Emerging Software Development and Acquisition Approaches: Panacea or Villain

Software Engineering Institute
Carnegie Mellon University

Dennis Smith
May 16, 2011
1. REPORT DATE
16 MAY 2011

2. REPORT TYPE

3. DATES COVERED
00-00-2011 to 00-00-2011

4. TITLE AND SUBTITLE
Emerging Software Development and Acquisition Approaches: Panacea or Villain

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

6. AUTHOR(S)

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Carnegie Mellon University, Software Engineering Institute, Pittsburgh, PA, 15213

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR’S ACRONYM(S)

11. SPONSOR/MONITOR’S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
Presented at the 23rd Systems and Software Technology Conference (SSTC), 16-19 May 2011, Salt Lake City, UT. Sponsored in part by the USAF. U.S. Government or Federal Rights License

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:
a. REPORT
unclassified

b. ABSTRACT
unclassified
c. THIS PAGE
unclassified

17. LIMITATION OF ABSTRACT
Same as Report (SAR)

18. NUMBER OF PAGES
30

19a. NAME OF RESPONSIBLE PERSON

Standard Form 298 (Rev. 8-98)
Proscribed by ANSI Std Z39-18
Agenda

DoD needs and challenges
Potential approaches to address challenges (current progress and gaps)
  • Service orientation
  • Cloud computing
  • User-controlled adaptation in field

Conclusions
Mission Planning

Evolving Situation

Dismounted Warfighters

Software delivered to warfighters does not keep pace with changing missions.

Warfighters cannot get the relevant information they need at the time they need it.

The closer that warfighters get to combat, the fewer resources they have available.
Agenda

DoD needs and challenges

Potential approaches to address challenges (current progress and gaps)
  • Service orientation
  • Cloud computing
  • User-controlled adaptation in field

Conclusions
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 SOA infrastructure enables discovery, composition, and invocation of services.
• Protocols are predominantly, but not exclusively, message-based document exchanges.
Components of a Service-Oriented System

- **End User Application**
- **Portal**
- **Internal System**
- **External Consumer**

**SOA Infrastructure**
- **Service A**
- **Service B**
- **Service C**
- **Service D**

**Enterprise Information System**
- **Legacy or New Service Code**
- **External System**

**Service Consumers**

**Infrastructure**

**Service Interfaces**

**Service Implementation**

**Internal Users**

**Discovered Security**

**Data Transformation**

**Internet**
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

• Via 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
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
Service-Oriented 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
<table>
<thead>
<tr>
<th>DoD Vision and Needs</th>
<th>SOA Technology</th>
<th>State of the Practice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Highly adaptable to changes in the environment</td>
<td>Design for Context Awareness</td>
<td>No agreement on how to represent context. No real implementation of contextual service discovery mechanisms.</td>
</tr>
<tr>
<td>Highly configurable to deal with multiple deployment choices</td>
<td>Design for Runtime Discovery and Composition</td>
<td>No standard for semantics. Tool support is very weak. No relevant examples of large-scale use.</td>
</tr>
<tr>
<td>Highly secure due to potentially classified content and malicious attacks</td>
<td>Securing SOA Infrastructures and Services</td>
<td>Federated identity management, security policies and policy enforcement, and trust establishment and trust brokering in SOA environments are all active areas of research.</td>
</tr>
<tr>
<td>Highly reliable and precise in a mission-critical context</td>
<td>Real-Time SOA</td>
<td>Current, widely-used SOA implementation technologies do not meet real-time requirements.</td>
</tr>
</tbody>
</table>
Extension of SOA to Address DoD Needs
CoT SOAP UDP App V1 (Camp Roberts – May 2010 TNT) – Fixed Station

- Assets (UAVs, cars) track a hostile vehicle and post CoT messages (video, location etc) to the CoT SOAP Server
- CoT SOAP Server consume raw CoT messages and provides CoT data as SOAP-over-UDP web service
- Android phone consume SOAP messages, processes and displays them
Experimental Engineering Decisions

Transport layer protocol defines interfaces available to applications that allow end-to-end communications; TCP is the most familiar and is best suited for situations with reliable transmission (solid network infrastructure)

- However, UDP was selected because TCP is not suited to situations where packet loss, mis-ordering, or garbling are more common
  - UDP tradeoff: it does not provide error correction

SOA uses two common messaging protocols SOAP and REST

- REST is simpler and increasingly more common
- We selected SOAP also hoping to take advantage of well-defined specifications, open source implementations, and support for security.
  - gSOAP on the CoT router-side and a modified kSOAP on the Android side
Agenda

DoD needs and challenges

Potential approaches to address challenges (current progress and gaps)
  • Service orientation
  • **Cloud computing**
  • User-controlled adaptation in field

Conclusions
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.”*

DoD Cloud Implementations

DISA
  - IaaS private cloud
  - Allows authorized users (government personnel and contractors) to use a
    credit card to purchase a computing environment and be up and running
    within 24 hours
  - PaaS/SaaS private cloud
  - Collaborative development and use of open source and DoD community
    source software

NSA
- Private cloud (based on Google’s Hadoop) to support a new collaborative
  intelligence data sharing application
- Distributed data centers host large amounts of disparate data that can be
  tagged, searched and analyzed by users
## Drivers for Cloud Computing Adoption

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scalability</td>
<td>Organizations have access to a large amount of resources that scale based on user demand</td>
</tr>
<tr>
<td>Elasticity</td>
<td>Organization’s can manually or dynamically decide on resource utilization based on changing needs</td>
</tr>
<tr>
<td>Virtualization</td>
<td>Each user has a single view of the available resources, independently of how they are arranged in terms of physical devices</td>
</tr>
<tr>
<td>Lower Infrastructure Costs</td>
<td>The pay-per-use model allows an organization to only pay for the resources they need with basically no investment in the physical resources available in the cloud. There are no infrastructure maintenance or upgrade costs</td>
</tr>
<tr>
<td>Availability</td>
<td>Organizations have the ability for the user to access data and applications from around the globe</td>
</tr>
<tr>
<td>Collaboration</td>
<td>Organizations are starting to see the cloud as a way to work simultaneously on common data and information</td>
</tr>
<tr>
<td>Risk Reduction</td>
<td>Organizations can use the cloud to test ideas and concepts before making major investments in technology</td>
</tr>
</tbody>
</table>
## Barriers for Cloud Computing Adoption

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security</td>
<td>The key concern is data privacy: organizations do not have control of or know where their data is being stored</td>
</tr>
<tr>
<td>Interoperability</td>
<td>A universal set of standards and/or interfaces has not yet been defined, resulting in a significant risk of vendor lock-in</td>
</tr>
<tr>
<td>Resource Control</td>
<td>The amount of control that the organization has over the cloud environment varies greatly</td>
</tr>
<tr>
<td>Latency</td>
<td>All access to the cloud is done via the internet, introducing latency into every communication between the user and the environment</td>
</tr>
<tr>
<td>Reliability</td>
<td>Many existing cloud infrastructures leverage commodity hardware that is known to fail unexpectedly (NOTE: Disappearing as a barrier)</td>
</tr>
<tr>
<td>Platform or Language Constraints</td>
<td>Some cloud environments provide support for specific platforms and languages only</td>
</tr>
<tr>
<td>Regulation</td>
<td>There are concerns in the cloud computing community over jurisdiction, data protection, fair information practices, and international data transfer</td>
</tr>
</tbody>
</table>
SaaS: Examples of Architecture and Design

Questions:

- What type of client is used to interact with the SaaS resource?
- How does the cloud system fit with the existing infrastructure?
- What data adapters and transformers are necessary to interoperate with other systems?
- What additional mechanisms need to be put in place to monitor system performance and usage?
- Is the SaaS security architecture compatible with the organization’s security architecture?
Cloud Challenges

Cloud Computing is in essence an economic model
- It is a different way to acquire and manage IT resources

There are multiple cloud providers— the cloud is real
- Currently most cloud consumers are small enterprises
- Large enterprises are exploring private clouds
- The number of providers will most probably grow as people start seeing greater savings and improvements to reduce adoption barriers

Cloud Computing adoption requires cost/benefit/risk analysis to determine
- What resources to move to the cloud (if any)
- What situations warrant use of cloud resources, even for one-time situations
- Implementation of private clouds vs. usage of public clouds
- What risks are associated with using resources on the cloud
- What risks are associated to providing resources in the cloud
Cloud Challenges 2

Decisions from a cloud consumer perspective depend on
• Required control level
• Required security level
• Compatibility with local infrastructure

Decisions from a cloud provider perspective depend on
• Market/user characteristics
• Established SLAs
• Available technology

In general, these are not fully technical decisions
• Processes — especially engineering practices
• Governance
• Cost/Benefit analysis
Research on Cloudlets for Resource Optimization for Mobile Platforms at the Edge

The closer you get to combat, the fewer computation, energy and network resources you have available

Battery life becomes critical:
Conserving energy is a primary concern

Computational capability is limited:
Mobile elements will always be poor in compute resources (CPU, memory, storage) as compared to static elements

Goal
Develop software-based strategies for optimization of energy and CPU consumption that consider both the individual device and nearby peer devices

References: [Satyanarayanan 1996], [NAP 1997], [Silven 2007], [Ravi 2008], [Fuller 2011]
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.
Cloud Computing in Tactical Environments

1. Capture Picture
2. Transfer Overlay
3. Send Picture
4. Return Results

Android-Based Smartphone

Cloudlet

- Face Recognition Front-End Application
- Face Recognition Processing Back-End Overlay
- KVM or Virtual Box
- Linux
Agenda

DoD needs and challenges
Potential approaches to address challenges (current progress and gaps)
  • Service orientation
  • Cloud computing
  • User-controlled adaptation in field
Conclusions
User-Controlled System Adaptation at the Edge

Capabilities delivered to mobile devices at the edge do not keep pace with rapidly changing mission needs

- Mismatch between the mission needs and the capabilities provided by the tools
- Warfighters currently cobble together solutions in theater to meet emerging needs
- Warfighter-created solutions are of uncertain quality and can threaten the mission

Goal
Develop end-user programming and architecture strategies for rapid adaption and validation of capabilities
User-Controlled System Adaptation at the Edge

End-User Programming Capability for Handheld Devices that is Usable by Warfighters

Develop end-user strategies to support adaptation of apps on handheld devices

- Employ *natural programming* to gather requirements for end-user adaptation [Myers 2008]

Create a domain-specific end-user programming environment that supports adaptation of mobile apps

- Enable dynamic creation of customized forms
- Incorporate additional sensors, data formats, more complex rules and layouts
User-Controlled System Adaptation at the Edge³

End-User Validation Strategies to Achieve Confidence in the Correct Operation of Handheld Apps Adapted by the Warfighter

Develop enhanced validation strategies for improved confidence

1. Provide feedback to warfighters on the effects of their modifications
2. Enforce firewalls on trusted parts of the system so that only new (untrusted) parts must be revalidated
3. Apply static analysis to verify selected properties of modified applications
4. Implement real-time monitoring to ensure that the application operates within its constraints
Agenda

DoD needs and challenges
Potential approaches to address challenges (current progress and gaps)
  • Service orientation
  • Cloud computing
  • User-controlled adaptation in field

Conclusions
Conclusions

DoD battlefield needs require

- Flexible adaptation
- Integration between diverse platforms and sources
- Discovery of available data and sensors
- Exploitation of mobile platforms
- Conservation of scarce resources of power and computation

Technologies and approaches offer potential to address these needs

- Mobile platforms
- Service orientation
- Rapid adaptation
- Cloudlets

These technologies are maturing in enterprise solutions (though they still have challenges)

- Initial experimental results offer a step forward for the future
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.