

The Importance of Software Architecture in the Acquisition Process

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Len Bass
Currently visiting NICTA, Sydney



Overview

My fundamental argument is that software is critical for system development.

Software architecture is critical for software development.

Therefore the acquisition process must pay attention to software architectural issues during concept formulation, contract preparation and after award.



The importance of software in system development

Today, all non-trivial systems have significant amounts of software

The U. S. Department of Defense estimates that software now accounts for 40% of all research, development, test and evaluation (RDT&E) spending

1 billion GBP of the 7 billion GBP spent on network-enabled warfare systems was for software – UK Defence Technology Strategy report from 2006. The report acknowledged explicitly that the amount was going to increase.

What percentage of Australian defence acquisition dollars go for software?

Is the fact that software is essential to the construction of systems in dispute?



What is Software Architecture?

Definition:

The software architecture of a program or computing system is the structure or structures of the system, which comprise software elements, the externally visible properties of those elements, and the relationships among them.

Key ideas:

- Externally visible properties
- Relationships among elements
- Multiple different structures

Uses

- Guide to construction
- Artifact for analysis



What are quality attributes?

From ISO 9126: Software Engineering – Product quality

Quality attributes are properties of a system such as

- Functionality
- Reliability
- Usability
- Efficiency
- Maintainability
- Portability

ISO 9126 lists 22 different quality attributes.

Quality attributes of a system determine whether a system is acceptable to

- Users in terms of how well it performs in use
- Acquirers and developers in terms of the difficulty of construction



Quality attributes and Software Architecture

The ability of a system to achieve desired qualities is constrained by its software architecture

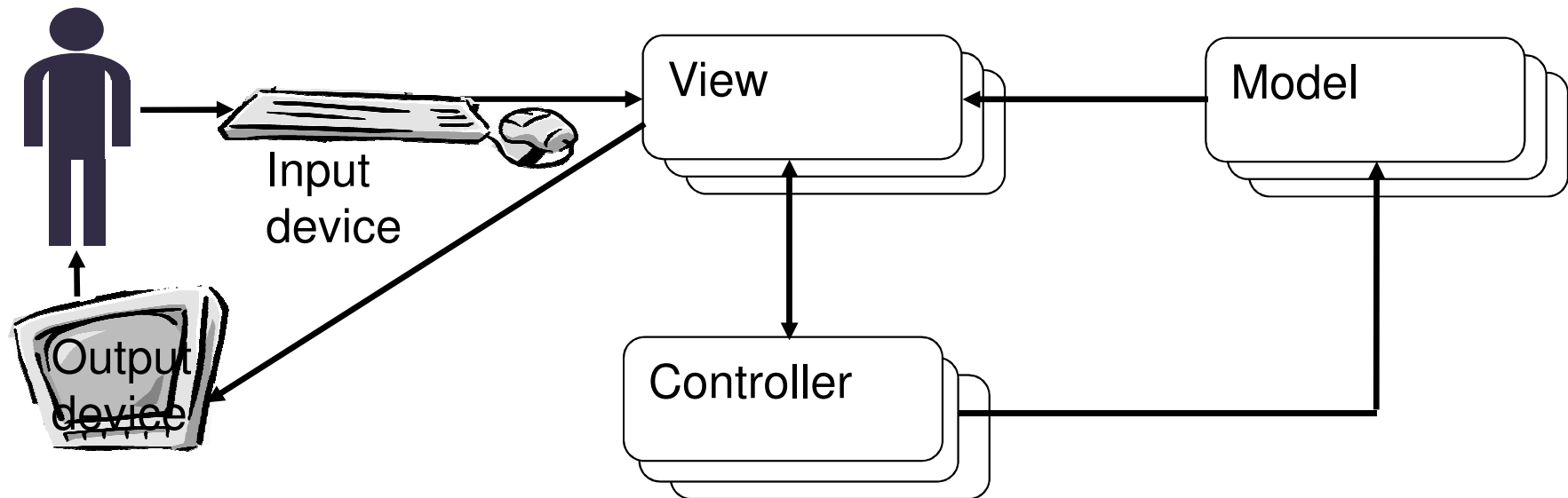
A software architecture can be designed to support particular quality attributes

A software architecture can be analyzed to determine how well it supports particular quality attributes



Example - MVC

Model View Controller (MVC) is a standard “architectural pattern”.



Model

- Application state and functionality

View

- Renders models, sends user gestures to Controller

Controller

- Transforms interactions with the View into actions to be performed by the model, selects View



What can be said about maintainability from MVC

Some changes are easy:

- Add a new method of viewing the data – that is what the View is for
- Change layout of screen – localized in the View
- Change order of screens – Controller determines which view to display next

Some changes are difficult:

- Add cancel to system that does not currently have cancel. This involves all three components
- Add ability to group data items. This involves all three components.

I.e. Maintainability of system based on MVC is constrained by use of that architectural pattern. The pattern can be analyzed to determine how well a system constructed using MVC will support different quality attribute scenarios.



Quality attribute requirements are the ones that “drive” the software architecture design.

If only concern was achievement of functionality, then a monolithic system would suffice

But we commonly see

- Distribution of function to different machines - (performance)
- Redundancy of function - (reliability)
- Hardware abstraction layer – (modifiability)
- Firewalls (security)
- ...



Importance of Software Architecture in Software Development

Software architecture is first artifact produced during the design process.

Software architecture can be designed to achieve particular qualities.

Software architecture can be analyzed to see how well it achieves particular qualities.

==> software development process must consider software architecture as central artifact.

Leads to several other areas of interest

- Getting the correct/important requirements
- Representing architecture
- Evaluation/design of software architecture
- Conformance of implementation to architecture
- Software architecture as a guide to cost models



Getting the correct requirements

Quality requirements drive software architecture design => important that quality requirements are specific enough to enable design.

Meaningless quality requirements:

- “the system shall be modular”
- “the system shall be secure”
- “the system shall be high performance”

There is no way to test these requirements.

Quality requirements – no less than functional requirements – must be testable.

Quality attribute scenarios are a form for specifying quality attributes in a testable fashion.



Getting the important requirements

Every functional requirement has a collection of associated quality requirements: who can execute this function, what are the reliability requirements for this function, what kinds of modifications can be expected for this function, etc.

Thousands of such requirements

“Architecturally significant requirements” are those quality requirements that drive the architecture.

Must locate the architecturally significant requirements among all of the quality attribute requirements in order to make design/evaluation problem manageable.

Architecturally significant requirements are the requirements that have high business/mission value and are pervasive in terms of design.



Software architecture design/evaluation

Software architecture evaluation is designed to determine how well a system will meet its quality attribute requirements.

Since business/mission goals typically are realized through quality attribute requirements, an evaluation enables discovery of differences between the goals for the system and the system as designed.

Evaluation can be done on an on-going basis by development team or on a “big-bang” basis involving external evaluators and multiple stakeholders.

It is also possible to use quality attribute goals as a basis for generating the design of system.

In the design case, architecturally significant requirements are used to generate an initial architecture. This architecture is refined until the architecture is suitable for its uses (instructions to developers and analysis).



Representing architecture

If the architecture is to serve as a plan for implementation and as an artifact to be analyzed, it must be represented

Software architectures have several different important representations

- **Module.** What are the elements of the architecture, what does each element do, and what are the relations among the elements. This is a static view
- **Component and connector.** How do the elements of the architecture interact during execution. What are the “execute before” relations among the elements, How is control passed among the elements? Which elements may execute in parallel? This is a dynamic view
- **Allocation.** How are the components allocated to hardware? What are the process and network elements necessary for the system to execute?



Conformance of implementation to architecture

Architecture acts as a guide to implementation

Means that implementation must conform to the architecture or the benefits are lost.

Several commercial tools perform architecture conformance tests, e.g. Lattix, Coverity Architect, Structure 101.

- These tools test conformance to module description.
- They can look for dependency among modules and look for layers, circular dependencies, and other poor programming practices.
- They are limited in terms of detecting problems with many quality attributes
 - performance
 - reliability.



Software Architecture as a guide to cost modeling

Most cost models are based on size of system to be developed – whether Lines of Code or Function Points. This is equivalent to estimating cost of producing equipment based on its size.

Software architecture enables estimating cost based on cost of producing various elements of the architecture. This is equivalent to estimating cost of producing equipment based on a bill of materials.



Acquisition perspective

Given the importance of software architecture to software development, there are three portions of the life cycle where architectural knowledge is important

- Requirements
- Proposal evaluation
- Architecture evaluation of system during development



Requirements

Developers (and hence proposal writers) need architecturally significant requirements during design

Architecturally significant requirements are, typically, quality attribute requirements

==>

- Requirements information must include quality attribute requirements
 - In a testable form
 - With some indication of importance/provenance
- Business/mission goals must be available to proposers/designers

Quality attribute requirements are, typically, crosscutting ==>

End to end uses of the system are important to proposers/developers.



Proposal Evaluation

Quality attributes determine how well a system is fit for use both by users and by developers

If software architecture constrains what quality attributes are possible then a portion of proposal evaluation should be a software architecture evaluation – requires

- architecture as a portion of the official submission
- trained personnel



During Development

Acquirers should evaluate architecture during development

- To ensure designed system satisfies mission goals
- To ensure conformance
- To enable adaptation to changes in mission



Resources - Books

- Software Architecture in Practice – Bass, Clements, Kazman
- A Software Architecture Primer - Reekie, McAdam
- Software Architecture: Foundations, Theory, and Practice –Taylor, Medvidovic, Dashofy
- Software Systems Architecture: Working With Stakeholders Using Viewpoints and Perspectives – Rozanski, Woods
- The Process of Software Architecting – Eeles, Cripps
- Essential Software Architecture –Gorton
- The Art of Software Architecture: Design Methods and Techniques – Albin
- Architecting Software Intensive Systems: A Practitioners Guide –Tony Lattenze
- Risk Centric Software Architecture- Fairbanks (in press)
- Documenting Software Architecture – Clements, et al.



Resources – courses

Courses offered by Software Engineering Institute

Courses offered by International Association of Software Architects



US Army Mandate

“...all Program Executive Offices (PEOs) will appoint a Chief Software Architect (CSWA) who shall be responsible for oversight and management of software development within the PEO’s portfolio....The CSWA shall provide guidance for software architecture design and reviews...The CSWA will complete the Software Engineering Institute software architecture course series for certification as a Software Architect Professional or equivalent.”

Gen N. R. Thompson, Principal Military Deputy of ASA(ALT)

ASA(ALT) is the Assistant Secretary of the Army for Acquisition, Logistics, and Technology



More Questions?

