

WHITE PAPER

Microsoft, PHP, and Open Source: A Pragmatic Alliance

Sponsored by: Microsoft

Al Hilwa

July 2009

EXECUTIVE SUMMARY

Microsoft has recently intensified its investment in interoperability with open source technologies in a variety of ways, not least of which is its support of PHP in Windows Server, the SQL Server database system, and other products such as the Expression Web development tool. While Microsoft's efforts are motivated by the desire to break adoption barriers for its Windows platform, the company is building bridges to the world of open source in a way that many of its customers and partners welcome.

IDC OPINION

Scripting languages such as Unix shells as well as the sophisticated Perl played an important role in the early evolution of the Internet, but a much simpler alternative, called PHP, eventually came to rule the genre as the key glue that binds many of the Internet's new applications. Microsoft was able to offer an alternative architecture from the early days with its Active Server Pages (ASP) technology, which later evolved to the more flexible and more complex ASP.NET. However, .NET is officially supported only on Windows, and a great deal of the world's Web serving runs on Linux and Unix, where PHP has become the de facto alternative. It was thus only a matter of time before Microsoft cast a bigger net to better support PHP and other open source technologies. Our research into Microsoft's efforts in this area has revealed the following:

- ☒ The latest release of Internet Information Services (IIS7) runs PHP faster and with more scale due to the implementation of a FastCGI gateway. This feature is also available as an add-on on IIS6.
- ☒ A new Microsoft SQL Server driver for PHP is available to support more native features in the SQL Server database and to increase overall performance and reliability of PHP applications wishing to use SQL Server as the database.
- ☒ PHP is competently supported by Expression Web 3, one of Microsoft's offerings targeted at Web designers and developers.
- ☒ Microsoft has begun to contribute to open source projects and to expand its ecosystem to open source communities that want to use non-Microsoft tools and frameworks that interoperate with the Microsoft Web platform.
- ☒ Microsoft's main play into cloud services with its Azure operating system is promising even greater interoperability with PHP and other open source technologies.

TABLE OF CONTENTS

	P
In This White Paper	1
Situation Overview	1
Running PHP Better with IIS7	2
Key Features and Benefits	3
Areas of Improvement	4
A New SQL Server Driver for PHP	4
Key Features and Benefits	4
Areas of Improvement	5
Developing PHP Applications with Expression Web	6
Key Features and Benefits	6
Areas of Improvement	7
Microsoft and Open Source	7
A Shift Is in the Air	8
Microsoft Open Source Technology Center.....	9
Port 25.....	9
CodePlex.....	10
Project Mono Collaboration	10
Moonlight and Eclipse4SL	10
Windows Web App Gallery.....	11
Microsoft as Open Source Contributor	11
Getting on Microsoft's Cloud	12
PHP on Azure.....	12
Challenges and Opportunities	13
Conclusion	14
Case Studies	14
MindTouch.....	14
PHP on Windows by Demand	15
SugarCRM.....	15
Leveraging the Microsoft OS and Database.....	15
Appliedi.....	16

IN THIS WHITE PAPER

In this White Paper we examine the most important developments that bridge the PHP world and the Microsoft platform. As PHP has become more popular, the demand to run more scalable PHP applications on Windows has increased. Technology limitations have been fixed to enable this scenario. We discuss the new capabilities that now allow PHP to run on Windows on an equal footing. We additionally discuss the broader context of open source applications and Microsoft's new attitude toward the open source community. Finally, we explore the implications of Microsoft's new cloud operating system, Azure, for interoperability with open source software.

SITUATION OVERVIEW

PHP is a language used in the Web tier of applications. Originating as a scripting language to put together simple HTML code that is sent to the browser for rendering, it has grown over the years to acquire more sophisticated functionality, much improved performance, and a considerably expanded base of developers. PHP, for example, now features an object model that has been improved and expanded in several releases. While new programming languages, especially for Web-tier client or server development, have continued to emerge, we have seen PHP pull away from other scripting languages such as Perl, Python, or Ruby as the most frequently used scripting language on the server side of Web applications. Despite being derided for the difficulty of writing maintainable applications, for its interpreted performance compared with compiled languages such as Java, or for the lack of strong integrated development environments (IDEs) available for it, PHP has continued to increase in popularity. This can happen to simple languages that grow progressively more sophisticated, attracting an ever larger community of developers and applications, but it can be amplified for open source processes that allow creative contributions from a broad base of interested developers. Studies conducted by IDC over the past five years have regularly shown that PHP is the language spoken by the largest number of Web developers. This critical-mass dynamic has led to an active community that has evolved PHP in a vigorous manner and in an ecosystem of both open source and commercial add-ons, including better frameworks and IDEs.

Most importantly, as PHP has matured, we have seen it selected in many applications of scale (e.g., Digg, Flickr, Friendster, YouTube) at higher frequencies than in its earlier years, often displacing more general-purpose languages such as Java. This phenomenon is not unusual as it aligns with long-running trends toward a higher abstraction level in programming and toward more dynamic languages that defer type bindings to execution time and allow the liberal and expressive mixing of objects of different types in the code. Scripting languages as a rule are designed to glue together other pieces of code and integrate multiparadigm approaches in programming. They are almost always dynamically scoped (i.e., with less rigid typing compared with compiled languages and with looser type enforcement and automatic implicit conversations on the fly). These characteristics lead to smaller code volumes as much of the noisy bookkeeping code thickening required by less dynamic languages becomes implicit and handled by the runtime in an automated fashion.

It is natural to see scripting gain popularity as hardware becomes more powerful since scripting languages are typically interpreted in a more resource-intensive and less optimized fashion compared with compiled languages (i.e., interpreted language instructions are performed without translation or compilation to either machine code or virtual machine code). At the same time, scripting languages have been evolved aggressively to support many of the large-program constructs of traditional programming languages (e.g., object orientation) while allowing them to retain their flexibility in operating as language glue for smaller chunks of code. Additionally, efficiencies of adoption scale are at work for all programming languages that cross certain critical-mass thresholds of developer availability, accumulated knowledge and practices, and volume of applications and code amassed. Language evolution often proceeds along a classic adoption bell curve until new technologies or architectures evolve that call into question the premise of the language, causing new ones to begin a new curve. While this is typically a slow process, often playing out over a couple of decades, now we are definitely seeing the fungibility of scripting languages, such as Ruby and Python, with compiled languages exemplified with Java. It should be noted that Java belongs to a class of languages that while compiled (just-in-time compiled) are virtual machine based with a defining characteristic that they internalize memory management into an asynchronous garbage collection system that operates automatically as part of the runtime. The .NET family of languages, such as C#, belongs to the same class, known as managed languages. Thus, while some workloads move from managed languages to dynamic languages, other workloads move from traditional compiled languages (also known as 3GLs, or third-generation languages) such as C and C++ to managed languages. In short, what used to be implemented in C/C++ is now often done in Java or C#, and what used to be implemented in Java or C# is now often implemented in PHP and similar scripting languages.

While this shift to dynamic languages has also progressed in the .NET space, the effect has been somewhat less pronounced because .NET has been evolved as a set of classes to offer a progressively higher level of functionality and abstraction. Additionally, the .NET toolset, Visual Studio, has also evolved into ever greater sophistication allowing significant productivity gains through error checking and debugging that has taken the edge off the shift to scripting. C#, for example, has evolved to accommodate the opportunistic requirements of Web applications while remaining rich and productive for larger programs. Nevertheless, we have observed increased interest in running PHP on Windows due to the sheer weight of adoption of PHP. Additionally, Microsoft has noted that Windows has not captured as much of the production workload of PHP applications even though it is often the primary development platform for PHP applications. Largely as a result of this dynamic, we have witnessed Microsoft mobilize to better support PHP workloads in multiple touch points of the stack. Today we are beginning to notice some of the fruits of this effort in terms of increased adoption. This paper describes this work along with a few case studies on users of the technology.

RUNNING PHP BETTER WITH IIS7

One of the most significant improvements to the state of PHP on Windows comes about with IIS7, which shipped with Windows Server 2008 (and Windows Vista SP2). IIS7 is a significant revving of the IIS code with some specific enhancements aimed at

improving the adoption of Windows Server in the hosting community. One of the focus points of the release was to enable PHP to run at a higher level of performance and isolation in order to attract PHP applications either by design or at the hosting level to run on Windows. Running PHP well is a crucial factor for hosting providers, many of which run blended environments with Windows and Linux side by side. Offering better PHP means that Microsoft can persuade these providers to run larger Windows farms and offer it on an equal footing with Linux. Hosting being a highly optimized offering with carefully calculated costs and profit margins, hosters would offer PHP on Windows only if it made sense from a performance and reliability perspective.

In the corporate space, where Web serving may be run by IT organizations, it is rare today to find a Global 1000 company without a significant deployment footprint on Windows server, and most operate massive farms of virtualized servers of different types of Web servers, ready to provision to various parts of the corporation the optimal solution for the application at hand. Many thus run a significant chunk of their Web tier on IIS, often hosting ASP applications, and like the flexibility to run PHP workloads on Windows as an option to balance out other considerations if the economics support this scenario. With the release of IIS7 and the FastCGI add-ons to IIS6 and IIS5, this scenario is now in play.

Key Features and Benefits

The key enabling technology in IIS7 that has provided the much-needed improvements in how PHP runs on Windows is the implementation of the FastCGI interface in IIS. FastCGI is a gateway technology that improves over CGI by avoiding process start-up times that have historically been the costliest part of running isolated PHP applications. FastCGI technology enables this by reusing the same processes for new PHP requests and performing the requisite cleanup to ensure that new requests are isolated properly. FastCGI relies on the new application pools functionality in IIS7, which effectively improves any scripting technology that is designed to exploit this feature. In fact, the native Microsoft scripting technology, ASP.NET, also benefits from the new application pools functionality.

Without the FastCGI improvements, PHP applications are left to one of two alternatives, known as CGI or ISAPI, both of which are less than ideal due to fundamental differences between Windows and Unix/Linux process architectures. In the CGI case, PHP requests are run one process at a time, undergoing the unnecessary overhead. In the ISAPI case, a direct link to the IIS code is established without process forking, but since the application runs in the server context, there are some risks if it is not the best-behaved piece of code. For most decision makers, this is not a reasonable choice, and many have avoided making it by not using Windows IIS at all.

The reason this choice is presented in the first place can be traced to the fact that Windows does not offer a construct exactly equivalent to the Unix process. Windows processes, while architecturally similar to Unix/Linux processes, are considerably more heavyweight and resource intensive and simply more time-consuming to start. Windows threads, which are intended for lightweight uses, run in the context of the calling process and thus do not provide the typical Unix process protection and isolation.

Most languages and interfaces designed for Unix/Linux systems use processes liberally. When these applications are ported to Windows, a hard decision has to be made on whether to use processes or threads. Thus, historically, most PHP applications ported to Windows have developed a reputation for being either too slow or too unstable compared with running on a Linux Web server (in reality they are usually at most either one or the other, depending on whether CGI or ISAPI is chosen).

The implementation of FastCGI in IIS presents a significant leap forward in the ability to run PHP applications on Windows. Reports from the field indicate that this capability has enabled many ISVs to offer their PHP-based applications to their customers and partners who are reporting considerable improvements in performance and stability. At the end of this paper, we profile two popular ISV applications and a hosting services provider that support PHP on Windows.

Areas of Improvement

It is early in the deployment cycle to get deep feedback on what might be additional improvements in the IIS architecture to better support PHP, but the early indication is that the product has met its requirements. Nevertheless, the development team is collecting feedback, and one observation is that IIS now does not respect time-out settings on HTTP requests and is hard-wired at 30 seconds.

A NEW SQL SERVER DRIVER FOR PHP

Until recently, even if you ran PHP on Windows, you were unable to exploit Microsoft SQL Server effectively. Most PHP applications are written with a MySQL database back end in true adherence to the LAMP order, but the state of drivers available from the community has historically lacked the performance needed to run high-scale applications. A number of open source drivers were available, such as FreeTDS, but they were not aggressively supported and did not typically support the most recent SQL Server product features. This situation changed in July 2008 when Microsoft put out a production release of the new SQL Server Driver for PHP v1.0. What's more, this driver was released in open source on Microsoft's CodePlex site, making it possible, at least in theory, for an enterprising developer to compile it for a Linux or a Mac OS X platform, for example.

Key Features and Benefits

Any functioning driver would have been a significant improvement over the status quo, but one that is produced and supported by Microsoft is truly welcome by the PHP customer base that runs its Web tier on Windows. As we discussed earlier, established and mature Windows Server operations, including for running the Web tier of applications, are present in almost all of the Global 1000 companies or governmental organizations of similar size. In the upper midmarket tier and midmarket, the concentration of Windows as a standard only increases with a large section of the midmarket running almost exclusively on Windows Server as a preferred enterprise standard platform. For these companies, SQL Server is increasingly also an enterprise standard and the deployment of other databases can be problematic due to the lack of support contracts or management resources. These are the primary targets of the PHP driver for SQL Server.

Since most applications on PHP tend to be designed and written to use MySQL, one of the key abilities is the support of the usage and access patterns of MySQL databases. The Microsoft driver team was aware of this as it appears to have made it a priority to ensure that the driver is functional in most of the basic scenarios for MySQL database access. Thus the driver simplifies coding for accessing PHP streams to handle content applications typical on the Web and exploits connection pooling features that have been available in the underlying ODBC technology for some time. The driver is available at no additional cost to all SQL Server users, including the free Express edition. Microsoft's additional motivation behind this investment is to potentially attract MySQL users to SQL Server in the highly competitive database market.

The key benefits of the new SQL Server driver revolve around the enablement of the rich data management feature set of the SQL Server database for PHP application developers. In addition to capabilities introduced in SQL Server 2005 for handling large database tables, such as table partitioning, the new version of the database, SQL Server 2008, adds further new features for large data sets such as data and backup compression and performance-enhancing capabilities such as "Partition Aligned Indexed Views." SQL Server 2008 also adds several new improvements in the area of security such as data auditing and transparent data encryption capabilities. A new policy-based management framework and a resource governor allow fine-grained control over how database resources are allocated to competing database workloads. Last but not least, interesting capabilities to support streamlined installation and hot-add CPU and memory to the hardware increase uptime and minimize application disruption. Application developers should take a look at the capabilities of SQL Server 2008 because it offers a variety of productivity-enhancing features for developers such as FILESTREAM data support and spatial data type and Virtual Earth integration support.

Areas of Improvement

Developer feedback has been generally positive on the new driver. The excitement around Microsoft's engagement in the PHP space has brought requests for additional features or capabilities to support new applications. One concern that seems to have come up in the community is the driver's reliance on the mature SQL Native Client technology (also known as ODBC). Some have suggested that ODBC is not the most current API for Microsoft technologies and as such the solution is not strategic. In fact, ODBC remains the standard programmatic mode of accessing SQL Server for nonmanaged code (e.g., code written in C/C++ languages). It just so happens that new applications are increasingly being developed with the .NET languages, primarily C#, which use a different driver architecture due to their layered just-in-time compiled approach. Thus this particular concern is largely unwarranted as Microsoft has introduced new releases of the ODBC/OLEDB driver with SQL Server 2005 and SQL Server 2008 and will debut a new release with Windows 7. We expect Microsoft to continue to provide ODBC drivers for a long time to come.

On the substantive side, the key areas of requested improvement to the Microsoft drivers revolve around essential features necessary to port specific applications to SQL Server. Many PHP applications are internationalized and thus demand Unicode support in UTF-8 format, which is the standard Unix/Linux format. Additionally, there

have been posts regarding cursor behavior and the need to support larger BLOB sizes. Finally, it should be pointed out that PHP developers have a preference for using the PHP Data Objects (PDO) extension architecture for data access and thus have asked Microsoft to implement a PDO-compliant version or wrapper of the driver. It remains to be seen if Microsoft can prioritize all of these improvements, but it is clear that Microsoft has now allocated resources to evolve this driver, and we expect the key improvements that will enable the largest section of applications to be worked on first.

DEVELOPING PHP APPLICATIONS WITH EXPRESSION WEB

Another touch point with and overture to PHP and Web standards technology from Microsoft is the new professional Web design tool known as Expression Web 3. This second iteration of one of the key products in the recently debuted Expression toolset (first introduced in 2006) brings Microsoft into the world of Web design and development using the standard Web protocols and languages. Expression Web represents an evolution of the now obsolete FrontPage Web design product combined with the best capabilities of Visual Studio for Web design, offered in an environment specifically tailored toward the professional Web designer. Expression Web 3 is able to handle both ASP.NET 3.5 and 2.0 technologies, including ASP.NET AJAX as well as Web standards such as HTML, XML, XHTML, CSS, and PHP.

Key Features and Benefits

Unlike its predecessor, Expression Web 3 is a PHP development tool that is a serious challenger to competing development tools in this space. Some of the key features provided by Expression Web 3 for the PHP developer are as follows:

- ☒ **Testing PHP code.** Unlike the relatively complex Apache or even IIS, Expression 3 comes with a built-in Web server that requires no configuration or even access to the corporate network. The built-in Web server also provides integration with the browsers installed on the machine to do simultaneous testing of PHP code in multiple browsers preconfigured with different screen sizes, as appropriate for the project. An integrated FTP client in Expression Web 3 allows the developer to upload code for testing or deployment to a remote Web server while also comparing version/date information.

- ☒ **PHP includes handling.** The ability to include PHP files with precoded parts of a Web page is now supported with Expression Web 3. Additionally, problems encountered with the first release of Expression Web related to byte-order marks have been eliminated by providing developer control and flexibility over which files use or default to using byte-order marks. These new features allow Expression Web 3 to work smoothly and predictably with PHP Include files. Additionally, simply navigating from the Insert menu allows PHP snippets to be made into new PHP Include files with little effort.

- ☒ **IntelliSense and color coding for PHP.** IntelliSense, one of Visual Studio's key productivity features available in Expression Web 1 for ASP.NET development, is now available for PHP in Expression Web 3. Similar to code-completion features of other IDEs, IntelliSense provides lists of methods or attribute options as text is typed in the editor, not only saving lookup time but also reducing error rates and acting as a training tool for developers getting up to speed on PHP. In addition to IntelliSense, Expression Web 3 provides color syntax highlighting and floating error messages for PHP code, also long recognized as a hallmark of coding productivity.
- ☒ **Cascading style sheets (CSS) support.** In addition to the ability to create new style sheets and to automatically link new or existing style sheets, Expression Web 3 provides a rich way of presenting CSS hierarchies that can help disambiguate these overlapping and often conflicting style sheets. A new task pane for managing style sheets is also provided, bringing this complex part of Web page design into the realm of the manageable.

Areas of Improvement

Expression Web 3 is now a formidable player in the PHP development space with a commendable newfound passion from Microsoft behind the support of Web standards. It is a worthy competitor, perhaps lacking nothing more than maturity, richness of templates, and broad designer adoption when compared with its major rivals. With time, most of the drawbacks are likely to be fixed, but one area requires more of a strategic shift than a tactical feature introduction, namely multiplatform support. A significant segment of Web designers and developers works on other platforms, especially the Apple Macintosh; for these designers, Expression Web is only a contender with the help of virtualization technology on the Macintosh. Another concern for the overall Web development offering that Microsoft is making to open source developers is the forcing of a difficult choice. Another product available in the Microsoft portfolio, called Visual Web Developer Express, is available free of charge and offers strong support for client-side standardized constructs such as HTML, JavaScript, and CSS but leaves out PHP, while Expression Web 3 is a for-fee product with strong support for PHP. This is an area that Microsoft may need to revisit as it better positions its tools portfolio. The current positioning of the two products puts Expression Web more in-line of Web designers who are unlikely to use Visual Studio, while Web Developer Express is viewed as an on-ramp towards fuller Visual Studio development. Microsoft has the opportunity to integrate PHP into its Visual Studio development offerings to potentially expose PHP developers to Visual Studio's rich development environment. Nevertheless, the tools portfolio is now better integrated with Expression Web 3 as Microsoft added Visual Studio Team Foundation Server support for source control of Expression Web files to make this project sharing even easier.

MICROSOFT AND OPEN SOURCE

Much has been said and written about how traditional commercial software players, epitomized by Microsoft, have been able to play in a world increasingly turning to

open source. While a deep discussion of the pros and cons of software available in open source and where the trend is going is beyond the scope of this paper, it is important to first outline the most often cited benefits of open source in application development software:

- ☒ Lowering of software acquisition costs by partially or totally removing the license component of software
- ☒ The generation of a strong ecosystem that can create a critical mass of resources and skills for a particular technology, thereby lowering the cost of developing, deploying, and maintaining it
- ☒ Improved interoperability through greater visibility of the source code, which, after all, is the most accurate form of specification of a piece of software
- ☒ Improved code quality and alignment of software function with its requirement through community participation
- ☒ A strategic shift in the control and governance equation of a given piece of software from its authors to its consumers (In this sense, open source provides a higher degree of democratization that can provide, for example, assurance that long-term evolution of software may not entirely depend on the decisions of a single vendor or group.)

It can be argued that even before open source became famous, Microsoft set itself the agenda to achieve some of the previously mentioned benefits through a mass-market approach that intrinsically realizes some of the lower-cost and democratization advantages when compared with the pre-PC market characterized by vertically integrated, all-inclusive hardware/software (e.g., mainframes) sold exclusively to businesses. In that sense, Microsoft's leverage of the PC revolution in the 1980s was premised on lowering the effective price of software by amortizing it over larger volumes of users (rather than offering parts of it for free) and creating a compelling ecosystem and consumer base that could help provide a passionate community of vested advocates similar in spirit to current open source communities. Additionally, the adoption of standards based on specifications was (and remains) the primary means of achieving interoperability, as imperfect as that might be. However, Microsoft and the traditional software industry maintained a strong grip on the strategic control and direction of software that is implied by the non-open source licenses attached to the software and generally resisted making much of the code itself visible.

A Shift Is in the Air

While intellectual property (IP) rights and patent licensing remain hot market spaces based on the volume of money flows that they generate, most commercial software makers have begun to warm up to, and in many cases proactively leverage, open source approaches to licensing their products. Microsoft, which was thought to be a vehement opponent of open source software for a long time, on account of its large collection of IP and the perceived threat of Linux or the LAMP stack to its established franchise, seems to have also warmed up over the past three years. Microsoft announcements of new open source projects or contributions to existing projects

have been accelerating recently and especially in the past 12 months. This is amply evidenced by the regular blogs on Microsoft's hip evangelism channels such as Port 25. It might be surmised that the intent of most of these initiatives is to simply win approval and goodwill from a community that has historically been antagonistic to Microsoft, but in almost every case, there are good business objectives and near-direct monetization of the open source effort. For example, the contributions to PHP discussed earlier in this paper are already resulting in more deployments of Windows as a Web server for PHP applications.

The following sections highlight some of the most interesting and impactful initiatives, projects, and contributions that showcase this shift.

Microsoft Open Source Technology Center

One of the most significant financial investments Microsoft has made in open source software is the Microsoft Open Source Technology Center. This center currently comprises two physical labs — one in Redmond, Washington, and one in Cambridge, Massachusetts — with over 400 servers running a wide array of Linux distributions. The Open Source Software Lab in Redmond, or OSSL, is tasked with ensuring that the most popular open source products run on Microsoft platforms. The lab is predicated on the notion that in order for Microsoft to build better products, it must pay special attention to the interoperability issues faced by its customers who are running open source products. The Microsoft Novell Interoperability Lab in Cambridge focuses more specifically on Novell's SUSE Linux distribution and other Novell products and their interoperability with Microsoft products. Work on running SUSE Linux on the Hyper-V hypervisor, as well as other projects such as running Windows Server 2008 on the open source Xen hypervisor, is done in the Cambridge lab. The projects worked on by the two labs and their deliverables are often chronicled on the official mouthpiece of the internal open source community at Microsoft, known as Port 25. Contributions from these labs are regularly integrated into various open source projects. Microsoft's investment in these labs is probably the best evidence of the company's new pragmatic attitude toward open source technology.

Port 25

Officially launched in March 2006, Port 25 is positioned as a channel of communication between Microsoft and its "open source" community. It has been a relatively successful communication channel set apart from Microsoft's main marketing sites (Microsoft.com) because it is understood that the OS community does not like to be marketed to. In reality, it is part of a healthy trend of more direct communication between Microsoft personalities (executives, architects, and key developers) and the active users of their products. This trend overall has resulted in documented changes in product design and, in some cases, strategies. Port 25 tries to interact with a community that otherwise does not interact with Microsoft. To that extent, its mere existence should be considered a positive step and a signal of the winds of change internally. The tone reflected in the Microsoft communication in Port 25 is that of a group that is truly passionate about its mission to increase Microsoft's participation in open source software, secure in the belief that such participation in and adoption of open source inside of Microsoft is a sound business decision. The evidence we have seen is that this group has had some high-profile successes in

driving collaboration between Microsoft and open source projects (e.g., collaboration with the Moonlight team for president Obama's inauguration). However, Microsoft has yet to jump into the open source software fray with one of its major revenue-generating projects.

CodePlex

Officially launched in June 2006, CodePlex is an open source project hosting Web site owned and provided to the community by Microsoft that now features over 8,000 projects. CodePlex allows the collaborative development of open source software and includes such features as source control based on Visual Studio Team Foundation Server and offers services such as bug and project tracking, project and release statistics and reporting, discussion forums, and wikis. CodePlex hosts many projects initiated by Microsoft and its key partners in which Microsoft employees are expected to provide key contributions. One of the most high-profile projects on CodePlex is the AJAX Control Toolkit, which provides a collection of components for use in ASP.NET AJAX Extensions applications.

Project Mono Collaboration

Mono is a project sponsored by Novell to implement the expansive set of .NET technologies (runtime, compilers, libraries, frameworks, etc.) on top of Linux and other Unix operating systems. Recently, Mono shipped version 2.2, which provides the core API of the .NET Framework as well as support for Visual Basic.NET and C# versions 2.0 with a partial implementation of .NET 3.x. Mono allows developers to work with .NET tools and leverage the highly productive .NET languages while deploying to Linux. While in theory a deployment on Linux does not result in licensing fees to Microsoft, the availability of Mono increases the likelihood that developed applications will also be deployed on Windows compared with a Java implementation of the application, for example. Thus the benefits to Microsoft are concrete in terms of building the .NET developer base and long-term prosperity for Windows. Mono is built based on the .NET ECMA specifications, and a collaboration between Microsoft and Mono, to produce an implementation of Silverlight in Mono called Moonlight (see the following section), was announced in late 2007.

Moonlight and Eclipse4SL

Moonlight is part of project Mono, but both projects relate to Microsoft's relatively new cross-browser runtime known as Silverlight, which brings Microsoft's rich desktop environments to the Web as a rich Internet application (RIA) browser plug-in. Web applications are expected to run in any browser and on any platform, and thus Microsoft has made the internal effort to support Windows and Macintosh but has relied on Novell to provide a Linux version called Moonlight. In February 2009 Moonlight 1.0 became generally available, delivering on the promise. To further enhance Silverlight's appeal, Microsoft is creating a tool that will allow Java developers to also leverage Silverlight by announcing a new Eclipse project to create an Eclipse plug-in (the most commonly used Java developer IDE) for Silverlight development. The new open source project, called Eclipse4sl, will help facilitate the integration of Silverlight-based applications into Java-based Web sites and services.

Windows Web App Gallery

Officially kicked off in the fall of 2008, this directory of open source applications provides a self-service contribution and installation mechanism for free software, which runs on Windows. Web App Gallery provides the ecosystem component of the Microsoft Web Platform, which is a set of tools comprising free or very inexpensive products that can be used to develop applications. While there is no requirement that the applications listed in the gallery be built with those exact tools, they do have to be available to run on the Microsoft Windows platform and must be available with a free license in some form. Most of these applications conform to a standard Web application architecture, though they may be built on top of any stack or combination thereof. Thus, some are ASP.NET based and others are PHP based, leveraging the PHP technology in Windows discussed in other parts of this paper. An internal validation process ensures that the established criteria are met for an application before it is listed. The concept builds on the current popularities of "App Stores" that are proliferating for many platforms (e.g., Apple iPhone), and while it appears to be receiving a regular stream of contributions, the number of applications is still small (just recently crossed the double-digit threshold). A directory from which applications can be discovered and installed is generally a good idea for any platform and is an overall improvement over what is typically available to commercial vendors, which after all prefer to collect some remuneration prior to giving up a software license. In fact, it is precisely the quality of this directory (click to install) that puts a fine point on the value proposition of open source software and why it is increasing in popularity vis-à-vis commercial software.

Microsoft as Open Source Contributor

As Microsoft's open source initiatives have matured, different groups have begun releasing a variety of product components of various sizes to meet targeted needs. In the past 18 months, Microsoft's open source engagement crossed a new threshold as Microsoft employees began contributing to other projects maintained outside of Microsoft and released under a variety of different open source licenses. The following is a list of representative projects that have been announced just in 2008 and 2009:

- ☒ Microsoft continues to contribute to the PHP project in improving PHP on Windows. A case in point is the recent (May 2009) announcement of a second release candidate for PHP 5.3 optimized for Windows.
- ☒ Microsoft made the source code for the Web Sandbox runtime available under the Apache 2.0 open source license.
- ☒ The Microsoft Interoperability Technical Strategy Team is, for the first time, participating as a code contributor to an Apache project: the Stonehenge incubator project.
- ☒ Microsoft's Powerset team also contributes to HBase, an open source, column-oriented, distributed database written in Java.

- ☒ Microsoft has contributed a patch to the ADOdb component of the PHP project, which is its first contribution to the PHP project. ADOdb is a popular data access layer for PHP licensed under LGPL and BSD.
- ☒ Microsoft and Red Hat recently signed agreements to test and validate their server operating systems running on one another's hypervisors.
- ☒ Microsoft recently announced that it is sponsoring research at the University of Michigan's Center for Information Technology Integration (CITI) to develop an open source Network File System (NFS) client for Windows (NFS is a very popular protocol used for sharing files among networked computers and/or storage devices in the Unix/Linux world).

GETTING ON MICROSOFT'S CLOUD

The notion of openness, while historically discretionary in traditional software, becomes intrinsic when architectures and delivery models begin to encompass cloud computing and cloud services. IDC defines cloud services as "consumer and business products, services, and solutions that are delivered and consumed in real time over the Internet." In that vein, Microsoft has been working hard to play in the cloud services space and is in the process of defining a new operating system intended for running applications in the cloud and a set of services it calls the Azure Services. Microsoft says that "Windows Azure is a cloud services operating system that serves as the development, service hosting, and service management environment for the Azure Services Platform." While a discussion of Windows Azure and the Azure Services Platform is beyond the scope of this paper, it is important to note that Microsoft's mere entry into the space presents the company with a higher bar for openness than it has ever faced. By their very nature useful, cloud services must offer APIs that must be interacted with externally and through standardized protocols from new and old applications written in any language. To this end, for example, Microsoft has already made a commitment that interfaces to the Microsoft .NET services, a key component of the Azure Web Services platform, will be available and supported for other languages by offering both Java and Ruby SDKs for .NET services. To the extent Microsoft is successful with its cloud ventures, it will undergo a gradual internal transformation toward openness that is likely deeper and broader than anything it has experienced so far.

PHP on Azure

Windows Azure promises the same FastCGI capabilities that have been outfitted to Windows Server 2008. If this is done, then it is destined to run PHP at a high level of performance even though PHP itself has to be downloaded separately. The vision of interoperability, which was highlighted in two recent Microsoft conferences (PDC 2008 in October 2008 and MIX09 in March 2009), has been surprising and inspiring in the range of technologies being blended. For example, an Eclipse IDE can be used to write a C# application that runs on Windows Azure or PHP applications that can access Windows Azure cloud storage or authenticate users with OpenID. This is a good sign of openness in that many PHP applications will have a chance to run in the cloud along with applications developed in .NET. The details are still being refined,

and Azure with its Live Services APIs, at least in their early incarnations, will likely be more suitable for newly developed applications requiring yet-to-be-defined migration processes for existing applications no matter the source language or architecture.

CHALLENGES AND OPPORTUNITIES

There is ample evidence that Microsoft is taking a new direction with open source technologies. Its efforts with PHP spearhead this new direction and represent a recognition of the realities of application development in an age when Web architectures have come to dominate and a significant chunk of development continues to take place with languages, frameworks, and tools not directly part of the Microsoft franchise. The new direction is commendable, but challenges and opportunities remain, such as:

- ☒ Microsoft should continue aggressive investment in FastCGI and SQL Server to ensure that essential features are supported to bring a larger portfolio of PHP applications to the Windows platform. Maturity and field testing of the FastCGI IIS capability and important additions to the SQL Server Driver for PHP v1.0, such as UTF-8 support, are still needed today.

- ☒ Microsoft's open source efforts, while signifying a change in attitude, may be progressing at too slow a pace. Microsoft still has to message a more candid position on how it reconciles its position as a bastion of proprietary intellectual property that it seeks to protect and its newfound interest in supporting open source players. While the dynamics of this reconciliation are not so mysterious and indeed the two can mix, Microsoft continues to be criticized and must engage more aggressively in a better articulation of its position. Additionally, Microsoft still has to figure out how to leverage an open source core model where a significant core of a revenue-generating product is offered through open source, with additional features, components, and services layered on top for revenue. This approach can be particularly appropriate in areas where more established leaders are competing heavily with Microsoft or in areas where Microsoft must accelerate traction in the market for a relatively new offering.

- ☒ In the area of cloud services, Microsoft has an opportunity to lay out a broader agenda that encompasses open source technologies and to do so with more concreteness and assertiveness. While cloud services are still in their relative infancy, their popularity has skyrocketed during the economic downturn due to their attractive pricing model. Cloud services are predicated on Web-oriented software architecture that allows consuming enterprises to retain minimal infrastructure on premises, and as such they pose a challenge to traditional software vendors, especially platform providers such as Microsoft. Microsoft has the opportunity to divest itself of its legacy reputation as a proprietary single-stack player and to play on the "choice" and "mixed" technology themes it has begun to cultivate.

CONCLUSION

With FastCGI, PHP has become a first-class workload on Windows Internet Information Server, and with the new SQL Server driver for PHP, many PHP applications can now viably deploy to SQL Server. PHP is one of the most important development technologies in the open source application development sphere, and its transition to this new status of support is a strong indication of the new direction Microsoft is taking with open source. While no billion-dollar Microsoft products have yet been turned into open source, a variety of smaller projects have begun distribution through the Microsoft open source license, signaling a shift in direction and attitude toward open source. Additionally, some Microsoft staffers make it their business to help other open source projects (e.g., Mono) achieve their intended level of interoperability with Microsoft technologies. Finally, Microsoft's new cloud initiatives, Windows Azure and the Azure Services Platform, put new emphasis on interoperability and have promised to run PHP and other open source languages on an equal footing with Microsoft's .NET technologies. IDC believes this bodes well for IT organizations that have to thread together a patchwork quilt of systems and applications to make a functioning enterprise. With the developments discussed in this document, there is significant evidence that Microsoft appears to have better aligned its priorities with the realities of IT organizations than at any point in its past.

CASE STUDIES

MindTouch

When the founders of MindTouch decided to launch MindTouch Deki, their enterprise collaboration and community software platform, along an "open source core" model, they looked around and quickly realized that the skill set they had on hand and the body of their application development expertise were based on Microsoft's development tools and languages. MindTouch Deki belongs to a new category of software, which enables users to connect and remix enterprise systems, social tools, and Web services in new and exciting ways that many organizations are finding compelling. MindTouch's founders philosophically understood the deep innovation and differentiation that the Visual Studio developer technology from Microsoft brought to market and its value and impact on the speed and productivity of the development process. "Visual Studio is probably Microsoft's most compelling technology and represents the evolution of the company's deep expertise from its roots in the MS-DOS Basic in the '70s," says Aaron Fulkerson, founder and CEO of MindTouch. MindTouch proceeded to develop its back-end application logic with C#. This created a dilemma because the majority of its customers and its specific go-to-market model were premised on open source deployment as a key adoption enabler. Running the business logic on Linux servers was a must for its marketing approach to work and so it resolved this dilemma by embedding its .NET application on the Mono runtime, which layers .NET on top of a Unix/Linux stack. While this layering is largely invisible to MindTouch customers, it meant that MindTouch had to work closely with the Mono open source project, which had begun to receive considerable support and help from Microsoft itself. On the Web tier, MindTouch used PHP partly because of the Web-tier skills at hand and partly because of its ubiquity, performance, and maturity on the Linux platform. Within three years of launch, MindTouch Deki was running at the sites

of over 1,000 licensed customers who opted to run the commercially licensed and formally supported version of the product, often after starting projects with the open source core made available for free downloads.

PHP on Windows by Demand

This enormous and rapid success quickly exposed MindTouch to a diversity of customers, including many who run Windows on their Web tier. This precipitated a set of experiments with PHP on Windows that brought MindTouch into intimate collaboration with the Microsoft IIS7 team, which had also set itself the goal of running a larger chunk of the PHP workload on Windows. MindTouch, like other early adopters of the technology, pushed Microsoft to meet commercial-grade performance and reliability requirements as the FastCGI interface was being refined by the IIS7 team. The result speaks for itself, with Windows now making up close to 20% of MindTouch's new customer deployment. "Overall, we are able to provide equivalent performance to our customers no matter what platform they choose for the PHP Web tier of Deki, allowing them to choose the platform that is most congruent with their IT infrastructure from a management perspective," says Fulkerson.

SugarCRM

Observing a vacuum around open source in the packaged application space and especially around the hot area of customer relationship management (CRM) software, the three founders of SugarCRM decided to fill it. They saw in open source an approach that could increase code quality through team collaboration and at the same time bring product to market at a rapid pace. They additionally understood the difficulties of gaining credibility and assuring customers of long-term credibility of a new entrant into an established space such as CRM and saw in the open source model an approach to providing such long-term assurance to customers. In a short five years since the company's inception in 2004, SugarCRM claims 4,000 paying customers in total, which is an outstanding number in a crowded market space. One of the secrets of the company's success is making bets on simple but powerful open source infrastructure such as PHP and Linux. The selection of the LAMP stack instead of more complex alternatives such as Java/J2EE not only provided speed-to-market benefits but also implied that more than any other CRM application of its type, SugarCRM relies on PHP for its business logic and database access. "We implement our business logic entirely in PHP, allowing us to leverage high-quality tools and talent in the market at lower costs than is possible with any other technology," says Majed Itani, software architect at SugarCRM.

Leveraging the Microsoft OS and Database

For a broad-based application provider such as SugarCRM, supporting multiple platforms becomes a matter of customer demand and service. The typical SugarCRM customer has a large footprint of Microsoft Windows and SQL Server in its IT infrastructure, and the typical CRM vendor SugarCRM competes with has more likely than not built its application on top of a Windows and SQL Server stack. Thus, from the beginning, SugarCRM found it essential to provide the Microsoft stack as a deployment option, forcing it to work with Microsoft on stabilizing PHP on earlier versions of IIS and supporting database access on SQL Server with efficiency. With

the new FastCGI support in IIS and the new PHP driver for SQL Server, SugarCRM has significantly improved the stability and performance of its offering on the Microsoft stack. According to Itani, "Not only can we wholeheartedly recommend the Microsoft stack today for our customers as an effective deployment option, but we have been pleased with the way Microsoft has extended a helping hand when we were field testing the beta days." Today, some 40% of SugarCRM's customers deploy on Windows, with about half of the deployments using SQL Server, including some international customers who utilize a SugarCRM workaround that allows SQL Server to handle UTF-8 data.

Appliedi

Having established itself as a small to midsize hoster with about 600 servers in custody, Appliedi is representative of many of the midtier hosters that cater to the small to midsize market. This space is dominated by Web site and Web application hosting with both ASP and PHP being the main technologies utilized. "Our story is probably indicative of hosters of our size everywhere in North America," says Jess Coburn, founder and CEO of Appliedi. It is also a tier of the hosting space where economies of scale are essential and managing to a high-level of service and to hard cost constraints is what separates success from failure. For this reason, Appliedi has decided to specialize in the Microsoft stack, running Windows Server for both ASP and PHP applications for the best utilization of available skills and resources. Appliedi was eagerly awaiting and testing the FastCGI improvements coming to IIS and was early to deploy the technology for its customers. "Our customers with PHP workloads don't much care what the underlying operating system is as long as we are able to provide them with the quality of service and performance they need. Our customers really like that they can run both PHP and ASP.NET on one platform and not require separate operating systems for these two workloads," says Coburn.

Copyright Notice

External Publication of IDC Information and Data — Any IDC information that is to be used in advertising, press releases, or promotional materials requires prior written approval from the appropriate IDC Vice President or Country Manager. A draft of the proposed document should accompany any such request. IDC reserves the right to deny approval of external usage for any reason.

Copyright 2009 IDC. Reproduction without written permission is completely forbidden.