

Extending Intel's Enterprise Private Cloud with Platform as a Service

By increasing programmer productivity, PaaS will enable us to extend the value of our private cloud to more groups and usages, thereby supporting our technology roadmap for using hybrid (private-public) clouds to further increase scalability and cost efficiency.

Catherine Spence
Enterprise Architect, Intel IT

Travis Broughton
Enterprise Architect, Intel IT

Murthy Upadhyayula
Project Manager, Intel IT

David Sisson
PaaS Engineer, Intel IT

Executive Overview

Intel IT is actively implementing platform as a service (PaaS) as the next logical step for our enterprise private cloud, to accelerate custom application deployment and promote cloud-aware application design principles.

Our PaaS environment will build on our already successful infrastructure as a service (IaaS) efforts. We will use open-source software (OSS) to provide an environment featuring self-service, on-demand tools, resources, automation, and a hosted platform runtime container. We anticipate that PaaS will facilitate creation of cloud-aware applications through the use of templates, resource sharing, reusable Web services, and large-scale multi-tenancy.

Our traditional application deployment, or path to production, process has several less-than-ideal characteristics. For example, the process to deploy a new custom application consists of multiple manual steps and can take as much as 140 days. Also, developers must have a highly technical understanding of the underlying infrastructure, such as virtual machine provisioning and configuration, OS and middleware, and storage mechanisms. Agility is hampered by a lack of standard business processes, templates, and on-demand scaling capabilities.

We have conducted a successful proof of concept that tested PaaS in our enterprise private cloud environment and demonstrated its positive potential:

- Enhanced agility and productivity
- Greater standardization and extensibility
- Reduced complexity
- Improved utilization
- More efficient security and business continuity

We believe PaaS will empower developers to be in control from development to deployment—exponentially reducing time to production, optimizing the use of resources, and encouraging the development of cloud-aware applications. By increasing programmer productivity, PaaS will enable us to extend the value of our private cloud to more groups and usages, thereby supporting our technology roadmap for using hybrid (private-public) clouds to further increase scalability and cost efficiency.

Contents

- Executive Overview..... 1
- Background 2
 - Current Application Environment ... 2
- Solution..... 4
 - Projected PaaS Benefits..... 4
 - Implementing PaaS at Intel..... 6
- Proof of Concept..... 7
 - Key Learnings..... 7
- Next Steps 8
 - Near-Term Goals 8
 - Long-Term Goals 9
- Conclusion..... 9
- Related Reading..... 9
- Acronyms..... 10

IT@INTEL

The IT@Intel program connects IT professionals around the world with their peers inside our organization – sharing lessons learned, methods and strategies. Our goal is simple: Share Intel IT best practices that create business value and make IT a competitive advantage. Visit us today at www.intel.com/IT or contact your local Intel representative if you'd like to learn more.

BACKGROUND

In today's fast-paced business environment, agility is the key to success and the primary motivator for Intel IT's cloud-computing program. In support of our enterprise private cloud, Intel IT has made significant progress in implementing infrastructure as a service (IaaS), improving server provisioning from months to under an hour using a self-service model.

We view platform as a service (PaaS) as the next logical step for Intel's enterprise private cloud, enabling acceleration of custom application development and deployment and promoting cloud-aware application design principles. We have set an aggressive goal of enabling developers to transition from innovative idea to production in a single day.

Current Application Environment

We believe PaaS has significant potential to address the following weaknesses in our current application environment.

LIMITED AGILITY

Today, the process to build and host a custom application is lengthy and complex, often taking several months after an application is initially developed to fully deploy it into production. Each application follows its own path to production process, which includes source code development, test, and production phases. Each phase of the path to production requires a dedicated environment to be provisioned, compounding the complexity of application setup and deployment.

The typical application lifecycle includes 75 individual steps, only 9 percent of which are fully automated. The entire process can take 130 to 140 days for new custom applications, and 30 to 40 days for version updates. Other milestones in the application lifecycle, such as maintenance, new releases, and end-of-life, are also characterized by multiple steps and minimal automation, as illustrated in Figure 1. By the time the application is landed, it could be out of date or no longer relevant, resulting in lost revenue opportunities.

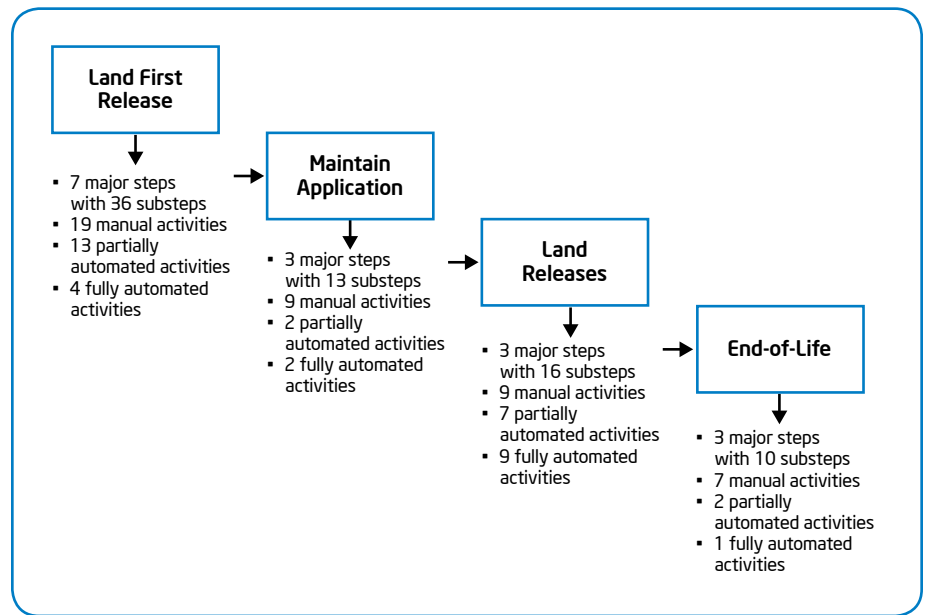


Figure 1. Currently, each milestone in the application lifecycle is marked by numerous steps and very little automation, creating an application environment that is slow and complex.

TOO MUCH COMPLEXITY

Currently, application development teams are responsible for provisioning their own infrastructure, with a great deal of IT assistance. Therefore, developers must have a deep technical understanding of the underlying infrastructure including compute, storage, network, manageability, and security resources. For example, they must know how much storage area network (SAN) and network-attached storage (NAS) to ask for. They must also request and deploy middleware that hosts the application, which requires them to know how the middleware interfaces with the infrastructure and make decisions about which underlying technologies they will employ. For them to configure their infrastructure stack, developers must know low-level details such as IP addresses and network storage device names and addresses.

This low-level knowledge comes with the additional cost of infrastructure lock-in, as developers are prone to hard-coding infrastructure details into the application's configuration settings, making it very difficult to migrate an application once it has been landed.

Governance, too is complex. Today, each new application must complete an extensive review process. The level of rigor is the same regardless of whether the application is a large, complex enterprise solution or a small, temporary web application. These reviews add time to the application deployment process, further reducing business agility.

POOR STANDARDIZATION AND EXTENSIBILITY

Development team agility is hampered by a lack of standard business processes, templates, and on-demand capabilities. Developers also do not have access to step-by-step instructions for moving an application from one phase to another, such as from development to test, or test to production.

Since the exact configuration of the application stack is left to the developer, there is a high degree of variation across stacks. For example, different developers

Understanding Platform as a Service (PaaS) at Intel

Platform as a service (PaaS) is a pre-provisioned environment (OS, abstracted middleware, and infrastructure) for building and rapidly hosting custom applications in the cloud. Developers code their applications using common programming languages and development frameworks, such as PHP, Java*, and .NET*, and deploy them into production without IT assistance using PaaS. PaaS provides several advantages for developers and lowers the barrier for creating cloud-aware applications:

- **Advantages for Developers.** PaaS facilitates application deployment through self-service, on-demand tools, resources, automation, and a hosted platform runtime container in the enterprise private cloud.
- **Cloud Attributes in Applications.** PaaS facilitates creation of cloud-aware applications through the use of templates, resource sharing, reusable Web services, and large-scale multi-tenancy.

With PaaS, application developers do not need an installation kit, don't have to order and configure servers or virtual machines, and don't have to copy files from one server to another. As Figure 2 shows, the application is pushed to the cloud from a command-line interface or directly from an interactive development environment (IDE) using a plug-in. The application is analyzed by PaaS and then hosted in the runtime container, which matches the application's resource requirements. The platform also provides elastic scaling, high availability, automatic configuration, load balancing, and management tools.

PaaS can be deployed on bare metal or static virtual machines (VMs), or within an IaaS environment. We have chosen to implement PaaS within a collection of VMs on IaaS due to the added flexibility and agility that IaaS affords. Within the IaaS environment, we can elastically scale capacity, not just for an application hosted on PaaS, but for the PaaS capability itself. We can also host a mix of custom and commercial applications together in the same infrastructure cloud. This should result in lower total cost of ownership and lower IT investment as PaaS grows in popularity within our developer community.

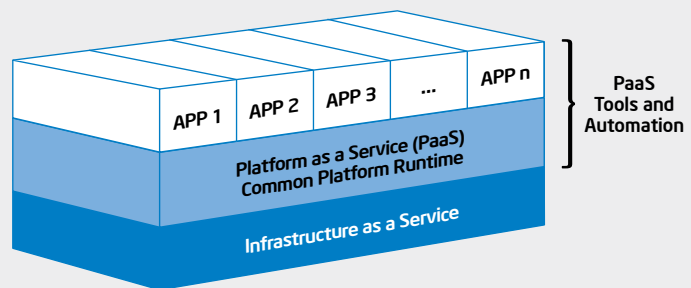


Figure 2. Intel IT's PaaS implementation capitalizes on our previous IaaS efforts.

might use different combinations of middleware. When applying a patch or performing business continuity and disaster recovery (DR) drills, this variation leads to an extensive validation process for all the individual aspects of a particular application stack. And, because each OS and middleware stack has the potential to be unique, we must back up each one to enable duplication and recoverability. Since we do not yet have a set of standards to provide, developers are left to their own preferences for development and landing of the application—making applications difficult to move after their initial landing due to hard-coded configurations.

Sometimes, Intel subcontracts work to companies to create applications on our behalf, for example, microsites, or targeted web applications, that support a product launch or other public relations event. These applications, which are created independently, often do not get the benefit of the capabilities or economies of scale of Intel's computing environment. Companies coding these microsites are typically given a functional specification but not any technical standards in terms of coding language, framework, or hosting provider to be used. Because both the language stacks and configurations are tightly coupled with the idiosyncrasies of their hosting environment, they are also difficult to move after initial landing.

Application monitoring is another area of concern. Due to the lack of a standard

monitoring framework, developers are responsible for application-specific monitoring—which makes centralized application support difficult. In addition, DR is not a built-in feature of the application development environment. Thus, development teams are responsible for planning, implementing, and testing DR for their applications. Providing redundancy at every level of isolated applications is not only expensive but also resource-intensive.

INEFFICIENT UTILIZATION OF RESOURCES

Application developers want to plan for scaling their applications, but have no way to automate this process. Therefore, they often significantly over-estimate their resource requirements. For example, a developer may request far more compute capacity and storage than the application needs, just in case it might be needed in the future. Over-estimating leads to poor resource utilization for the enterprise as a whole.

Because of the complexity involved in setting up an environment for an application, developers typically leave each of their path-to-production instances up and running for the life of the application, even if they are only required for a few days or weeks per release. This results in a large number of virtual or physical machines that are underutilized most of the time.

SOLUTION

We believe that PaaS will improve our support of Intel's developers and streamline our deployment cycle by providing a quick, efficient way to deploy applications into production, scale them, and take them out of production. PaaS will increase developer productivity and optimize the use of resources, thereby supporting our technology roadmap for using hybrid clouds for further efficiency.

PaaS supports Intel's "inside out" cloud strategy by encouraging the development of more cloud-aware applications. These applications are, by their nature, on-demand, self-service, scalable, elastic, multi-tenant, and metered. In addition, PaaS will specifically address the inefficiencies that currently exist within our application environment, as shown in Figure 3.

Projected PaaS Benefits

We believe PaaS will have several advantages over our current application environment.

ENHANCED AGILITY

Through on-demand self-service development and hosting, PaaS will allow Intel businesses to evolve more quickly by reducing the time it takes to launch an application. In addition, the PaaS environment will enhance application developer productivity by automating tasks and simplifying business processes. When

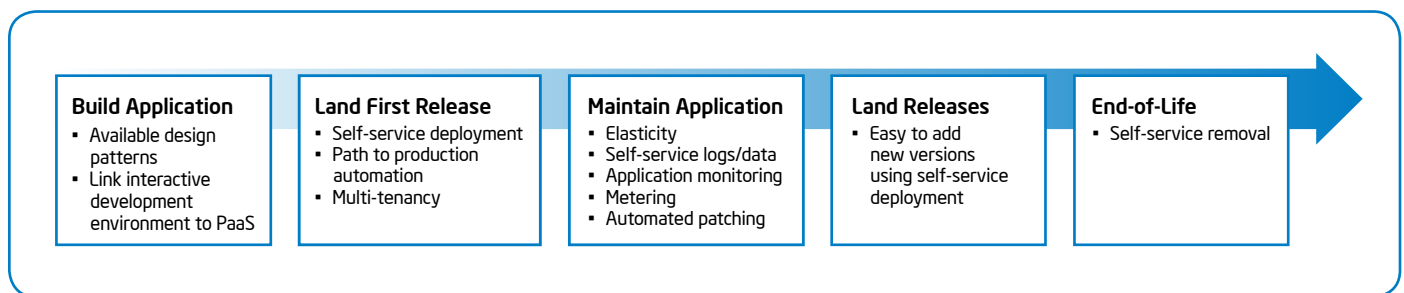


Figure 3. PaaS will provide a quick, efficient way to deploy applications into production, scale them, and take them out of production.

written to a PaaS environment, all layers of an application will receive rapid elasticity, enabling the application to quickly respond to changing demand levels. Furthermore, by building our PaaS stack on top of IaaS, we gain the agility benefits that IaaS provides. The PaaS service itself can be elastic, rapidly changing its total capacity to meet the aggregate needs of its tenants.

REDUCED COMPLEXITY

PaaS abstracts infrastructure, middleware, and configuration details, allowing developers to concentrate on code development. The environment is provisioned on demand, and elastic so developers can simply use it, not build it.

PaaS will enable us to simplify the governance of new applications by pre-approving the PaaS environment itself and by streamlining the governance process for certain application design patterns. By reviewing and approving the PaaS, the underlying environment is already certified for the applications that land on it, eliminating the need to duplicate the review process for each application. For example, a generic two-tier architecture can be deployed without necessarily having to be reviewed for host-to-host communication and interdependencies.

GREATER STANDARDIZATION AND EXTENSIBILITY

With PaaS, we can quickly recreate applications in the cloud. A new instance of an application can be launched for additional capacity or for DR reasons. PaaS provides standardized application monitoring for developers to measure availability, health, and performance. Instead of standard troubleshooting, we can easily stop and restart or redeploy the application, only spending time to troubleshoot when patterns emerge. In addition, with PaaS, we can provide standard, built-in DR capabilities. Standardization and automation of platform components and business processes promotes multi-tenancy, re-use, and common platforms.

Also, by employing a standard hosting platform that supports multiple standardized application frameworks, we will improve the portability and reusability of our code, including software developed by third parties. With PaaS, we can specify a set of frameworks and PaaS environments in the public cloud to validate against, which will ensure that microsites developed by external contractors can be redeployed at Intel or one of our preferred hosting providers. While we may not give contract developers the ability to directly commit code to our production environment, we can quickly transition from

their path to production to ours without having to rewrite any code.

IMPROVED RESOURCE UTILIZATION

PaaS will improve the efficient utilization of resources. In our current environment, we provision a full virtual machine (VM) for each application. With PaaS, we will create multi-tenant VMs, which will reduce the memory, disk, and CPU cycles required per application. Also, by allowing developers to scale applications using a self-service, on-demand process, PaaS will eliminate the necessity of over-estimating resource requirements.

Because developers will be able to instantaneously provision new application instances through self-service, we expect that path to production environments will be created when needed and destroyed upon completion of that development phase rather than being maintained indefinitely. In this way, PaaS will build on the increased efficient asset utilization already gained by virtualization and resource consolidation provided by IaaS.

The benefits of PaaS have the potential to improve the timeliness and efficiency of the application development process, from provisioning and installing new code to ongoing management. Table 1 provides some examples of these activities and how PaaS will improve them.

Table 1. Projected Benefits from Paas for Sample Path to Production Activities

| Development Team Task | PaaS Improvements | Result |
|---|--|---------------------------------|
| Provision server environment and services | <ul style="list-style-type: none"> Virtual machines pre-provisioned with platform Abstracted infrastructure Single-click deployment | Days to hours |
| Load and install new code | <ul style="list-style-type: none"> Single-click deployment | Hours to minutes |
| Respond to problems | <ul style="list-style-type: none"> High availability of PaaS Application design for failure Restart on fatal error Fast rebuild and application redeployment | Hours to minutes |
| Initiate hosting | <ul style="list-style-type: none"> Developer-controlled deployment Abstracted infrastructure management | Developer self-service |
| Manage application capacity | <ul style="list-style-type: none"> Resource consumption data Tools to add/remove additional application instances and components, such as a front-end web server | Elasticity |
| Improve asset utilization | <ul style="list-style-type: none"> Consolidated platform across applications Fewer idle preproduction environments Metering | IaaS + PaaS resource efficiency |

Principles of Cloud-aware Application Design

The use of PaaS accelerates cloud adoption. For traditional application developers, PaaS introduces cloud concepts and promotes a new approach to software design that eliminates application silos in favor of simplified, fault-tolerant, modular services that run in a virtualized, elastic, multi-tenant environment.

The principles of cloud-aware application design, shown in Table 2, can help Intel developers make the shift to PaaS more easily.

Table 2. Cloud-Aware Application Design Principles

| Principle | Details |
|--|---|
| Virtualization and elasticity | <ul style="list-style-type: none"> Auto-scale application based on demand to grow and shrink Compartmentalize components so that they can run across multiple virtual machines (VMs) Plan for dynamic IP addresses |
| Web services | <ul style="list-style-type: none"> Consume standard services Design application as a component for mashups, using a secure web services layer, such as representational state transfer (REST) |
| Design for failure and high availability | <ul style="list-style-type: none"> Use a load-balancing strategy with VMs in multiple availability zones Avoid maintaining application states in memory; instead, write to a shared database so user context persists across application hosts and can survive individual host failures Use process threads that resume on reboot Resynchronize the state by reloading messages from queues |
| Multi-tenancy | <ul style="list-style-type: none"> Provision new tenants on-demand using scripts and configuration Use security rating to select the multi-tenancy model |
| Avoid lock-in | <ul style="list-style-type: none"> Assume application will land on multiple clouds Use progressive features when available, such as detecting which platform is being used, to enable more advanced capabilities |
| Design for performance | <ul style="list-style-type: none"> Distribute applications across multiple geographies for fault-tolerance and improved performance Use application performance management tools to monitor and improve code base |

Implementing PaaS at Intel

To implement PaaS, Intel IT had to consider many factors.

APPLICATION CONSIDERATIONS

We envision using PaaS for small- to medium-size custom applications that have a cloud-hosted component. We anticipate that most applications using PaaS will be new, although some may be ported applications. Mobile applications, hybrid applications (those hosted on multiple clouds), and service-oriented applications are specific types of applications being targeted for PaaS. We see these types of applications as important to the enterprise, so building an architecture that enables them to work well is important.

We have also investigated deploying certain open-source applications, such as content management systems, in a PaaS environment. We are evaluating the tradeoffs of hosting them in PaaS compared to a dedicated IaaS configuration.

Some examples of Intel applications well-suited for PaaS include Intel.com public applications, Intel applications provided as a service (SaaS) to Intel customers, temporary applications for marketing campaigns, and specialized departmental and business-to-business applications.

DEVELOPER CONSIDERATIONS

We plan to provide application templates and sample code to accelerate the development of the targeted applications and encourage cloud-aware behavior by providing a starting point for developers to begin coding. While we can build some cloud capability into the platform itself like high availability and load balancing, some fundamental changes are required to traditional application design. (See the sidebar, "Principles of Cloud-aware Application Design.") PaaS can be a way to reduce the learning curve for those developers making the transition to cloud.

PAAS ENVIRONMENT CONSIDERATIONS

While we are structuring our PaaS solution around a single common platform, tooling, and business process, we are not limited to deploying a single instance of PaaS. In fact, we envision that we will likely instantiate multiple copies in the same or multiple clouds, depending on the needs of the business and the required level of isolation for a set of applications. PaaS provides us with the ability to instantiate new environments, on-demand, as needed.

For example, we can deploy two separate PaaS environments to meet the unique needs of our internal- and external-facing applications. While we will keep these applications segregated to foster security, we will use consistent tools and practices to provide developers with a sense of familiarity when developing for both. We have already deployed multiple private clouds using IaaS for internal, external, and de-militarized zone use, and we will deploy the PaaS hosting environments as tenants of those clouds. PaaS will also foster new opportunities for internal-facing applications, from rapid development of enterprise services to developing personalized mashups, where applications combine local resources and data with a variety of Web services.

PROOF OF CONCEPT

Over the past six months, Intel IT has researched PaaS and conducted a three-month proof of concept (PoC) to determine the requirements, scope, and implementation strategy associated with offering PaaS on Intel's enterprise private cloud.

Our research led us to standardize on open-source software (OSS) programming language stacks. This approach provides a single broad, flexible solution that supports the programming languages and frameworks in use at Intel. An OSS-based implementation enables us to reap the benefits of working with the OSS community (such as rapid updates, low cost, and ability to collaborate), while still being able to customize our PaaS implementation.

The objectives of the PoC were as follows:

- Evaluate the virtualization, elasticity, security, monitoring, and manageability capabilities of traditional and open-source PaaS solutions
- Provide a recommendation for a support model for the traditional and OSS solutions and document any known gaps
- Model the process to deploy the representative application through the full path to production in the tested PaaS stacks

- Communicate any recommendations we had to improve the possible production release of the tested PaaS stacks to the open-source community and to specific suppliers

The PoC was successful, enabling us to further understand PaaS, how developers will use it, and how it can benefit Intel's enterprise private cloud. Table 3 summarizes the accomplishments during the PoC.

Key Learnings

After completing the PoC activities, we held focus group discussions with the participating developers. It is difficult to quantify the different developer experiences, and just seven developers cannot be construed to completely represent Intel's dispersed community of application developers. However, our dialogue with them provided valuable insights.

We discovered that developers are still learning what the cloud means for application design, and we can help them shift to the cloud by providing an easy-to-use PaaS environment that meets their needs. We also learned that developers highly value on-demand self-service—they desire independence. New hosting solutions must support a continuous, agile application development process where teams can rebuild and redeploy their applications instantaneously, multiple times over the course of a day, as opposed to a regularly scheduled release cycle.

Table 3. Activities and Results During the Paas Proof of Concept

| Activity | Result |
|----------------------------|--|
| Installation | <ul style="list-style-type: none"> ▪ Successfully installed two PaaS stacks on Intel's enterprise private cloud, in addition to providing PaaS in a stand-alone developer version |
| Path to production process | <ul style="list-style-type: none"> ▪ Exceeded our target of deploying two applications in each environment ▪ Uploaded multiple applications coded in a variety of programming languages using PaaS ▪ Completed feedback collection sessions with seven developers |
| Production assessment | <ul style="list-style-type: none"> ▪ Conducted high-level security and manageability appraisals ▪ Developed PaaS strategic architecture ▪ Analyzed cost considerations including budgetary product pricing and environment sizing ▪ Established strong relationships with suppliers and provided product-specific feedback to them |

AGILITY, SCALABILITY, AND ELASTICITY

Developers indicated that they like being able to scale their application instances up and down through the user interface, especially to accommodate a sudden demand increase. They expect the platform to scale without having to make any application or platform configuration changes. Providing a user interface that allows developers to control their instances is a major improvement over what they have today, where they must deploy a new application instance from scratch.

Developers did not rate automated elasticity as high as self-service. They claim that they are able to predict demand without elasticity. The reality, though, is that currently they significantly over-estimate their application resource needs. Because developers are not charged back for their consumption of resources, they spend little time right-sizing their utilization. However, if developers can see the cost and consumption, we think that this will have an impact on their behavior. In addition, automated elasticity will simplify the ongoing process of making adjustments. By automating the growing and shrinking of the environment based on demand, we can achieve more efficient utilization.

COMPLEXITY AND FLEXIBILITY

We have chosen to work with a single platform based on open-source solutions. This approach supports a variety of programming languages and frameworks and is flexible enough to address different development needs.

We also learned that, although we need to simplify the application development process, we need to be careful to avoid over-simplifying it so much that everything becomes an exception that must be handled individually. Therefore, although we intend to automate as much as possible, certain things will still need to be done manually.

Also, tradeoffs between solution complexity and flexibility are unavoidable. For example, we must decide how much middleware to

include in the platform, and weigh multi-tenancy efficiencies against security considerations in areas such as database and storage consolidation.

STANDARDIZATION AND MULTI-TENANCY

Our goal is to support standardization and encourage multi-tenancy as much as possible. We can enhance resource efficiency and scalability if applications share standard application stacks and use common resources. Platform-level multi-tenancy goes beyond the virtualization of applications running on individual VMs or set of physical servers. PaaS enables the virtualization of applications across VMs, each of which supports a particular stack of resources. When applications are pushed to the cloud, they are analyzed by PaaS to determine which resources are required and are then hosted on the VM that contains those resources, along with other similar applications. With PaaS, even specific portions of an application can be hosted on a VM that is built to run a particular application tier. For example, sets of VMs can be configured as web servers, application servers, or database servers. This creates an environment that supports a high degree of multi-tenancy, as similar tiers of multiple applications are consolidated.

NEXT STEPS

Having completed the successful PoC, we have created a roadmap for adding capabilities to our PaaS implementation that includes near-term and longer-term goals. These capabilities will support our overall goals of fast path to production, design for failure, efficient utilization, and component reuse.

Near-Term Goals

We are in the process of offering an early adopter pilot, which we hope will lead quickly to full production deployment. We will work to

create awareness of PaaS, as well as develop an understanding of the existing demand for PaaS. Communication techniques that we are using include giving demonstrations to application developers, publishing a PaaS decision framework, and conducting developer surveys. As developers begin to adopt PaaS and become more proficient at developing cloud-aware applications, we will adjust our PaaS implementation to respond to changing developer requirements.

As part of our PaaS deployment, we are developing best practices for using PaaS to develop cloud-aware applications. By becoming cloud-aware, applications are able to fully leverage the underlying infrastructure for improved scalability, performance, and resiliency. Cloud-aware applications also help support business continuity by enabling us to host applications in redundant data centers or cloud providers, to avoid a disaster zone or point of failure. We will provide developers with the necessary tools to balance loads across availability zones or cloud providers. Applications that conform to these standards will be able to subscribe to services similar to those used to provide automated elasticity, which will trigger corrective actions should the application or infrastructure fail.

We will also finalize our baseline for the application development process, then develop a mechanism for measuring PaaS improvements by establishing metrics and indicators associated with aspects of application development, such as provisioning, new code installation, mean time to restore service (MTRS), resource utilization, and user satisfaction. We also need to deliver cost analysis, comparing PaaS with traditional software hosting models.

In terms of the technical solution, we plan to further explore the benefits and drawbacks of our open-source approach and deliver cost analyses. We have initially found that open-source solutions enable Intel to benefit from the fast pace of community updates, while enabling us to differentiate the solution for Intel's business. However, it will take us time

to develop best practices for embracing the fast rate of change and taking advantage of the opportunities to differentiate.

In the near future, we anticipate automating path to production, hosting, provisioning, and patching; finalizing our Web service strategy; and providing design patterns for mobile and hybrid applications. Later, we'll add auto-testing frameworks, quality-of-service, and end-to-end analytics for both the platform and applications, mobile application platform automation, showback and billing models, and advanced troubleshooting capabilities.

Long-Term Goals

In the future, we will expand the functional capabilities of PaaS. We will integrate the platform more closely with web services to make it easy for applications to consume existing services as well as publish new services that can be consumed by other applications. We will explore ways to automate the governance process based on use cases derived from application characteristics. Plus, we would like to offer development coding environments that are available on-demand and integrated with the cloud.

As part of our elasticity and DR enablement, we will experiment with bursting applications to the public cloud. We will look at both full migration of the application and partial migration in a hybrid cloud model. We will have several options for doing this since we could either create a PaaS environment in a public cloud IaaS hosting service or use a public cloud PaaS provider that uses the same underlying PaaS stack that we are adopting.

CONCLUSION

Agility is critical for succeeding in today's fast-paced business environment. By offering PaaS in Intel's enterprise private cloud, we can accelerate time to market for new custom applications, as well as promote the development of more cloud-aware applications and support Intel's vision of using a hybrid cloud model to meet spikes in business demand.

Based on our research and successful PoC, we believe that PaaS will address many of the challenges associated with our current application development process:

- Streamline the path to production, removing IT processes from the critical path
- Abstract infrastructure details, so developers can focus on code development
- Increase standardization of application stacks and business processes
- Improve resource utilization by providing self-service, on-demand scalability
- Enhance security and business continuity

Our implementation of PaaS will use OSS solutions to provide a pre-provisioned application environment that is on-demand, self-service, scalable, elastic, multi-tenant, and metered. Building on our already successful IaaS efforts, PaaS is the next logical step for Intel's enterprise private cloud.

RELATED READING

- "Infrastructure Data Management for Intel's Private Cloud," Intel Corp., June 2012
- "Building an Enterprise Private Cloud," Intel Corp., December 2011
- "Architecting Software as a Service for the Enterprise," Intel Corp., October 2009
- "Better Together: Rich Client PCs and Cloud Computing," Intel Corp., March 2009
- "Developing an Enterprise Cloud Computing Strategy," Intel Corp., January 2009

For more information on Intel IT best practices, visit www.intel.com/it.


ACRONYMS

| | |
|-------------|-------------------------------------|
| ACL | access control list |
| DR | disaster recovery |
| IaaS | infrastructure as a service |
| IDE | interactive development environment |
| MTRS | mean time to restore service |
| NAS | network-attached storage |
| OSS | open-source software |
| PaaS | platform as a service |
| PoC | proof of concept |
| SaaS | software as a service |
| SAN | storage area network |
| VM | virtual machine |

This paper is for informational purposes only. THIS DOCUMENT IS PROVIDED "AS IS" WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NONINFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION OR SAMPLE. Intel disclaims all liability, including liability for infringement of any proprietary rights, relating to use of information in this specification. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted herein.

Intel and the Intel logo are trademarks of Intel Corporation in the U.S. and other countries.

* Other names and brands may be claimed as the property of others.

Copyright © 2012 Intel Corporation. All rights reserved. Printed in USA  Please Recycle

0612/ACHA/KC/PDF

327465-001US

