no wonder that Adobe Flash has been so successful.

Whether someone wants to create a professionally designed website, an online game (or any number of other applications), or even a simple advertisement, Flash has been a natural choice for escaping the limitations of HTML. The Flash development experience leaves much to be desired, however. Flash (the runtime environment, as well as the tool) suffers from the same basic problem as HTML: Many people are trying to use it for creating rich applications, but it was originally designed for something else (in this case, simple animations).

2. ABOUT SILVERLIGHT

Microsoft's new Silverlight is a direct competitor to Flash. Like Flash, Silverlight allows you to create interactive content that runs on the client, with support for dynamic graphics, media, and animation that goes far beyond ordinary HTML. Also like Flash, Silverlight is deployed using a lightweight browser plug-in and supports a wide range of different browsers and operating systems. At the moment, Flash has the edge over Silverlight, because of its widespread adoption and its maturity. However, Silverlight boasts a few architectural features that Flash can't match—most importantly, the fact that it's based on a scaled-down version of .NET's common language runtime (CLR) and thus allows developers to write client-side code using pure C#.

The heart of the plugin is the graphics subsystem, which supports a certain subset of WPF. It also includes the codes responsible for displaying audio and video content.

The architecture of Silverlight is quite complex, but it can be broken down into big chunks. The presentation system takes care of everything UI, including animation, text rendering, and audio/video playback. The plugin itself integrates into the browser so that the content can be shown, as well as accessed

using the JavaScript DOM. Finally, using some JavaScript code (or, optimally, the ASP.NET AJAX framework), Silverlight applications can be enriched to access server APIs like web services.

Silverlight 1.0 applications are created with a mixture of XAML (Extensible Application Markup Language), HTML, and JavaScript, so they are easy to integrate into existing web content and compatible with popular AJAX libraries and techniques.

The Silverlight plug-in has an impressive list of features, some of which are shared in common with Flash, and some which are entirely new and even revolutionary. They include the following:

- **Widespread browser support**: It's too early to tell how well the Silverlight browser works on different platforms. Currently, the builds of Silverlight 1.1 work on Windows Vista and Windows XP (in the PC universe) and OS X 10.4.8 or later (in the Mac world). The minimum browser versions that Silverlight 1.1 supports are Internet Explorer 6, Firefox 1.5.0.8, and Safari 2.0.4. Although Silverlight 1.1 doesn't currently work on Linux, the Mono team is creating an open-source Linux implementation of Silverlight 1.0 and Silverlight 1.1. This project is known as Moonlight, and it's being developed with key support from Microsoft.
- **Lightweight**: In order to encourage adoption, Silverlight is installed with a small-size setup (about 4 MB) that's easy to download. That allows it to provide an all-important "frictionless" setup experience, much like Flash (but quite different from Java).
- **2D Drawing**: Silverlight provides a rich model for 2D drawing. Best of all, the content you draw is defined as shapes and paths, so you can manipulate this content on the client side. You can even respond to events (like a mouse click on a portion of a graphic), which makes it easy to add interactivity to anything you draw.
- **Animation**: Silverlight has a time-based animation model that lets you define what should happen and how long it should take. The Silverlight plug-in handles the sticky details, like interpolating intermediary values and calculating the frame rate.
- **Media**: Silverlight provides playback of Windows Media Audio (WMA), Windows Media Video (WMV7–9), MP3 audio, and VC-1 (which supports high-definition). You aren't tied to the Windows Media Player ActiveX control or browser plug-in—instead, you can create any front-end you want, and you can even show video in full-screen mode. Microsoft also provides a free companion hosting service (at http://silverlight.live.com) that gives you 4 GB of space to store media files.
- **The CLR**: Most impressively, Silverlight includes a scaled-down version of the CLR, complete with an essential set of core classes, a garbage collector, a JIT (just-in-time) compiler, support for generics, and so on. In many cases, developers can take code written for the full .NET CLR and use it in a Silverlight application with only moderate changes.
- **Web service interaction**: Silverlight applications can call old-style ASP.NET web services (.asmx) or WCF (Windows Communication Foundation) web services. They can also send manually created XML requests over HTTP.

3. SILVERLIGHT ADOPTION

Silverlight is very new. For that reason, it's difficult to predict how well Silverlight will stack up against Flash's real strength: adoption.

At present, Silverlight is only on a fraction of computers. However, Microsoft is convinced that if compelling content exists for Silverlight, users will download the plug-in. There are a number of factors that support this argument. Flash grew dramatically in a short space of time, and Microsoft has obvious experience with other web-based applications that have started small and eventually gained wide adoption.

A key point to keep in mind, when considering the Silverlight development model is that in most cases you'll use Silverlight to augment the existing content of your website (which is still based on HTML, CSS, and JavaScript). For example, you might add Silverlight content that shows an

advertisement or allows an enhanced experience for a portion of a website (such as playing a game, completing a survey, interacting with a product, taking a virtual tour, and so on). Silverlight pages may present content that's already available in the website in a more engaging way.

Although, it's easily possible to create a Silverlight-only website, it's unlikely that you'll take that approach. The fact that Silverlight is still relatively new, and the fact that it doesn't support legacy clients (most notably, it has no support for users of Windows ME, Windows 2000, and Windows 98) mean it doesn't have nearly the same reach as ordinary HTML.

4. SILVERLIGHT VS FLASH

Some people refer to Microsoft's Silverlight technology as a "Flash killer," but I'm not sure whether that is really true. However, the similarities are striking. Both Adobe Flash (formerly Macromedia Flash) and Silverlight come as browser plugins. Both support vector graphics, audio and video playback, animations, and scripting support.

The technology basis is different. Flash uses a semi-open binary format, Silverlight is based on WPF. Before it was called Silverlight, the technology was codenamed WPF/E (Windows Presentation Foundation Everywhere). And thanks to good browser support, Silverlight can really be run everywhere, at least in theory.

In practice, the penetration of the browser plugin is a key issue. At the time of this writing, Silverlight plugins are available for the Windows platform (no surprise here) and support the two big players, Microsoft Internet Explorer and Mozilla Firefox. Also, a Mac OS X plugin exists that targets Safari and Mozilla Firefox on the Apple platform.

According to Microsoft, other platforms were investigated, but given that Windows has such a high market share in terms of desktop operating systems and Mac OS X is number two on that list, these browsers were given priority.

The most successful browser plug-in is Adobe Flash, which is installed on over 90 percent of the world's web browsers. Flash has a long history that spans more than ten years, beginning as a straightforward tool for adding animated graphics and gradually evolving into a platform for developing interactive content. It's perfectly reasonable for ASP.NET developers to extend their websites using Flash content. However, doing so requires a separate design tool, and a completely different programming language (ActionScript) and programming environment (Flex).

Silverlight aims to give .NET developers a better option for creating rich web content. Silverlight provides a browser plug-in with many similar features to Flash, but one that's designed from the ground up for .NET. Silverlight natively supports the C# language and uses a range of .NET concepts. As a result, developers can write client-side code for Silverlight in the same language they use for server-side code (such as C# and VB), and use many of the same abstractions (including streams, controls, collections, generics, and LINQ).

Let's take a look at some key points in the Silverlight vs. Flash war:

4.1 ANIMATION

The Flash format itself has no notion of animation other than transformation matrices. You can apply a matrix to an element on a per frame basis to move it around. Want to move something across the screen in 3 seconds? Calculate how many frames 3 seconds will take, then calculate the matrixes required for each frame along the way.

Silverlight supports the WPF animation model, which is not only time based instead of frame based, but lets you define the start and end conditions and it will figure out how to get there for you. No need to deal with matrixes. No need to calculate positions on various frames.

4.2 SHAPES

Flash stores its shapes using binary shape records. In order to write shape definitions, you will need to either license a 3rd party Flash file format SDK, or build your own. It isn't too difficult, but it does require a bit of a learning curve and the ability to manipulate things at the bit level, since shape records

don't align on byte boundaries. Needless to say, it isn't the kind of thing most people can write and have all debugged in one afternoon.

Silverlight uses XAML. XAML is text based and can be output using a simple XML object. No need to buy special libraries to write files. No need to write your own libraries. Just stream some text to a file and you're done--easily the type of thing that can be debugged and finished in an afternoon.

4.3 TEXT

Flash stores its fonts glyphs using the same exact shape definitions that are used for any other shape. The player itself does not understand TTF files, so you'll end up digging deep into the Win32 APIs and the fairly vague definitions in the Flash file format documentation to come up with something that sort of does the trick. You'll probably spend ages trying to deal with all the intricacies of fonts, because it turns out that typography is actually fairly complex… and you will have to deal with all those complexities yourself.

WPF/E lets you embed true type font information directly into your projects, and download that information with the downloader object. No need to do anything special. No need to handle anything yourself.

4.4 VIDEO/AUDIO

Flash supports multiple video formats. The latest codec is really high quality and the bandwidth usage is nice. There is one problem though if you are creating a tool that outputs Flash content… the formats it supports aren't really used by anyone else. The original video codec, Sorenson's proprietary H.263 implementation is a mutant version of H.263. The compression follows the spec fairly closely, but there are a bunch of features dropped out and you can't exactly just go find a complete spec on how to build your own encoder. The later codec from On2 puts you in an even worse position. Licensing Sorenson's codec isn't that expensive, but On2 will rape you with fees. They are relying on revenue from licensing the codec used by Flash to revive their $2 a share stock price. It is also a completely proprietary format (where at least the Sorenson one was loosely based on a standard). The audio formats Flash supports are all proprietary, except for ADPCM, which no one uses because of its horrible compression, and MP3, which is decent but dated, and still requires licensing fees and 3rd party conversion libraries.

Silverlight implements industry standard VC-1 codec for video, as well as offering support for WMV and WMA. Just about everyone already has Windows Movie Maker, but if they don't it's not a big deal. Why? Because Microsoft makes available a free Encoder SDK for producing WMA and WMV. So, not only are you using formats that people are more likely to be able to encode themselves, but Microsoft also provides your product with SDKs if you want to do the encoding yourself. The best part about it is that Microsoft doesn't rely on WMA/WMV licensing revenue to keep themselves alive, so not only is it easier to integrate, but it's also cheaper.

4.5 SCRIPTING

You can reuse C# classes from your tool inside your exported content. There is no development environment out there for creating real desktop applications which is based on ActionScript. If you go the Flash route, this means that all your classes and objects have to be written twice. You need .NET classes to handle the author time experience and Flash classes to handle the run-time. If you have server components, once again you need to switch back to .NET and throw out all the classes that the run time is using. For example, let's say you are creating a tool that outputs rich media quizzes. With Silverlight / .NET, the same entity classes you use to deal with results in the player could be reused on the server side. With Flash, you'd have to write all that logic 2x and keep it in sync as your tool changes.

4.6 TOOLS

You can create Silverlight content with the same tools you use on a daily basis. Visual Studio.NET is by far the most powerful and most popular IDE. You can potentially have all the code for the server components, the

authoring tool components, and the runtime/player components inside the same project. No extra skills required. No needing to hire some special Flash guru to do the graphics junk. Every developer can contribute to every part of your application.

5. CONCLUSION

Silverlight is a new technology that's evolving rapidly, and it's full of stumbling blocks for business developers who are used to relying on a rich library of prebuilt functionality. Not only does Silverlight lack any sort of data binding features, it also includes relatively few ready-made controls.

Some basics, like buttons, are relatively easy to build yourself. But others, like text boxes, are not. At present, Silverlight is primarily of interest to developers who plan to create a highly graphical, complete customized user interface, and who aren't afraid to perform a fair bit of work.

REFERENCES

[1] Matthew MacDonald, "**Silverlight and ASP.NET Revealed**", Apress 2007

[2] Christian Wenz, "**Essential Silverlight**", O'Reilley 2007

[3] Adam Nathan, "**Silverlight 1.0 Unleashed**", Sams 2008

[4] OnFlex, http://www.onflex.org/ted/2007/04/m-silverlight-vs-adobe-flash-player.php

[5] ASP .NET http://weblogs.asp.net/jezell/archive/2007/05/03/silverlight-vs-flash-the-developer-story.aspx

[6] MySites Advisor, http://blogs.mysites-advisor.com/index.php/2007/06/03/silverlight-vs-flash/