Software Architecture Fundamentals: Technical, Business, and Social Influences

Rob Wojcik
Software Engineering Institute

Webinar
July 8th, 2010
# Software Architecture Fundamentals: Technical, Business, and Social Influences

## 1. REPORT DATE
08 JUL 2010

## 2. REPORT TYPE

## 3. DATES COVERED
00-00-2010 to 00-00-2010

## 4. TITLE AND SUBTITLE
Software Architecture Fundamentals: Technical, Business, and Social Influences

## 5. AUTHOR(S)

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

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

## 8. PERFORMING ORGANIZATION REPORT NUMBER

## 9. SPONSOR/MONITOR'S ACRONYM(S)

## 10. SPONSOR/MONITOR'S REPORT NUMBER(S)

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

## 12. SUPPLEMENTARY NOTES

## 13. ABSTRACT

## 14. SUBJECT TERMS

## 15. SECURITY CLASSIFICATION OF:

<table>
<thead>
<tr>
<th>a. REPORT</th>
<th>b. ABSTRACT</th>
<th>c. THIS PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>unclassified</td>
<td>unclassified</td>
<td>unclassified</td>
</tr>
</tbody>
</table>

## 16. LIMITATION OF ABSTRACT
Same as Report (SAR)

## 17. NUMBER OF PAGES
34
Today’s Speaker

Rob Wojcik is a member of the technical staff in the Research, Technology, and System Solutions Program at the SEI, a position he has held since 2004. In his current position, he performs training and consulting in software architecture technology and software architecture evaluations. Prior to his position at the SEI, over twenty five years of his career were dedicated to developing software systems that utilize traditional as well as artificial intelligence and object oriented technologies.
Polling Question #1

How did you hear about this webinar?

1. Email invitation from the SEI
2. SEI Website
3. Website with webinar calendar (i.e., www.webinar-directory.com)
4. Social Media (i.e., LinkedIn, Twitter)
5. SEI Member Bulletin
Software Engineering Institute (SEI)

Department of Defense R&D Laboratory (FFRDC)
Created in 1984
Under contract to Carnegie Mellon University
Offices in Pittsburgh, PA; Washington, DC; and Frankfurt, Germany
SEI Mission: advance software and related disciplines to ensure the development and operation of systems with predictable and improved cost, schedule, and quality.
Research, Technology, and System Solutions (RTSS) Program

Vision

• Enable assured and flexible system capabilities at all scales.

Mission

• Focus on the structure and behavior of software-reliant systems
Portfolio of RTSS Program Work

Initiatives
- Architecture-Centric Engineering
- System of Systems Practice
- System of Systems Software Assurance
- Product Line Practice

Cross-Cutting Efforts
- Concept Lab
- Integrating Solutions
- Ultra-Large-Scale Systems

Independent Research and Development
Today’s Topics

- What is Software Architecture?
- Why is Software Architecture Important?
- Which Requirements Are Most Important To Architectural Design?
- What Else Influences Software Architecture?

I’ll take questions at the end of the presentation.
How Much Do You Know?

a) I know a whole lot about software architecture
b) I know enough about software architecture to get by
c) I know very little about software architecture
d) What the heck is software architecture?
Today’s Topics

- **What is Software Architecture?**
- Why is Software Architecture Important?
- Which Requirements Are Most Important To Architectural Design?
- What Else Influences Software Architecture?
Many Definitions for Software Architecture

Alfred: Software architecture consists of the rules and principles for how a system is decomposed into its component parts, the rationale for how responsibilities are allocated among those parts, and the policies and mechanisms that coordinate the interactions between those parts as they collaborate to fulfill the purpose of the system. Software architecture is at once the partitioning of a system into its significant elements, and the organization and integration of those elements into a cohesive whole.

RUP: the set of significant decisions about the organization of a software system, the selection of the structural elements and their interfaces by which the system is composed, together with their behavior as specified in the collaborations among those elements, the composition of these structural and behavioral elements into progressively larger subsystems, and the architectural style that guides this organization---these elements and their interfaces, their collaborations, and their composition

Ramanujam: an iterative framework between software components required to meet the stated objectives of the business, in terms of cost to develop/maintain the software components, time to market, and life expectancy of the components

Pandey: the blueprint of the various framework components that coordinate together to satisfy the design guidelines of a specific domain

Software Engineering Institute: the structure or structures of the system, which comprise the software elements, the externally visible properties of those elements, and the relationships among them

ANSI/IEEE: the fundamental organization of a system, embodied in its components, their relationships to each other and the environment, and the principles governing its design and evolution

Adabala: a style that is proven scientifically and adopted by the engineering discipline, with which a software is developed so as to sustain and adopt to the growing needs of the industry from time to time...

Ahmed: a coherent set of abstract patterns, or principles, guiding the design of each aspect of a large software system...

Matthaeus: A configurable skeleton of any kind of software beast on which you hang implementation specific muscle to make it live

Riemenschneider: the mapping from the problem space to the solution space, where the problem space is devoid of implementation concerns and the solution space is the sum total of all implementation concerns

Mulvaney: a set of implementation elements together with the mechanisms through which they collaborate to provide the system's required functionality
Software architecture

- is an abstraction that describes software elements
- addresses the roles, responsibilities, behaviors and properties of software elements
- addresses the relationships between software elements
- shows what software elements provide to and require from each other
- shows the relationship to non-software elements
- is described from many different perspectives
Every software system has an architecture.

A software architecture is not inherently good or bad.
Today’s Topics

• What is Software Architecture?
• Why is Software Architecture Important?
• Which Requirements Are Most Important To Architectural Design?
• What Else Influences Software Architecture?
Why Is Software Architecture Important?

It’s a vehicle for communication.

It’s a manifestation of earliest design decisions that
- defines implementation constraints
- relates to organizational structure
- provides the basis for project artifacts and activities
- permits/precludes achieving requirements
- allows us to predict system qualities
- allows us to control complexity
- allows us to reason about and manage change
- allows us to develop a skeletal system
- provides a sound basis for cost and schedule estimates

It’s a transferable, reusable abstraction.
Today’s Topics

- What is Software Architecture?
- Why is Software Architecture Important?
- **Which Requirements Are Most Important To Architectural Design?**
- What Else Influences Software Architecture?
Which Requirements Are Most Important to Architectural Design?

Functional requirements
Design constraints
Quality attribute requirements

Architectural Design

Software Architecture

What determines whether these requirements are met?
Which requirements are the most important when it comes to structuring an architecture?
Something to Consider

What’s wrong with designing a system that has one big source module, one big object module, and one big executable as long as it functions properly?

- buildability
- modifiability
- testability
- complexity
- maintainability

<table>
<thead>
<tr>
<th>Module</th>
<th>Others?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Others?</td>
</tr>
<tr>
<td>2</td>
<td>Others?</td>
</tr>
<tr>
<td>3</td>
<td>Others?</td>
</tr>
<tr>
<td>...</td>
<td>Others?</td>
</tr>
<tr>
<td>1,999,999</td>
<td>Others?</td>
</tr>
<tr>
<td>2,000,000</td>
<td>Others?</td>
</tr>
</tbody>
</table>

portability
reliability
distributability
availability
reusability

Which requirements do you think would be negatively impacted by this “design”?
Here’s the Point!

If functionality is the only thing that matters, any software architecture will do!

It’s the requirements that are above and beyond functionality that require us to structure an architecture. They include:

- design constraints
- quality attributes
Eliciting quality attribute requirements

Yet these are critical to architectural design!

Eliciting functional requirements and design constraints

Good Job

Bad Job
Difficulties in Eliciting Quality Attribute Requirements

Non-Operational requirements

- “The system must be easy to use.”
- “The system must have high performance.”
- “The system must be portable.”

Debating the quality attribute to which a system behavior belongs

- “The system must process 10,000 messages per second.”

Vocabulary variations

- Everyone knows what “high performance” means, right?
Today’s Topics

• What is Software Architecture?
• Why is Software Architecture Important?
• Which Requirements Are Most Important To Architectural Design?
• What Else Influences Software Architecture?
Software architecture is influenced by the technical, business, and social environment.
Examples of Other Influences

Stakeholders

- customers, users, managers, marketing, developers, maintainers, etc.

Development organization

- immediate and long term business goals
- organizational structure

Technical environment

- object oriented, WWW, intelligent agents, EJB, service oriented, J2EE, thin client, .NET, etc.

Background and experience

- architect and organizational experience
- education and training
An architecture can influence the technical, business, and social environment.
Examples of What Architecture Can Influence

Development organization
  • structure, goals, artifacts, etc.

Stakeholder requirements
  • demand for similar features, existing components and system

Technical environment
  • relational databases, WWW, service oriented architectures, etc.

Background and experience
  • promote approaches that have been successful
  • reject approaches that have failed
Understanding these influences helps us to plan for and manage change throughout the lifetime of a system.
Conclusion

Software architecture is important!

Every software system has an architecture!

Quality attribute requirements are critical!

Requirements aren’t the only things that influence software architectures!
For More Information

Software Architecture in Practice, 2nd edition
written by Len Bass, Paul Clements, & Rick Kazman
and published by Addison-Wesley as part of the
SEI Series in Software Engineering

Other information is provided at
http://www.sei.cmu.edu/architecture/.

Contact  Rob Wojcik
Software Engineering Institute
rwojcik@sei.cmu.edu
Proof of your skill from a world leader in software engineering.

Become an SEI-Certified ATAM Leader.

www.sei.cmu.edu/certification/architecture/atam
Want a Closer Connection to the SEI?

Become an SEI Member!

www.sei.cmu.edu/membership
CERT's Podcast Series: Security for Business Leaders
www.cert.org/podcast/
For more than 20 years, the SEI has been at the forefront of software engineering.

By becoming an SEI Partner, you join forces with a software engineering pioneer and an institute whose credibility provides a solid foundation during uncertain economic times.

SEI Partner Network

www.sei.cmu.edu/partners
Do you have the knowledge you need?

SEI Training

www.sei.cmu.edu/training