Application Containers

Transcending the private-public cloud frontier

High performance. Delivered.

Digital Product Ecosystem platform provides manufacturers new channels for customer engagement and monetization of product ideas

accenture

Strategy | Consulting | Digital | Technology | Operations

Today's digital enterprises recognize the myriad opportunities available from the modern data supply chain. To unlock the full value of big data, they are applying data acceleration principles to move data swiftly from its source, process it to gain actionable insights quickly, and enable faster data interactivity among users and applications.[1] This requires not only exploring new architectural components, but also deciding on the proper balance of infrastructure to conduct the processing steps. Making the correct architecture and infrastructure decisions now will position companies to become tomorrow's Intelligent Enterprises, as discussed in the Accenture Technology Vision 2015.[2] These pervasively data-driven companies of the future will rely on software intelligence—including automation, machine-learning and cognitive computing—to optimize operations on all fronts.

In terms of current infrastructure decisions, there has been an overwhelming popularity in the last few years of on-premise private clouds hosted within the enterprise's own datacenters, giving it complete control and configurability of the infrastructure and security. But to truly maximize value and provide for growing capacity, companies should not only make the most of their existing on-premise infrastructure, but also enable on-demand scalability by expanding to the public cloud. This hybrid cloud architecture is inevitable if the enterprise has to balance the growing demand for resources against the need for secure management of data. "For most large organizations, the coexistence of cloud and traditional on-premises application infrastructures will be the norm in the context of hybrid integration platforms."[3]

Recognizing this pattern in cloud extensibility, this paper provides suggestions on making the leap from on-premise private cloud to hybrid architecture through the use of application containers. In the next sections, we describe the key principles and benefits of on-premise private clouds, provide recommendations and use cases for using application containers to move toward a hybrid cloud model, elucidate the value proposition for the business, and finally describe the microservice application architecture used with containers. Taking these infrastructure steps now will make it possible for companies to take data acceleration to the next level in the future by automating application scaling.

Key Principles and Benefits of On-Premise Cloud

This section elaborates some of the benefits of on-premise private cloud technology. Some of these overlap with other cloud environments as well. The section is aimed to set the context for the rest of the paper.

In-House Management

The primary feature and benefit of on-premise private cloud is control. In house management gives the enterprise complete control over the procurement, processing, storage and usage of their data.

Compliance

Companies have security regulations and compliance requirements around the storage and use of data. These could be in the form of mandatory overnight security updates and patches, data encryption requirements or access control policies.

Resilient to internal outages

With on-premise private clouds, the enterprise has a dedicated IT team and custom disaster recovery plans to recover quickly from internal outages providing it with options outside the scope of public cloud providers. They are self-reliant and do not have to depend on external help for failure recovery.

Customization

Since the infrastructure and services are built specifically for the enterprise, the cloud providers can establish custom configurations and offerings specific to the company's use cases and requirements.

Same experience as public cloud

While the administrators of the private cloud can customize offerings and publish to a catalog, the end users of the private cloud share the same experience as public cloud users. "The experience includes selecting the best-fit service from a catalog that is common to all qualified subscribers."[4]

Multitenancy

Multitenancy is the capability that allows different user groups, such as functions within an enterprise, to create clusters and run workloads on these clusters with no access to the sensitive data of other groups (i.e., users of a tenant share the same view of the tenant but have no access or visibility into other tenants).

Modular scalability

As the business needs evolve, resources can be scaled dynamically on-demand without disrupting ongoing operations. Modular scalability has to do with scaling specific business logic services without having to redeploy the entire application.

Containers—Moving to the Hybrid Architecture

Although an on-premise private cloud has the advantage of added control, security and resilience, it is limited when the company needs to scale beyond the available resources. The enterprise pays an upfront cost to procure the necessary servers and other hardware. If scaling up beyond the existing resources is not a pragmatic way to meet the growing demands of users, eventually the enterprise needs to scale out to the cloud in a hybrid-style architecture. The hybrid environment allows infrastructure users to distribute their workloads between the on-premise private cloud and the public cloud.

Application containers are rapidly becoming a viable option for seamless transition from on-premise to hybrid architecture models.

Containers are lightweight application-hosting environments that share the kernel of the host operating system (OS) and allow multiple isolated user spaces, precluding the need for a guest OS. This makes it lightweight compared to a full virtual machine (VM) and more performant when running multiple instances of the same application. For example, consider a company that has an OS image of size 1GB. For a full VM, it will require 1GB times the number of VMs. For 100 VMs, that is 100 X 1GB at least. Whereas with a container framework, the host OS is shared and so all 100 containers together take up a little more than 1GB of space (Figure 1).

One of the biggest benefits of containers is portability. They provide a way of packaging applications, the configuration values and dependencies such that they can be run anywhere, eliminating the need for vendor lock-in. They can be deployed on bare metal OS or a virtualized platform on-premise, but can easily be ported to a public cloud deployment. In contrast, VMs are heavyweight containers sporting a full guest OS and are tightly coupled to their compatible hypervisors, adding their own complexities when making the shift to a hybrid environment.

However, containers by themselves cannot address the challenges of managing a cluster of compute nodes that require multiple application environments (development, staging and production), high availability, quality assurance and performance monitoring. To handle these additional bookkeeping and management tasks, containers work in tandem with other tools or frameworks. The next section describes two implementation techniques using containers and benefits of each.

Figure 1: Architecture comparison of virtualization vs. containers

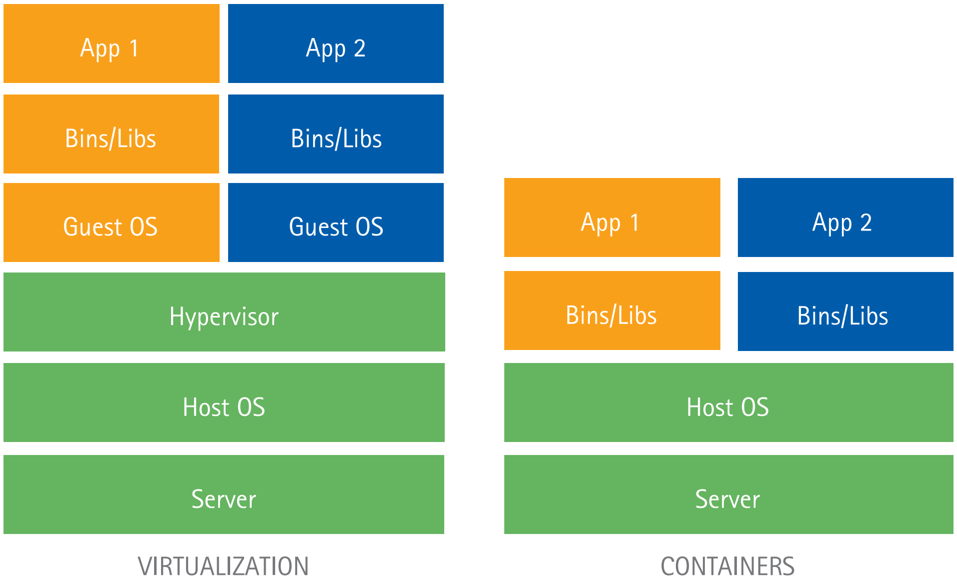
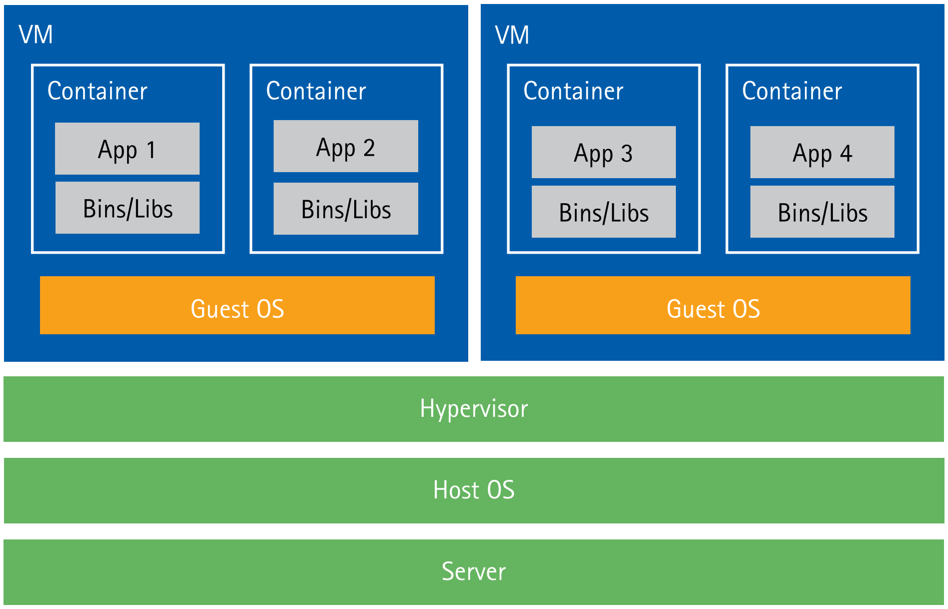


Figure 2: Container on PaaS



Container on PaaS

Technically, platform as a service (PaaS) providers are the earliest adopters of containers. They have an abstraction layer that encapsulates the PaaS from the underlying infrastructure as a service (IaaS). They provide a strong isolation between applications, which can run on the same physical or virtual instance. They also provide rapid resource management in terms of creating, scaling and destroying applications.

In fact, according to Gartner, "in most cases OS containers will not replace hypervisor-based virtualization technologies. Even when OS containers are used, hypervisor-based virtualization will often be used to enforce additional isolation for security purposes, and may be used to provide additional resiliency; in other words, OS containers will often be deployed within."[5] In a container on PaaS implementation, enterprises can create multiple VMs (limited by the IaaS quota) and deploy multiple containers in each VM, of course limited by the size of the VM (Figure 2). PaaS orchestration monitors the containers and restarts them on failure. Additionally, it allows the setting of dependencies between containers running in the same virtual machine. This way, if a container fails and is restarted, it will restart all of the dependent containers. Furthermore, the PaaS can also monitor the health of the VMs themselves, restarting them in the event of a failure.

Container on bare metal with orchestration

This setup involves running containers natively on bare metal OS and providing the cluster management, scaling, load balancing, scheduling and performance monitoring via orchestration engines. Orchestration engines help to horizontally scale multicontainer deployments across a container host and interconnect multiple layers of the application stacks. This enables companies to orchestrate services running in multiple containers into unified, large-scale business applications.

Figure 3: Container on bare metal with orchestration

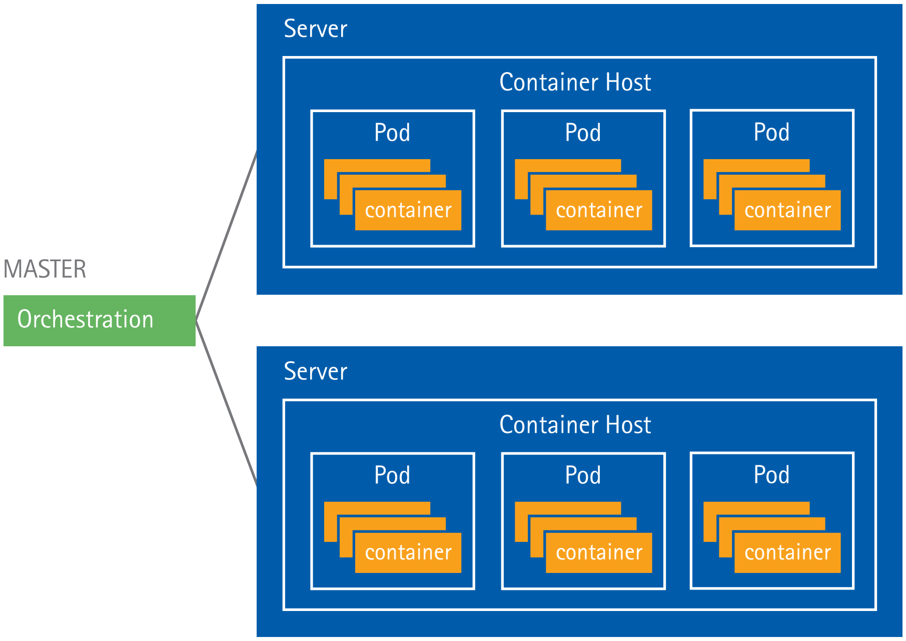
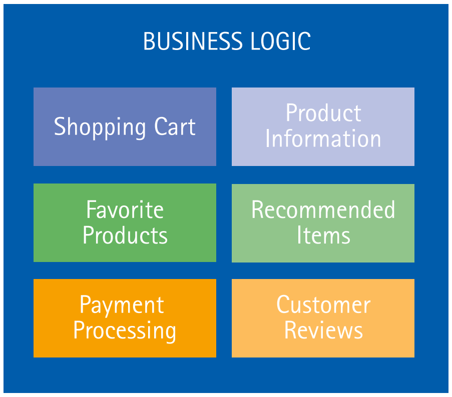


Figure 4: Monolithic business logic



The basic setup consists of one or more container hosts and a master server that controls them. A deployment unit or pod consists of a group of containers running on a single host that share resources, such as storage volumes, shared network name space and Internet protocol. A single compute node may then host several such pods (Figure 3). The orchestration engine handles the spinning up and spinning down of application instances across the cluster, including load balancing and communication between containers.

One of the other benefits of using an orchestration engine is the automated management and self-healing capabilities of containers. When a company defines a pod, the orchestration engine tries to ensure that it is always running. Within a pod, if the container dies or is killed for any reason, the orchestration engine will detect the failure and will create a new container after a few seconds.

Application Pattern for Containers

As enterprises transform to a hybrid-cloud model using containers as units of deployment, their applications must be designed to take advantage of this type of compute environment. Containers enable enterprises to slice compute resources at a finer grain than virtualization. This allows an increased utilization of on-premise infrastructure, while enabling the flexibility and scalability to deploy the same containers to the cloud.

Monolithic architecture

Traditional web applications are designed using monolithic architecture. It typically has three layers or tiers: a presentation layer that users interact with, a business logic layer that holds the application logic, and a persistent layer that stores application data. In this architecture, the business logic layer is made up of many modules that perform business functions, packaged in a single file that is deployed to a server. Figure 4 shows the business logic layer of a typical web store application and the several business functions modules it may have. The modules' codes are invoked as users interact with the web application.

Monolithic applications are simple to develop as most development environment tools default to monolithic architecture. Developing simple monolithic web applications is easy with specific teams responsible for developing specific modules, even though the modules are deployed together.

Figure 5 shows how to scale a monolithic application. Enterprises deploy clones of the application on four different servers and use a load balancer to distribute client requests. This means all the application's modules reside within the application with every application instance containing exactly one instance of the module.

Figure 5: Scaling monolithic business logic

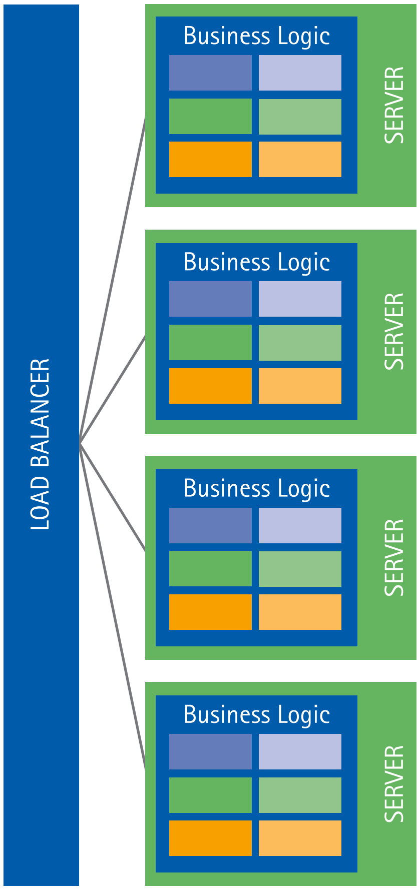
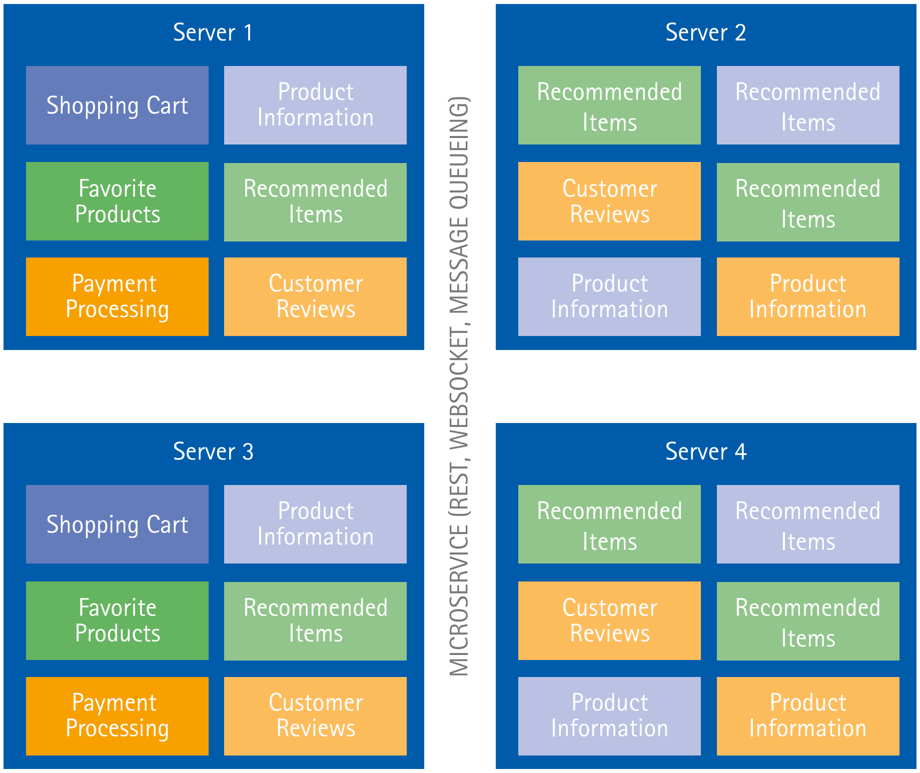


Figure 6: Scaling microservices



While it may be simple to deploy monolithic architecture, it has drawbacks. One disadvantage is that the application is deployed as a whole, cloned many times for scaling. This means if one module that handles specific business logic needs to be updated, the whole application must be re-deployed. This presents release-timing issues and coordination overhead as the other teams responsible for other modules will try to push an update for their module, or else have to wait for the next update cycle.

Another drawback is resource utilization. If a particular business function for the application requires more resources, it cannot be scaled independently from the other modules, as all the modules are part of a larger whole. The scaling of the entire application presents an inefficiency of resources, as a module that does not need to be scaled is scaled along with the modules that do.

Lightweight microservice architecture

Recently, microservice architecture has become a popular pattern for constructing enterprise applications. In microservice architecture, the application is functionally decomposed into a set of coordinated services with business logic functions. Each microservice is a stand-alone execution with no dependency on other microservices. Microservices communicate with each other using lightweight protocols such as REST, WebSocket or MessageQueueing.

Figure 6 shows how to perform horizontal scaling using microservice architecture. Each server runs several microservices that encapsulate a module. Enterprises can perform scaling in the microservice level, instantiating many clones of a business module that requires the most instances, while limiting the number of modules that do not need to be scaled as much, obtaining better resource utilization than monolithic architecture. Resource allocation can also be adjusted at runtime, decreasing the number of instances of a particular module to make compute resources available for another.

Deployment strategy

Microservice architecture works well with the continuous delivery practice of delivering software. In continuous delivery, companies integrate bug fixes and feature enhancement incrementally (continuous integration) with automated testing performed to catch potential issues and check for compliance early on. A new version can then be pushed continuously.

Since microservices are independent of each other, enterprises can update a module in a microservice independently without having to redeploy the other modules. There is also an isolation of execution, where if a microservice happens to fail, a new instance can be redeployed without bringing down the entire application. It is also possible to deploy different versions of a module at the same time. Companies can deploy a newer version and track its performance. If quality metrics are met, they can ramp up the deployment of the new version, while ramping down the number of deployed older modules.

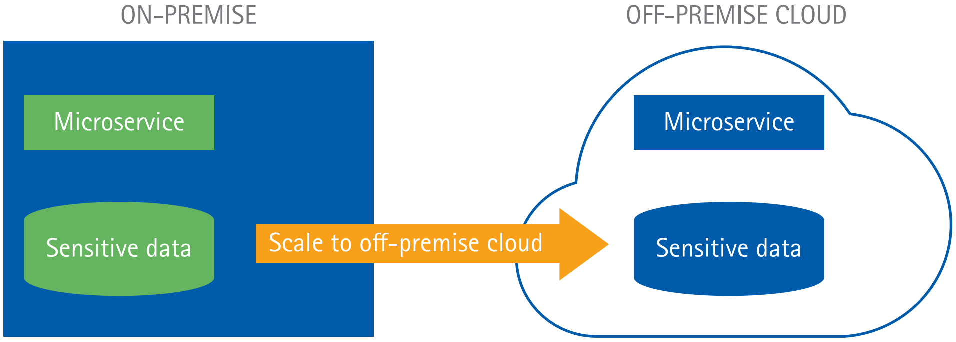
Microservices, containers and scalability

The new container technology is the perfect environment for deploying microservices. Containers will allow for more efficient deployment of microservices as multiple containers can be deployed to a VM (in virtualized enterprise environment), or to native environments (on commodity hardware).

Portability is another benefit of deploying services in containers. With companies beginning to offer container engine as a service, enterprises will be able to deploy the same containers they run on-premise to the cloud, making scalability more dynamic without the overhead of standing up VMs and deploying applications to them.

Since functional decomposition happens in a finer grain level, it is possible for enterprises to scale only services that do not handle sensitive data to off-premise cloud providers, while microservices that handle sensitive data stay on-premise (See Figure 7).

Figure 7: Porting containers



Cross-functional teams

Microservices can also impact how an enterprise organizes their teams. A practice often found in companies that use monolithic architecture is that deliverables are project based, delivered by teams with technological expertise such as a GUI team, business logic team and database team. The interfaces between these teams are defined during design, and then they implement their tasks independently with little interaction. With microservices, enterprises can build cross-functional teams composed of members with different skills, which together are responsible for building and maintaining an end-to-end service. This means different enterprise departments can have a cross-functional team that understands the department's domain create applications specifically for the department, rather than having teams partitioned along technology tiers creating applications for all departments. This allows a team to use the best tools and technology available for their domain. Integration with the rest of the enterprise can be accomplished by using REST application programming interfaces.

The practice of forming cross-functional teams responsible for building, maturing and keeping a service up and running will become more prevalent as enterprises find new ways to monetize data. Currently, a large online retailer uses this practice by creating cross-functional teams in charge of developing data products, and offering them as a service that gets displayed on the store front. Data products that different teams offer include bestseller list, recommendations based on customer's purchase history, recommendations based on customer's wish list, and recommendations based on customer's browsing habit. The retailer's cross-functional teams are composed of data scientists, developers and designers working together to deliver a product as a service.

Conclusion

Although there has been a tremendous popularity in on-premise cloud, companies are realizing the need to not only utilize their existing on-premise infrastructure, but also support on-demand scalability via the public cloud. With large enterprises moving from an on-premise deployment to a hybrid model, there is a need to bridge the gap and make the transition smoother. This point of view prescribes the use of lightweight application containers and microservices architecture to develop scalable applications that can be easily ported from an on-premise to hybrid environment.

Contact us

Vince Dell'Anno
Managing Director
Data Supply Chain Practice
vincent.u.dellanno@accenture.com

Nathan Shetterley
R & D Senior Principal
Accenture Technology Labs - Data Insights
nathan.shetterley@accenture.com

Authors

Carl Dukatz
R & D Principal
Accenture Technology Labs—Data Insights
carl.m.dukatz@accenture.com

Sonali Parthasarathy
R & D Associate Principal
Accenture Technology Labs—Data Insights
sonali.parthasarathy@accenture.com

References

[1] Data Acceleration—Architecture for the Modern Data Supply Chain. http://www.accenture.com/SiteCollectionDocuments/PDF/Accenture-Data-Acceleration-Architecture-Modern-Data-Supply-Chain.pdf

[2] Accenture Technology Vision 2015.

[3] "Platform as a Service: Definition, Taxonomy and Vendor Landscape, 2014," Gartner, August 27, 2014.

[4] "Hype Cycle for Cloud Computing, 2014," Gartner, July 24, 2014.

About Accenture Analytics

Accenture Analytics, part of Accenture Digital, delivers insight-driven outcomes at scale to help organizations improve their performance. With deep industry, functional, business process and technical experience, Accenture Analytics develops innovative consulting and outsourcing services for clients to help ensure they receive returns on their analytics investments. For more information follow us @ISpeakAnalytics and visit www.accenture.com/analytics.

This document makes descriptive reference to trademarks that may be owned by others. The use of such trademarks herein is not an assertion of ownership of such trademarks by Accenture and is not intended to represent or imply the existence of an association between Accenture and the lawful owners of such trademarks.

About Accenture

Accenture is a global management consulting, technology services and outsourcing company, with more than 323,000 people serving clients in more than 120 countries. Combining unparalleled experience, comprehensive capabilities across all industries and business functions, and extensive research on the world's most successful companies, Accenture collaborates with clients to help them become high-performance businesses and governments. The company generated net revenues of US$30.0 billion for the fiscal year ended Aug. 31, 2014. Its home page is www.accenture.com.

About Accenture Technology Labs

Accenture Technology Labs, the dedicated technology research and development (R&D) organization within Accenture, has been turning technology innovation into business results for more than 20 years. Our R&D team explores new and emerging technologies to create a vision of how technology will shape the future and invent the next wave of cutting-edge business solutions. Working closely with Accenture's global network of specialists, Accenture Technology Labs help clients innovate to achieve high performance. The Labs are located in Silicon Valley, California; Sophia Antipolis, France; Arlington, Virginia; Beijing, China and Bangalore, India.
For more information follow us @AccentureLabs and visit www.accenture.com/technologylabs

15-2185U/9-9795