Solving Legacy Browser Dependencies in the Enterprise with a Single Browser Solution

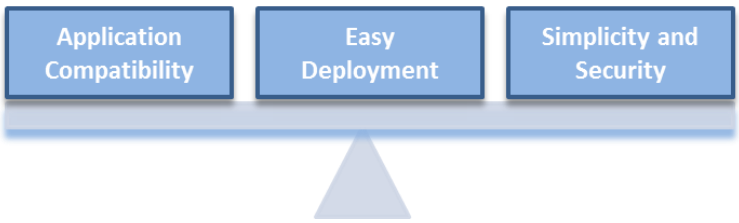
Executive Summary

Line-of-business applications have become increasingly web-based in recent years, moving away from traditional 'heavy client' applications. This makes the web browser an essential part of day to day functions for the modern information worker. Significant growth of line-of-business (LOB) web applications occurred early in the new millennium when usage of Internet Explorer 6 (IE6) was particularly strong. Consequently, the custom and off-the-shelf web applications deployed by most organizations were built for, and ultimately dependent on, Microsoft's proprietary and now outdated and insecure browser platform. Additionally, legacy web applications tend to rely on other legacy plug-ins for full functionality, including older versions of Java and Adobe's Reader ActiveX controls.

During the past decade, increased competition helped the browser market mature, drove innovation and solidified the broad adoption of web standards. The latest versions of all major web browsers are built around these web standards – not proprietary platforms – so many older web applications no longer work properly. Until recently, keeping those old applications working meant staying with older browsers, creating redundant, virtualized infrastructure to support older browsers, or re-writing all IE6-dependent applications. Those painful choices have made it difficult for many enterprises to adopt new technologies that make the organization work faster and better. With Windows XP end of support slated for April 2014, the pressure is on for Enterprises to upgrade.

It has been difficult to upgrade software platforms while continuing to use existing web applications which were built for previous versions of Internet Explorer. Attempting this is tantamount to replacing your car's engine while driving down the highway. It's possible to do, as long as you have two cars running side-by-side, and you're able to jump safely between them. (You'll also need one heck of a good mechanic.) Similarly, many organizations are stuck

with outdated browser plug-in requirements like older versions of Java or ActiveX controls used in ERP systems. Adding complexity to the business decisions around browser management is the need for IT organizations to strike a careful balance between running older, less secure programs (like recently exploited versions of Java) and settings to maintain web application compatibility or deploying newer, more secure technologies that break line-of-business requirements. Until now, IT organizations typically were forced to choose between security and compatibility.



IT decision-makers have a number of options available to help balance these requirements, and they are looking for a solution that delivers compatibility, easy deployment, simplicity and security.

Delivering on these pillars directly leads to reduced total cost of ownership, and an increased return on investment for LOB applications.

Browsium Ion delivers in each of these areas. This lightweight, highly configurable browser add-on enables IT staff to decouple decisions and requirements regarding upgrading browsers and operating systems. Ion allows organizations to upgrade to Windows 7, using IE8, IE9, or IE10 while still maintaining compatibility with their IE6-dependent line-of-business web applications – ensuring they control their own software deployment schedule. Legacy IE6-dependent web applications continue to work, the organization gets the immediate benefit of enhanced security and performance of a modern browser offered through IE8, IE9, or IE10 – but it isn't necessary to deploy parallel infrastructure to virtualize browsers or operating systems or perform any other company-wide changes until desired. With Ion, employees continue to use web applications as they always have, via a single browser, maintaining continuity and avoiding unnecessary training costs. Ion runs natively on Windows 7, as well as Windows XP, Windows Vista, Windows Server 2003 and Windows Server 2008.

The Need to Upgrade

Microsoft confirmed at their Worldwide Partner Conference in July 2011 that, while Windows 7 has been a huge success overall with 400M licenses sold, enterprise deployments have been painfully slow. According to Tami Reller, CVP and CFO of the Windows Business at Microsoft, "Two-thirds of business PCs are still running Windows XP." Nine months later, in the spring of 2012, there has been little improvement, with Gartner reporting that nearly 80% of desktops and half of enterprise laptops in large enterprise are still trapped on Windows XP. Many of these Windows XP PCs are still running IE6, which first shipped in that operating system. The continued use of Windows XP perpetuates the use of IE6, with the web metrics company Net Applications pegging IE6 usage share at 7% as of March 2012 - making it the fifth most-used browser version. IE6 usage is much higher in enterprise, where usage often doesn't register on Net Applications' tracker, bringing the total user base to high as 150 million PCs worldwide.

Enterprise usage of IE6 is particularly strong given the predominance of IE6-dependent applications. Enterprise hesitance to migrate is reinforced by a Gartner study indicating that 40% of businesses browser-based applications don't work with IE8. This reality makes it difficult for organizations to upgrade despite the myriad security and compatibility issues associated with IE6. Companies have been in the unenviable position of exposing themselves to security risks (and financial and productivity costs) to maintain compatibility, or engaging in an earlier-than-planned upgrade cycle, including modifying or rewriting their IE6-based web applications. The latter option also has financial consequences: In the same study referenced above, Gartner estimates that 20% of organizations will run over time or over budget on Windows 7 migration projects due to IE8 compatibility issues.

The impending expiration of Windows XP and IE6 in April of 2014 compounds the pressure on enterprises to migrate to newer platforms quickly.

Microsoft itself strongly encourages businesses and consumers to upgrade from IE6, going so far as to create The IE6 Countdown website with a countdown clock to help drive the move away from IE6 by shaming regions around the world that still have high IE6 market share.

To summarize, corporations are squeezed by the following:

- Browser technology and web applications have made significant advances since the release of IE6 a decade ago. This has left IE6 behind the curve in terms of security, functionality, support for current web standards, and general compatibility with modern sites on the Internet.
- Microsoft has indicated that it is focused on more current releases and thus won't be devoting resources to IE6 compatibility, as evidenced by the fact that Windows 7 doesn't include IE6 in "Compatibility View" or run the full version of IE6, and IE9 isn't available for Windows XP.
- Modernizing line-of-business web applications will take time and money – and cannot be done incrementally. Organizations wish to address this daunting task on their own schedule.

Organizations therefore are looking for a solution that allows them to run a modern version of Internet Explorer, provides IE6 compatibility, and enables them to update their line-of-business applications on their own timetable. The balance of this paper will discuss these objectives.

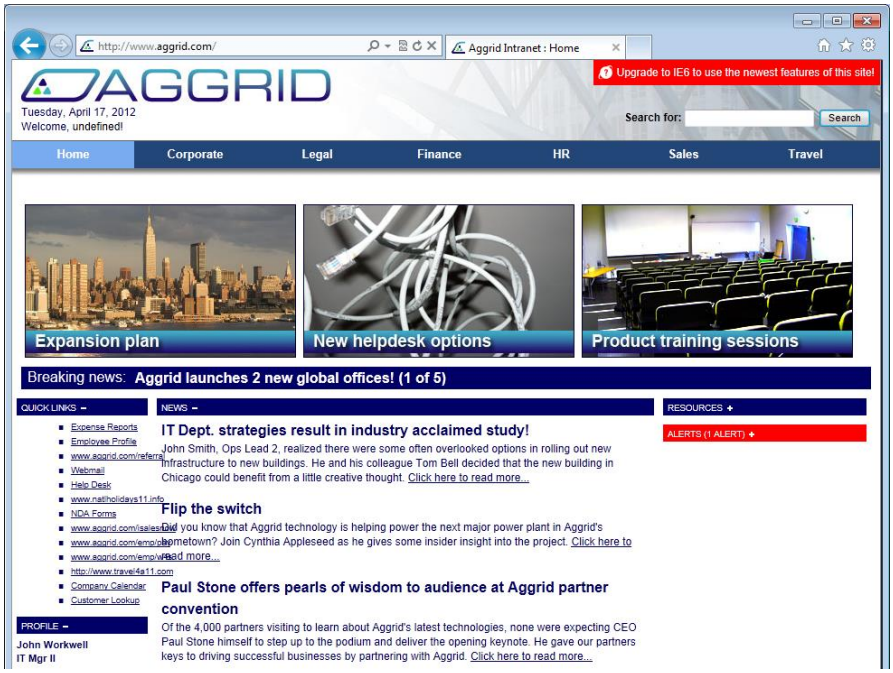
Web Application Compatibility – A Barrier to Operating System Upgrades

The application compatibility issue for IE6-dependent web applications is based in part on the fact that IE8 – IE10 utilize an updated rendering and scripting engine (or browser platform) that is incompatible with IE6. The browser platform governs many browser attributes, spanning design issues such as where a button appears and what part of the button is clickable, to the execution of web application code. Some of these issues, while annoying, don't greatly impact productivity – but many others can make a web application completely non-functional, causing significant business disruption.

Additionally, web applications designed for IE6 use many older ActiveX controls, some of which don't run in more recent versions of Internet Explorer, present security risks in themselves, and don't have the capability to offer side-by-side installation, so users end up with an updated control that causes their web application to not function properly, or worse, crash when they access it.

Example of an IE6-Dependent Web Application

The screenshot on the following page, taken from a sample web application built explicitly for IE6, illustrates the end-user experience of accessing an IE6-dependent web application in IE8 – IE10.



Sample IE6-dependent site rendered in IE9

While it is easy to see the many obvious visual issues in the adjacent screenshot (such as overlapping columns), there are functional problems on the page as well. Though not visible in the image, the pull-down navigation menus along the top of the page cannot be clicked, making many features on the site inaccessible. Collapsible and expandable elements on the page (such as the 'Resources' section header) do not function, instead yielding JavaScript errors. Also notable on this page is the banner at the top right asking the user to upgrade to IE6 – a common problem with web applications hard-coded to look for IE6 and blocking key features, if not the entire application, if IE6 is not detected (even when accessed with a newer version of Internet Explorer). There are many more issues with this application, which can be experienced by accessing the site at www.aggrid.com using IE8 - IE10.

Binary Settings Force Security/Compatibility Compromises

Beyond ActiveX controls and rendering, organizations also need to address deeper issues regarding their browser configurations. Many Internet Explorer settings are binary (they're either all on or all off for the entire browser application) and different applications require that a given setting be set differently – so "fixing things" for one application can break another. Among the negative effects of this reality is the fact that this negates the value of some of the important features in IE8 – IE10.

For example, DEP/NX Memory Protection is an important Windows security feature that prevents code that is marked non-executable from running in memory – a common attack vector of malicious hackers. DEP/NX protection shipped in Windows XP SP2 but was not enabled by default in Internet Explorer until IE8. Because DEP/NX is not enabled in IE6, many web applications written for IE6 took liberal advantage of memory without worrying about the ramifications of running non-executable code. Those same applications crash instantly when run in IE8 – IE10. Worse, the DEP/NX toggle switch in IE8 – IE10 is binary, so it can only be turned on/off for the entire browser instead of per application.

This forces companies to turn it off completely in order to enable their legacy line-of-business applications to work properly – a tremendous security compromise to achieve legacy application compatibility.

Alternatives to Achieve Application Compatibility

Organizations that are not ready to replace their legacy web applications with expensive new systems and solutions need to find ways to cost-effectively balance their need to maintain compatibility, leverage new technologies, and minimize impact on end users - a challenge for any enterprise IT organization. There are a variety of options available – unfortunately most of them are neither cost-effective, nor practical. This next section will analyze a few of the common alternatives.

Virtualization

In some circles, virtualization is proposed as a solution to ensure compatibility while deploying a more modern operating system and browser. Virtualization is used to simulate an underlying hardware or software infrastructure, making it possible to run multiple operating systems and services per physical computer, provide system isolation for stability, and support disaster recovery and system restore capabilities. From an architectural perspective, these virtualization capabilities can be deployed in a variety of ways including: Desktop Virtualization; Application Virtualization; Server-side Virtualization using Terminal Services, and Server-side Virtualization using VDI. With Terminal Services, end-users access a shared copy of the operating system, while in the VDI scenario, each end-user accesses their own copy of the operating system running on a hypervisor on the server. Each differs from the others in terms of technology and implementation, but the most commonly used solution for legacy compatibility with IE6-dependent web applications is the Terminal Services option.

While virtualization in its various forms can be an effective tool to meet many objectives, it is a complex and expensive option if the primary objective is limited to delivering web application compatibility. Virtualization requires IT to create one or more virtualized operating system environments, which by proxy would include a virtualized browser. For example, Windows Server 2003 with Terminal Services running IE6 can deliver a Windows XP-like experience with native IE6. Or Microsoft MED-V can deliver a complete VDI-based Windows XP + IE6 experience. These users could then be upgraded to Windows 7 and access the virtualized Windows environment, utilizing a separate IE6 browser window, if not an entirely separate desktop environment, when they needed to access legacy web applications.

Using a virtual operating system can deliver many important benefits – but to deploy a virtualized solution simply to enable access to incompatible web sites and applications is a bit like cracking an egg with a sledgehammer: You can make an omelet, but it's very messy. Yes, the virtualization solution will deliver compatibility, but the cost and complexity are high because an entire parallel infrastructure is being deployed simply to provide legacy web application compatibility.

Consider these factors:

- You must purchase a license for every operating system from Microsoft. So if a typical user requires four different browser environments, you need to buy four Windows licenses (or Terminal Services CALs). This drives the IT CAPEX cost per user staggeringly high.
- After deployment, IT must secure, patch, and configure each virtual operating system and each browser. Many of these operating systems and browsers require specialized knowledge and tools to administer, increasing both support and bandwidth costs merely to support an operating system that should be retired.
- Most virtualization solutions require investment in additional servers and improved network capacity to support the virtualization infrastructure, as each server can only support a small group of concurrent users and bandwidth needs can be substantial. This may include a large number of costly new servers, expensive virtualization software, specialized management software, and upgraded client PC hardware to handle the workload.
- Providing multiple functional environments at the desktop almost guarantees user confusion, resulting in more user training and increased help desk costs. For example, users could easily:
    - Forget which browser they need to use for which LOB application
    - Choose the wrong browser, leading to errors or security issues
    - Remain accustomed to IE6 and continue using it for all browsing due to inertia
- Windows XP support ends in April 2014, whether running natively or virtualized using MED-V.
- Support for Windows Server 2003, which enables a Terminal Services solution to run IE6, expires in 2015, so one end-of-support problem is being traded off for another right behind it.

While virtualization can provide important benefits to the enterprise, web application compatibility can be accomplished more effectively through other means, and in a way that complements virtualized solutions that are deployed to accomplish other objectives.

Application and Web Site Rewrites

While many organizations have plans to eventually reengineer, redesign and replace their legacy IE6-dependent web applications to suit newer technologies and business practices. Re-writing applications merely to gain parity with their existing platforms is an unproductive and expensive strategy – both in financial terms and in terms of opportunity cost, as the organization has less flexibility to use technology to innovate and improve competitiveness. Some companies estimate that re-writing just one IE6-dependent application to function properly in IE8 – IE10 can cost millions of dollars. Compounding the problems – and increasing the costs – is the possibility that the developers who wrote the code may no longer be working for the organization or be available for legacy application maintenance work, may be out of business (if a 3rd party was used), or may rely on technology that is no longer supported. Furthermore, IT groups often don't have managerial control over development teams, so IT has limited ability to manage priorities and schedules.

Consequently, many organizations live with these legacy applications far longer than originally anticipated, trapping themselves on legacy operating system and browser platforms. In effect, they're making the ultimate tradeoff of compatibility for security, choosing to run decade-old browser

technology (IE6) in order to keep their legacy applications working as the financial alternative of recoding seems a far worse tradeoff.

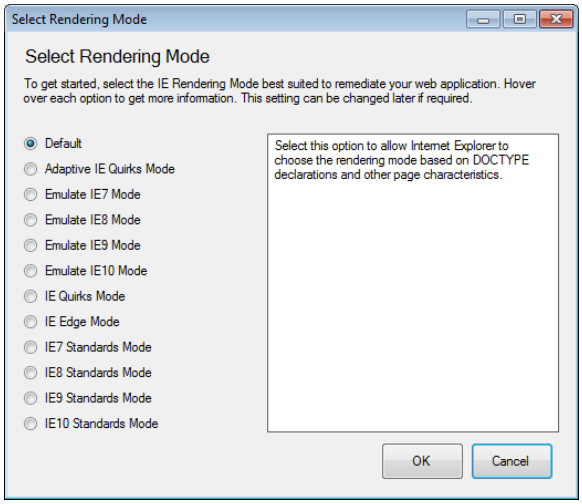
Manual Browser Configuration Management

As mentioned previously, manually configuring browser settings to solve compatibility issues is a complicated endeavor. Some problems can be fixed by changing compatibility or security settings, but many can't. Some settings fix one issue but are incompatible with other applications. Every "fire" that IT puts out seems to create flames somewhere else. In a nutshell, web pages and applications simply aren't incompatible *in the same way*, with the new or planned version.

In an ideal world, an administrator could analyze the incompatibilities, determine which features and settings cause that incompatibility, make configuration changes, and then use those same changes with each client browser. Unfortunately, it is rarely easy to identify problematic settings, and to have the desired fix work across all apps and environments.

The Browsium Ion Solution

Browsium Ion enables legacy IE-dependent web applications to run modern versions of Internet Explorer, paving the way for large-scale enterprise migration to Windows 7 while continuing to leverage existing web application investments. This legacy web app remediation solution avoids the cost and complexity of virtualization for web application compatibility – but can seamlessly complement those solutions when they're deployed to meet broader IT objectives – such as upgrading to Windows 7 via VDI.



Browsium calls its approach "Web Application Continuity". It extends the ROI of already deployed and paid-for applications by freezing the platform for web applications at a "known good state". It also provides granular control over all Internet Explorer settings, freeing customers from the painful tradeoff of security vs. compatibility. With Ion, enterprises can control Internet Explorer settings at the web application, or even the web page level, maximizing compatibility for all business-critical applications while simultaneously maximizing security for Internet browsing – all within a single-browser solution that is completely transparent to end-users. No other solution provides this level of compatibility and control.

Ion meets the needs of any organization that has to support legacy web applications, but is eager to upgrade to more current versions of Internet Explorer. It addresses the issues mentioned above in the following ways:

Application Compatibility

- Renders IT-specified sites using legacy rendering modes built into IE – all other sites use modern IE defaults
- Adaptive Quirks boosts compatibility by intelligently choosing between Quirks and IE7 standards rendering
- Inject or replace HTML, JavaScript or CSS from client in real time – no server-side code changes
- Create custom registry, security and feature control settings per app or even per web page
- Run legacy, current versions of Java side-by-side

Easy Deployment

- Works on any PC running IE8, IE9 or IE10 – no client upgrades needed
- Requires no new infrastructure or additional servers
- Uses the tools you have in place today – deploy client software via any software distribution system, Rules & Profiles via Group Policy
- Rules & Profiles can also be saved as XML files for flat-file distribution to client PCs ActiveX controls can be invoked without installation on client systems, easing deployment and improving security

Simple and Secure

- Transparent to end users – no need to switch between separate environments or launch virtual machines
- No user training needed – Rules ensure sites use required components no matter how they're accessed
- Improves security by enabling organization-wide upgrade to a modern, secure browser
- Use of legacy IE modes and settings contained to only websites under your control – all other sites rendered with most current, secure settings

The Way Forward – a Single Browser Solution

The fast pace of operating system and web browser development makes it difficult for an IT staff to both keep pace and maintain desired compatibility. They need to provide robust collaboration tools, govern access to internal and external sites, and ensure a secure desktop for users, all within a tight budget. Custom browser applications provide necessary business functions, but in some cases their platform requirements hold back the deployment of newer technologies that are also necessary. While there are a number of solutions available today that could help address this issue, many of them introduce unnecessary complexity and cost to deliver web application compatibility, and others become too technically complex to manage and implement.

Ion is the most viable and cost-effective solution available today because it is a single–browser solution, completely controlled by the IT organization. It supports existing investments in business applications and infrastructure, while allowing IT to improve security on every PC. End users don't need to learn anything new, and if desired, Ion can be utilized as part of a broader VDI solution for desktop operating system upgrades.

Learn more at www.browsium.com.

Email sales@browsium.com for pricing and JumpStart Program information.

Browsium, Inc.
8201 164th Ave. NE, Suite 200
Redmond, WA 98052
+1. 478.227.6973