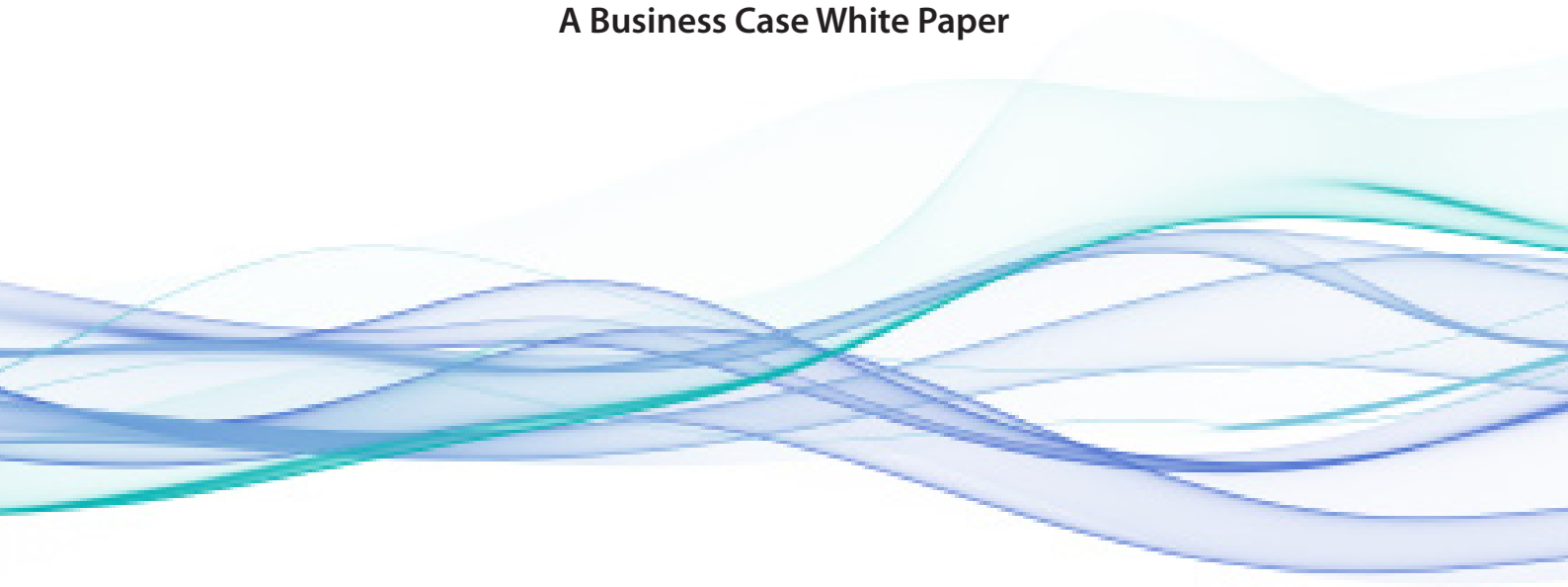
The PHP Company

Breaking the Deployment Bottleneck

Accelerating application roll out with Zend Server

A Business Case White Paper



Introduction

PHP has come a long way. Designed to solve issues around creation of dynamic server-side web applications, it has graduated into a de facto development standard, running almost half of the world's web sites. PHP has moved from the hobbyist garage and the halls of academia into large corporate applications with complex workflows and data structures.

By and large, the move into business critical applications has been successful. Many of the most recognizable names on the internet—including Yahoo, Wikipedia, and Facebook—are PHP-based, as are public sector sites including whitehouse.gov. Corporations across all business sectors, including General Electric, the New York Stock Exchange, and Bell Helicopter, have implemented PHP-based projects.

How has an open source platform become a linchpin for so many enterprises?

- PHP's maturity – language capabilities, frameworks, 3rd-party tooling -- has reached the same or better level as other enterprise-class technologies
- PHP enables rapid application development, which aligns well with the need for CIOs to keep pace with and often lead operational improvements and revenue increases through technology
- PHP's ubiquity in the programmer population offers a readily available pool of PHP-savvy talent that CIOs and IT managers can tap into as the need arises

Another advantage, and one that has paved the way for more reliance on PHP by large enterprises, is the Zend Server platform. Zend Server provides a tested and supported PHP runtime along with application server features that ensure high reliability and scalability, enhance security, and increase developer productivity.

Still, with all the pluses, there is room for improvement. Matching PHP's agility on the development side of the IT department can present challenges on the deployment side. As a result, application delivery can slow considerably as production staff seek and create workarounds to permit effective deployment.

This bottleneck has been broken. Zend Server now extends PHP agility to operations by addressing issues that cause deployment slow down.

Anatomy of a Deployment Bottleneck

What causes the deployment slow down? To answer this question, let's look at what must occur to get an application from development to production in a large enterprise. We'll do that with a fictional but not uncommon scenario.

A big brand national retailer needs to improve its e-commerce site by making it easy for customers to personalize their shopping experience. This is considered a business imperative as competitors have rolled out robust personalization and social media features that have led to increased sales. Ted, the director of IT, has agreed to a hard deadline of six months to go live with this new functionality.

Ted's development team launches the project using PHP because it allows rapid application development. It will also allow the team to roll out features incrementally on a weekly or daily basis, feature by feature. This agility has a number of advantages over the traditional approach of building a whole project that launches on a "grand opening" release date:

1. Customers will see immediate improvements in their online experience
2. Upper management will quickly see positive results from the e-commerce improvement
3. Small iterative deployments are easier to address if something goes wrong
4. All features of the new application will be live within the required time frame

The first features are ready for deployment according to the plan timeline. While the development team moves on to the next set of features, Ted transitions the tested feature set to Mark, a project lead on the operations side of the IT department, so that he and his team can prepare and implement the first round of changes and be ready for the next round.

Issues crop up almost immediately.

Mark and his team attempt to keep pace with the speed of the development team, but the lack of processes leads to errors. Deployment steps are run in the wrong order. The database schema doesn't get synched with the new version of the code. The project slows as the operations team goes back and corrects errors. In the meantime, the development team is ready to transition the next set of features for roll out.

The original goal of deploying features in incremental fashion, along with the benefits that this approach would create, begins to evaporate as the operations team gets bogged down. Ted and Mark seriously consider adapting a deployment technology originally designed for use with a different language. However, nobody on the team has the expertise to adapt any of those technologies to a PHP platform, and hiring the needed talent will take time and money. This would simply add to the delay, so Ted and Mark decide not to pursue the idea.

In order to avoid more process errors, the operations team is forced to slow down the pace of deployment further. This effort, designed to make a strategic difference to the corporation in a short space of time, has soaked up far more resources than intended without reaching a resolution. Ted dreads C-suite meetings because he will not only have to report continued delay, he will have to listen to the problems the delay is causing elsewhere in the business:

- Sales will not meet its targets, which were based in part on the ecommerce improvement going live on time
- Marketing must keep revising its campaigns, which had centered on getting maximum brand leverage from the "rolling improvements" Ted initially promised that aren't showing up on time
- Finance points angrily to both sides of the ledger, as Ted's department once again spends more than allocated without any revenues to justify the extra expense

On top of everything else, Ted considers the application's future scalability requirements, which leads to more concern. If the operations team is running into problems from manual errors and lack of processes as they deploy onto a single server, how bad will things be when they need to deploy *exactly the same way* onto an eight-server cluster?

Zend Server as Bottleneck Breaker

Let's rewind the scenario to the point where Ted transitions the first round of features to the operations team and make one early addition that changes everything.

While the development team moves on to the next set of features, Ted transitions the tested feature set to Mark, a project lead on the operations side of the IT department, so that he and his team can prepare and implement the first round of changes and be ready for the next round.

Mark has had experience deploying applications in previous jobs and is concerned about problems that may arise from the absence of a standardized deployment process. He knows that his team will not be able to match the development team's agility without a consistent and automated deployment process that cuts down on manual errors. He recommends implementation of Zend Server to provide the needed structure and Ted agrees.

As development team members finish each incremental build, they create deployment packages with Zend Server. Each package contains:

- The application code
- Prerequisites that make sure the system has all the correct PHP extensions and libraries to run the application
- All the parameters that the administrator will need to enter to install it correctly (e.g. database host and credentials)
- Installation PHP scripts that can be run at various points in the deployment process (e.g. after staging or activation)

The application prerequisites save time by ensuring compatibility, avoiding unexpected "wrong library version" crashes at some later point. Predefining installation parameters eliminates potential for deployment error. Zend Server deployment also enables integration with continuous integration servers (e.g. Hudson) for continuous building and deployment of code.

To maximize the application's scalability and flexibility, Ted's team decides to deploy it in the cloud, knowing that Zend Server can easily handle application deployment to multiple PHP servers. Using Zend Server Cluster Manager, they are able to auto-scale the application, spinning up new virtual hardware plus PHP middleware (Zend Server) as well as the actual application itself. This means that they are able to automatically adjust their application capacity based on load.

The result exceeds expectations. The project is completed within the required timeline, customer feedback is positive as features go live, and the business sees value in the form of improved bottom line performance and competitive advantage. Behind the scenes, Ted's team has a structured, easily maintained, business-critical application that will form the basis for a new approach to future development.

And there is more. An unexpected business benefit is the new kind of connection that the company makes with customers. Because of the short iterative deployment cycle, Ted's team is able to get continuous and ongoing feedback from users and stakeholders. They respond quickly to this feedback with seamless roll out of incremental improvements, which has a positive impact on consumer perceptions of the company and the quality of its customer service.

Conclusion

As enterprise applications take on increasingly business-critical tasks, IT departments must support more rapid development iteration. Risky, error-prone deployment processes that slow things down must be eliminated. Zend Server meets the demand for reliable deployment by allowing production staff to:

- Easily create deployment packages for applications that include source code, metadata and scripts
- Deploy applications on any number of servers using a simple wizard or through a Web API
- View deployment errors in the Zend Server or Zend Server Cluster Manager Events view
- Rapidly deploy or roll back application updates
- Simply auto-scale applications to add or remove capacity based on demand
- Receive full support for on-premise and cloud deployments

By taking the complexity out of PHP application deployment and post-deployment management, Zend Server breaks the deployment bottleneck and allows enterprises to accelerate the movement of applications from development to production.