Cloud computing service models, Part 2: **Platform as a Service**

Dan Orlando
CEO
Creative RIA

28 January 2011

Find out how Platform as a Service (PaaS) offers opportunities for developers to benefit from cloud computing, and how business owners and decision-makers can prequalify PaaS vendors, minimizing the potential for catastrophic mistakes.
In this three-part series, straightforward, real-world examples of cloud computing help eliminate the confusion around the concepts. Each article in this series covers one of the three service models: Infrastructure as a Service (IaaS), Platform as a Service, Software as a Service (SaaS).

View more content in this series

Platform as a Service is often the most confusing classification of cloud computing, because it can be difficult to identify, often being mistaken for either Infrastructure as a Service or Software as a Service. In this second of a three-part article series, learn what makes PaaS unique and how it can be leveraged in your business.

Frequently used acronyms

API	Application programming interface
HTTPS	Hypertext Transfer Protocol over SSL
IDE	Integrated development environment
ROI	Return on investment
SQL	Structured Query Language
SSL	Secure Sockets Layer
UI	User interface

The defining factor that makes PaaS unique is that it lets developers build and deploy web applications on a hosted infrastructure. In other words, PaaS allows you to leverage the seemingly infinite compute resources of a cloud infrastructure.

Of course, what appears to be an infinite amount of computing resources is an illusion, where the limitation is based on the size of the infrastructure. However, as you learned from the first article in this series, Google's infrastructure is estimated to contain more than 1 million x86-based computers. In addition, because the infrastructure for PaaS is elastic—a concept also discussed in Part 1—the cloud can be expanded to provide even more computing resources as needed, so the illusion of infinite resources isn't entirely imaginary.

PaaS for developers

Develop skills on this topic

This content is part of progressive knowledge paths for advancing your skills. See:

- Cloud computing: Fundamentals
- Cloud computing: Introduction to Platform as a Service

A common misunderstanding for developers is that cloud computing applies only to network administrators. But this misconception overlooks the many possibilities that cloud computing brings to development and quality assurance teams.

Consider some of the things that are often problematic during the software development life cycle. In my experience, the process of setting up the server environment that will host the Web application the development team has been assigned to build can be a huge hassle. Even in the largest enterprises, there is typically a single network administrator resource assigned to several developments teams. When PaaS is not being used, setting up a development or test environment typically requires the following tasks:

- Acquire and deploy the server.
- Install the operating system, run time environments, source control repository, and any other required middleware.
- Configure the operating system, run time environments, repository, and additional middleware.
- Move or copy existing code.
- Test and run the code to make sure everything works.

More times than not, the administrator resource is already stretched thin, so getting a new environment deployed can be a painful process. Another major problem for web application developers for both the client and server side is duplicating the run time environment locally for your own testing purposes.

Now imagine that you're part of a development team where you are using PaaS. In such a situation, you would have a virtual machine (VM) that contained the entire server environment and could literally be passed around on a USB flash drive.

I'd like to turn your attention to the cross-concept matrix shown in Part 1 to use as a reference as the series continues to analyze PaaS. That matrix is also provided in Table 1.

Table 1. Cross-concept matrix of the three classifications of cloud computing

	Paradigm shift	Characteristics	Key terms	Advantages	Disadvantages and risks	When not to use
IaaS	Infrastructure as an asset	Usually platform-independent; infrastructure costs	Grid computing, utility computing, compute instance,	Avoid capital expenditure on hardware and	Business efficiency and productivity largely depends	When capital budget is greater

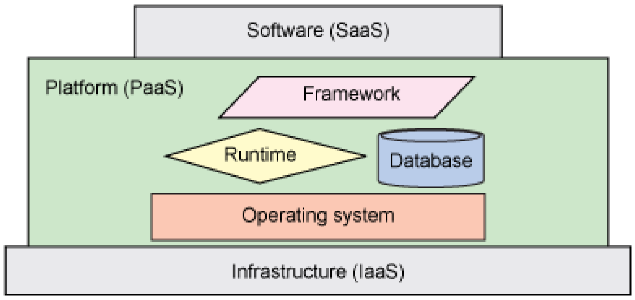
		are shared and thus reduced; service level agreements (SLAs); pay by usage; self-scaling	hypervisor, cloudbursting, multi-tenant computing, resource pooling	human resources; reduced ROI risk; low barriers to entry; streamlined and automated scaling	on the vendor's capabilities; potentially greater long-term cost; centralization requires new/ different security measures	than operating budget
PaaS	License purchasing	Consumes cloud infrastructure; caters to agile project management methods	Solution stack	Streamlined version deployment	Centralization requires new/ different security measures	N/A
SaaS	Software as an asset (business and consumer)	SLAs; UI powered by "thin client" applications; cloud components; communication via APIs; stateless; loosely coupled; modular; semantic interoperability	Thin client; client-server application	Avoid capital expenditure on software and development resources; reduced ROI risk; streamlined and iterative updates	Centralization of data requires new/ different security measures	N/A

The main ingredients of PaaS

Perhaps the best way to understand PaaS is to break it apart into its main components: platform and service. Now, consider the service being provided, which is referred to as a *solution stack.* That said, it is logical to assume that the two main ingredients of PaaS are the computing platform and the solution stack.

To illustrate these two "ingredients," let's look further into their definitions. A *computing platform,* in its simplest form, refers to a place where software can be launched consistently as long as the code meets the standards of that platform. Common examples of platforms include Windows™, Apple Mac OS X, and Linux® for operating systems; Google Android, Windows Mobile®, and Apple iOS for mobile computing; and Adobe® AIR™ or the Microsoft® .NET Framework for software frameworks. The important thing to remember is that you're not talking about the software itself but rather the platform on which it is built to run. Figure 1 provides an illustration to help you understand this relationship.

Figure 1. A graphical interpretation of the relationship between classifications of cloud computing and the elements of PaaS



Now that you understand the concept of platform computing, let's figure out what a *solution stack* is. A solution stack consists of the applications that will assist in the development process as well as the deployment of the application. These applications refer to the operating system, run time environments, source control repository, and any other required middleware.

Choosing a provider

The solution stack also sets the different PaaS companies apart, which is something you will need to explore in greater depth before making a decision to jump on board the PaaS train.

Here are a few of the basic questions you may want to ask prior to committing to a specific PaaS provider:

- What frameworks and languages does it support? Ideally, a PaaS should support any frameworks that are based on the language of choice for that platform.
- How many applications can I create? Most PaaS providers limit the number of applications you can build based on the plan or package you signed up for. Make sure the provider offers a plan or package that meets your needs.
- What type of content is allowed? The infrastructures that support PaaS offerings typically involve a concept known as *multi-tenant computing,* where many "tenants" share "residencies" on a single server, separated by VM instances managed by a *hypervisor.* A PaaS provider may have certain restrictions regarding the type of application and content you plan to host.
- What kind of databases are supported? This answer is very important if you have data that you will be moving over as part of your application. You must make sure that the database on offer from the provider is compatible with the format you intend to use for importing your data.
- Does it support SSL (HTTPS)? This is another important factor for security reasons. You're going to run into major problems if you plan on conducting transactions through your applications and you find out that SSL is not supported.

PaaS anatomy

Now that you've come this far in learning about PaaS, let's explore what features you should consider when comparing PaaS providers:

- Application development framework. A robust application development framework built on technology that is widely used. Ideally, you should beware of the potential for vendor lock-in here. Open source platforms such as Java™ technology are usually a safe bet in this regard.
- Ease of use. A PaaS should come with easy-to-use WYSIWYG tools that have pre-built widgets, canned UI components, drag-and-drop tools, and support for some standard IDEs. It should facilitate rapid, iterative application development.
- Business process modeling (BPM) tools. You need a strong BPM framework that allows you model your business process and build the application around it.
- Availability. The platform of choice should be accessible and available anywhere, anytime.
- Scalability. The platform should be smart enough to leverage the elastic capacity of an underlying infrastructure to handle the loads the application will be put under.
- Security. To effectively combat threats, the platform should address things like cross-site scripting, SQL injection, Denial of Service, and encryption of traffic and make it ingrained into

the application development. In addition, the platform must support single sign-on capabilities for you to be able to integrate it with your remaining on-premise applications or any other cloud applications.
- Inclusive. The platform should provide the ability to include, embed, and integrate other applications built on the same platform or others.
- Portability. The platform should be agnostic to the underlying infrastructure and allow companies to move the application from another IaaS to another.
- Porting tools. To facilitate an easy and quick migration of data from the legacy on-premise application to the application based on the new platform, bulk import transformation tools are a necessary part of the platform's toolkit.
- API. To perform tasks such as user authentication and storing and retrieving files (for example, Web application files and assets) and sometimes even making calls directly to a database, the platform should have a well-documented API. This will allow your business to have the flexibility of creating and customizing a software application to interface with the platform that meets the specific needs of the company.

Managing vendor lock-in

Vendor lock-in means that a customer is dependent on a vendor and unable to use another without being subjected to substantial switching costs. The opportunity to create an environment that supports vendor lock-in arises with technologies that are relatively new and growing in popularity, just like cloud computing. Early adopters must be aware of what they are getting themselves into before signing any long-term IaaS and PaaS agreements right now.

One way of avoiding lock-in is through standardization of APIs and platform technologies. Organizations such as Simple Cloud (see Resources) have started working with vendors of all sizes to participate in this open source project to bring consistency to PHP in the cloud. To create Simple Cloud, Zend Technologies, Microsoft, IBM, and Rackspace teamed up with a goal of providing an abstraction layer across different platforms.

The Simple Cloud API objective is to create common interfaces for file storage, document storage, and simple queue services. This would allow you to write applications that are portable across major cloud vendors. *Vendors who are taking initiatives like this to standardize cloud computing should be commended for doing so and encouraged to continue in these efforts.* When choosing a vendor to provide your company with PaaS services, I strongly recommend taking a good, hard look at providers who support standardization. Standards make life easier for all of us in the IT sector, and most importantly, they save businesses money.

To rid the PaaS marketplace of vendor lock-in opportunities, service providers who support the same underlying API are needed. The answer is simple: Service providers who are stuck on proprietary technologies must agree to support standardization initiatives such as Simple Cloud.

Conclusion

In this second of a three-part article series on the anatomy of cloud computing, you learned how to characterize and identify PaaS. You also learned the questions that should be asked when choosing a PaaS provider as well as the concerns you should have when choosing a provider, such as vendor lock-in. The last article in this series will dive into the anatomy of SaaS and show how to characterize and identify it. You'll also learn about the things that you need to be careful of when choosing an SaaS provider in order to protect your business. In the meantime, check out the Resources section for links to more information on PaaS.

Resources

- Grace Walker's developerWorks article "Cloud computing fundamentals" provides a good introduction to cloud computing.
- The Open Cloud Manifesto is a document that is "dedicated to the belief that the cloud should be open."
- Check out the article Cloud Computing: Waiting Too Long for Standards Will Cost You from CIO.com.
- Learn more about the Open Virtualization Format initiative.
- The Simple Cloud API initiative is the product of cooperation among Zend, Microsoft, IBM, Rackspace, and others.
- Explore developerWorks Cloud Computing zone, where you will find valuable community discussions and learn about new technical resources related to the cloud.
- In IBM Smart Business Cloud Computing, get valuable business advise to enhance performance and efficiency in the cloud.
- Follow developerWorks on Twitter.
- Watch developerWorks on-demand demos ranging from product installation and setup demos for beginners to advanced functionality for experienced developers.
- Get involved in the My developerWorks community. Connect with other developerWorks users while exploring the developer-driven blogs, forums, groups, and wikis.

About the author

Dan Orlando

Dan Orlando is a recognized leader in the enterprise development community. As a long-time consultant, Dan's expertise on Adobe technology platforms is often called upon by industry leaders as well as publications such as IBM developerWorks, Adobe Developer Connection, and Amazon Web Services. Dan can also be found blogging regularly at DanOrlando.com.