SOA, The Cloud, and Mobility

Service Orientation
Service Orientation

• Service orientation has become a common approach for implementation of distributed, loosely-coupled systems
  – Services provide reusable business functionality via well-defined interfaces.
  – Service consumers are built using functionality from available services.
  – There is a clear separation between service interface and service implementation.
    • Service interface is just as important as service implementation.
  – An service-oriented infrastructure enables discovery, composition, and invocation of services.
  – Protocols are predominantly, but not exclusively, message-based document exchanges.
Benefits Associated with Service Orientation

• Cost-Efficiency
  – Services provide functionality that can be reused many times by many consumers
  – Services become a single point of maintenance and management for common functionality

• Agility
  – Through service discovery mechanisms, developers can find and take advantage of existing services to reduce development times

• Legacy Leverage
  – Separation of service interface from service implementation provides true platform independence

• Adaptability
  – Separation of service interface from service implementation allows for incremental deployment of services and incremental modernization
Tradeoffs

• Security
  • Breaking systems into accessible services, service consumers, and infrastructure components increase the attack surface of a system
  • Using an SOA-based system to enable inter-organizational functionality exposes organizations to threats that were previously hidden by firewalls

• Performance
  • SOA infrastructure adds agility, reusability, and adaptability but is costly in performance, particularly when using notations such as XML
  • The need for increased security requirements degrades performance
Common Misconceptions About SOA

1. SOA provides the complete architecture for a system
2. All legacy systems can be easily integrated into an SOA environment
3. SOA is all about standards and standards are all that is needed
4. The use of standards guarantees interoperability in an SOA environment
5. SOA is all about technology
6. It is very easy to develop applications based on services
7. Testing service-oriented systems is no different than testing any other type of system
8. Everything in a service-oriented system has to be a service
What Has Worked Well in SOA Implementations?

• Standardization
• Loose coupling
• Strategic service identification
• Service discovery mechanisms
• Governance
What Has Not Been Solved With Service Orientation?

• Multi-organizational SOA implementations, e.g.
  – Distributed development tasks, e.g. assurance
  – Multi-organizational concerns, e.g. trust, federation, security

• Standardization on how to specify quality attributes
  – Key for service discovery and SLA management and monitoring

• Support for interoperability at higher levels
  – Above syntactic: semantic and process interoperability

• Automation of service discovery
  – Many efforts, but still not widely used
SOA, The Cloud, and Mobility

Cloud Computing
Cloud Computing

• “A large-scale distributed computing paradigm that is driven by economies of scale, in which a pool of abstracted, virtualized, dynamically-scalable, managed computing power, storage, platforms, and services are delivered on demand to external customers over the Internet.”* 

Cloud Usage Examples

- Make data available to large communities
  - NASDAQ Market Replay places stock data in text form in the cloud so brokers can build their own “programs” that use the data

- Perform computation-intensive activities
  - Washington Post uses the cloud to perform text searches on scanned files

- One-time usage
  - New York Times used cloud resources to convert all their archives to PDF
Cloud Usage Examples

• Support for short-term projects
  – Eli Lilly uses cloud resources for all projects that last less than three months

• Make applications available to mobile users
  – Avon has a cloud-deployed application for sales representatives that can be accessed from mobile devices

• Backup
  – Many organizations use cloud resources for remote backup as a form of contingency
Cloud Usage Examples

• General-purpose software
  – Many organizations use generic information systems such as human resources, or generic software such as e-mail, that fully operate in the cloud
Cloud Computing — Core Concepts

• Distributed computing paradigm
  – Large-scale distributed system

• Resource-based and service-based
  – Provides access to a resource and/or service

• Elastic and on demand
  – Has the ability to scale up or down based on demand

• Economies of scale
  – Large number of nodes reduce the overall cost of resource acquisition and operation

• Utility payment models
  – Based on pay-per-use model

• Based on existing technologies
  – Leverages existing technologies such as virtualization, service-orientation and grid computing
Cloud Computing Types

- Software-as-a-Service (SaaS)
- Platform-as-a-Service (PaaS)
- Infrastructure-as-a-Service (IaaS)

Based on Type of Capability

- Public Cloud
- Private Cloud

Based on Who Can Access Resources
Infrastructure-as-a-Service (IaaS)

• Mainly computational infrastructure available over the internet, such as compute cycles and storage
• Allows organizations and developers to extend their IT infrastructure on an on-demand basis
• Examples of IaaS Providers
  – Amazon Elastic Compute Cloud (EC2)
    • Provides users a special virtual machine (AMI) that can be deployed and run on the EC2 infrastructure
  – Amazon Simple Storage Solution (S3)
    • Provides users access to dynamically scalable storage resources
  – IBM Computing on Demand (CoD)
    • Provides users access to highly configurable servers plus value-added services such as data storage
  – Microsoft Live Mesh
    • Provides users access to a distributed file system; targeted at individual use
Infrastructure-as-a-Service (IaaS) Example

The cloud resource is a part of the system.
Infrastructure-as-a-Service (IaaS) Example

LaaS providers enforce minimal restrictions on their consumers to allow them maximum control and configuration of the resources.

From the organization’s perspective, resources appear to be identical to resources that are owned, operated, and maintained by the organization.

LaaS resources typically provide a variety of interfaces to facilitate interaction.

| Color Code | Within the organization’s control | Outside the organization’s control |
Platform-as-a-Service (PaaS)

- Application development platforms that allow the usage of external resources to create and host applications of a larger scale than an individual or small organization would be able to handle.

- Examples of PaaS providers
  - Akamai EdgePlatform
    - Large distributed computing platform for web application deployment (focus on analysis and monitoring of resources)
  - Force.com
    - Platform to build and run applications and components bought from AppExchange or custom applications
  - Google App Engine
    - Platform to develop and run applications on Google’s infrastructure
  - Microsoft Azure Services Platform
    - On-demand compute and storage services as well as a development platform based on Windows Azure
  - Yahoo! Open Strategy (Y!OS)
    - Platform to develop and web applications on top of the existing Yahoo! Platform (focus on social applications)
The system runs in the cloud

The system will most probably will have external clients

Platform-as-a-Service (PaaS) Example
Platform-as-a-Service (PaaS) Example

From the organization’s perspective, PaaS providers offer significant functionality out-of-the-box.

PaaS provider places restrictions on the consumer by specifying various aspects of the platform, such as the programming languages supported, available libraries, data storage mechanisms, and resource monitoring capabilities.

Includes development and deployment platform.

Color Code

- **Red**: Within the organization’s control
- **Blue**: Outside the organization’s control
Software-as-a-Service (SaaS)

• Model of software deployment in which a third-party provider licenses an application to customers for use as a service on demand

• Examples
  – Google Apps
    • Web-based office tools such as e-mail, calendar and document management tools
  – Salesforce.com
    • Full customer relationship management (CRM) application
  – Zoho
    • Large suite of web-based applications, mostly for enterprise use
Software-as-a-Service (SaaS) Example

The full system runs in the cloud.
The cloud SaaS resource will most probably be one of many systems.
From the organization’s perspective, SaaS enables organizations and developers to use out-of-the-box, business-specific capabilities developed by third parties instead of acquiring, hosting, and managing large software packages or developing proprietary solutions.
Levels of Software-as-a-Service (SaaS)

• Multiple Levels
  – Level 1: An application is specifically run for one customer at an SaaS provider, similar to the traditional ASP (application server provider) model
  – Level 2: The SaaS application is customizable via configuration and one instance of the application serves only one customer.
  – Level 3: The SaaS application is customizable and a single instance of the SaaS application serves multiple tenants.
  – Level 4: The SaaS application is developed as a single instance multi-tenant application and several instances are run in a load-balanced server farm.

• Levels 3 and 4 of SaaS fall under the definition of cloud computing.

---

Cloud Computing Types — Based on Access

• Public
  – Offered as a service, usually over an Internet connection
  – Typically charge a pay-per-use fee
  – Users can scale on-demand and do not need to purchase hardware
  – Cloud providers manage the infrastructure and pool resources into capacity required by consumers

• Private
  – Deployed inside the firewall and managed by the user organization
  – User organization owns the software and hardware running in the cloud
  – User organization manages the cloud and provides cloud resources
  – Resources typically not shared outside the organization and full control is retained by the organization
Additional Cloud Computing Types Proposed by NIST

• NIST defines two additional types of cloud deployment models based on type of access
  – **Community clouds**: shared by multiple organizations and support specific needs and concerns of a community
  – **Hybrid clouds**: Combination of two or more public, private, and community clouds.

• In practice, both community and hybrid cloud are specializations of public and private clouds.

• Additional information: [http://csrc.nist.gov/groups/SNS/cloud-computing/](http://csrc.nist.gov/groups/SNS/cloud-computing/)
<table>
<thead>
<tr>
<th>Drivers for Cloud Computing Adoption</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Scalability</strong></td>
</tr>
<tr>
<td><strong>Elasticity</strong></td>
</tr>
<tr>
<td><strong>Virtualization</strong></td>
</tr>
<tr>
<td><strong>Lower Infrastructure Costs</strong></td>
</tr>
<tr>
<td><strong>Availability</strong></td>
</tr>
<tr>
<td><strong>Collaboration</strong></td>
</tr>
<tr>
<td><strong>Risk Reduction</strong></td>
</tr>
<tr>
<td><strong>Reliability</strong></td>
</tr>
<tr>
<td>Barriers for Cloud Computing Adoption</td>
</tr>
<tr>
<td>--------------------------------------</td>
</tr>
<tr>
<td><strong>Security</strong></td>
</tr>
<tr>
<td><strong>Interoperability</strong></td>
</tr>
<tr>
<td><strong>Resource Control</strong></td>
</tr>
<tr>
<td><strong>Latency</strong></td>
</tr>
<tr>
<td><strong>Platform or Language Constraints</strong></td>
</tr>
<tr>
<td><strong>Regulation</strong></td>
</tr>
</tbody>
</table>
SOA, The Cloud, and Mobility

Mobility
Mobile Computing

• “Taking a computer and all necessary files and software out into the field”*
• “The ability to use computing capability without a pre-defined location and/or connection to a network to publish and/or subscribe to information.”**

* U.S. Bureau of Land Management Online Glossary
** Uwe Vielle, AAMC.org
Mobility is Commonplace

• Mobile computing is ubiquitous
  – Easy access to platforms
  – Easy access to applications
  – Access everywhere

• Wide variety of platforms
  – PDA
  – Smartphone
  – Tablet
  – Ultra-mobile PC
  – Wearable computer
Drivers for Mobility

• Provide instant access to data and/or enterprise capability
  – Increases productivity
  – Increases accuracy
  – Permits instant feedback

• Access to business intelligence
  – In 2008, only 17% of surveyed companies delivered BI to mobile phones
  – In 2010, 23% of surveyed companies were doing so
  – Expectation was that by mid-2011, 50% of companies would provide mobile access to BI*

* Source: Aberdeen Group Surveys
Limitations

• Insufficient bandwidth
  – As mobile computing grows so do the demands on the networks
  – Areas of limited bandwidth
  – Mobile dead zones

• Security standards

• Power consumption

• Transmission interference

• Human interface
SOA, The Cloud, and Mobility

Weaving the Threads
SOA and the Cloud

• Services
  – Some capabilities available in the cloud are steps in business processes
  – Some are not (e.g., persistent store, whole applications

• Discovery
  – Limited numbers of discoverable services
  – There is no consistent policy on making registry entries for every capability in the cloud

• Composition
  – Ability and method of composition depends on the nature of the capabilities
  – Composition may not be possible (more likely for SAAS)

• Invocation
  – Has to be present for a cloud capability to be accessible
Cloud and SOA

• The services in a cloud can be defined as services in a SOA context.
  – As an example, some of the cloud environments presented earlier offer
    web service interfaces to their services (one specific implementation of
    SOA)
• From an architectural perspective, a cloud infrastructure could be built
  on top on an SOA infrastructure by adding a layer or virtualization and
  self-provisioning.
  – Some even consider SOA adoption a pre-requisite for cloud computing
• Similarly, an SOA could be built in the cloud using the IAAS model,
  assuming the cloud provides appropriate infrastructure and
  infrastructure services (e.g., discovery)
Mobility and the Cloud

• Increased dependence on mobile computing leads to greater dependence on services in the cloud
• Tendency toward use of public clouds due to cost of maintaining a private cloud infrastructure
Challenges -1

• Rapid changes in the environment
  – Design the services to be context aware, but
    • No agreement on how to represent context
    • No real implementation of contextual service discovery mechanisms
• Multiple deployment choices demand high configurability
  – Design services for runtime discovery and composition, but
    • No standards for semantics
    • Tool support is very weak
    • No relevant examples of large-scale use
• Sensitive content and protection from attack requires high security
  – Secure the infrastructures and services, but
    • Federated identity management, security policies and policy enforcement, and trust establishment and trust brokering in service-oriented environments are all active areas of research
Challenges - 2

• Information overload
  – Access to too much information makes it hard to find the right information
  – Inability to predict user needs
    • Limited experience with edge-enabled “programming”

• Critical needs
  – Bring the cloud closer to the application, but
    • The notion of a cloudlet is still highly experimental
    • Latency may still be an issue
Cloudlet Concept

- Offloading expensive computation to the cloud for remote execution

Similar to traditional client server.

Very common and mature architectural pattern used in today’s mobile applications.

Still an area of research and is still not widely adopted by the mainstream.
Conclusion

• SOA and cloud technologies offer a lot of promise
• SOA and cloud are largely independent of each other
• Increased demand for mobile computing will drive toward service- and cloud-based environments
• There are many benefits such as agility and cost savings
• There are also significant barriers such as securing the systems
Resources

• Courses
  – SEI SOA-related courses
    [http://www.sei.cmu.edu/training/find/courses.cfm?category=Service-Oriented%20Architecture](http://www.sei.cmu.edu/training/find/courses.cfm?category=Service-Oriented%20Architecture) (even courses as a service with e-learning)
  – Others, for example SOA School [http://www.soaschool.com](http://www.soaschool.com) also [http://www.cloudschool.com](http://www.cloudschool.com)

• Podcasts such as
  [http://www.sei.cmu.edu/library/assets/podcasts/SMART%20podcast.mp3](http://www.sei.cmu.edu/library/assets/podcasts/SMART%20podcast.mp3)

• Books
  – Thomas Erl is a prolific author (Prentice-Hall Service Oriented Computing Series) [http://www.soabooks.com](http://www.soabooks.com)

• Articles such as [http://apsblog.burtongroup.com/2009/01/soa-is-dead-long-live-services.html](http://apsblog.burtongroup.com/2009/01/soa-is-dead-long-live-services.html)
Contact Information

Patrick R.H. Place
Research, Technology and Systems Solutions (RTSS) Program
Software Engineering Institute
4500 Fifth Avenue
Pittsburgh, PA 15213-2612
USA

Phone: +1 412-268-7746
Email: prp@sei.cmu.edu
• NO WARRANTY

• THIS MATERIAL OF CARNEGIE MELLON UNIVERSITY AND ITS SOFTWARE ENGINEERING INSTITUTE IS FURNISHED ON AN “AS-IS” BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

• Use of any trademarks in this presentation is not intended in any way to infringe on the rights of the trademark holder.

• This Presentation may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

• This work was created in the performance of Federal Government Contract Number FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The Government of the United States has a royalty-free government-purpose license to use, duplicate, or disclose the work, in whole or in part and in any manner, and to have or permit others to do so, for government purposes pursuant to the copyright license under the clause at 252.227-7013.