Software Architecture in Practice

Chapter 10: Software Architecture Evaluation

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213-3890

Sponsored by the U.S. Department of Defense

Lecture Objectives

This lecture will enable students to

- **explain when and why to analyze software architectures**
- **list and explain the steps of the SAAM evaluation method**
- **understand how SAAM works in practice**
- **appreciate costs and benefits of architectural reviews**
- **give examples of review technique types**
- **describe good review practices**

When and Why To Analyze Architecture -1

Analyzing for system qualities early in the life cycle allows for a comparison of architectural options.

When building a system

- **Architecture is the earliest artifact where trade-offs are visible.**
- **Analysis should be done when deciding on architecture.**
- **The reality is that analysis is often done during damage control, later in the project.**

When and Why To Analyze Architecture -2

When acquiring a system

- **Architectural analysis is useful if the system will have a long lifetime within organization.**
- **Analysis provides a mechanism for understanding how the system will evolve.**
- **Analysis can also provide insight into other visual qualities.**

Properties Can Be Analyzed in Isolation

Different techniques can be used for different properties.

Analyzing for modifiability does not affect analyzing for performance.

Analysis is performed on an artifact and reveals properties.

The design method must take into account trade-offs and interactions between properties.

Qualities Are Too Vague for Analysis -1

Is the following system modifiable?

- **Background color of the user interface is changed merely by modifying a resource file.**
- **Dozens of components must be changed to accommodate a new data file format.**

A reasonable answer is

- **“yes” with respect to changing background color**
- **“no” with respect to changing file format**

Qualities Are Too Vague for Analysis -2

Qualities only have meaning within a context.

SAAM specifies context through scenarios.

SAAM is most useful in evaluating non-runtime qualities, although information generated can be used for other qualities.

Scenarios

A scenario is a brief description of a stakeholder's interaction with a system.

When creating scenarios, it is important to consider all stakeholders, especially
- **end users**
- **developers**
- **maintainers**
- **system administrators**

Steps of a SAAM Evaluation

Identify and assemble stakeholders

Develop and prioritize scenarios

Describe candidate architecture(s)

Classify scenarios as direct or indirect

Perform scenario evaluation

Reveal scenario interactions

Generate overall evaluation

Step 1: Identify and Assemble Stakeholders - 1

Stakeholder	Interest
Customer	Schedule and budget; usefulness of system; meeting customers' (or market's) expectations
End user	Functionality, usability
Developer	Clarity and completeness of architecture; high cohesion and limited coupling of parts; clear interaction mechanisms
Maintainer	Maintainability; ability to locate places of change

Step 1: Identify and Assemble Stakeholders - 2

Stakeholder	Interest
System administrator	Ease in finding sources of operational problems
Network administrator	Network performance, predictability
Integrator	Clarity and completeness of architecture; high cohesion and limited coupling of parts; clear interaction mechanisms

Step 1: Identify and Assemble Stakeholders - 3

Stakeholder	Interest
Tester	Integrated, consistent error-handling; limited component coupling; high component cohesion; conceptual integrity
Application builder (if product line architecture)	Architectural clarity, completeness; interaction mechanisms; simple tailoring mechanisms
Representative of the domain	Interoperability

Step 2: Stakeholders Develop and Prioritize Scenarios

Scenarios should be typical of the kinds of activities that the system must support:
- **functionality**
- **development activities**
- **change activities**

Scenarios also can be chosen to give insight into the system structure.

Scenarios should represent tasks relevant to all stakeholders.

Rule of thumb: 10-20 scenarios

Step 3: Describe Candidate Architectures -1

It is frequently necessary to elicit appropriate architectural descriptions.

Structures chosen to describe the architecture will depend on the type of qualities to be evaluated.
- **Static structures will be used to evaluate modification scenarios.**
- **Dynamic structures will be used to evaluate runtime qualities.**

Step 3: Describe Candidate Architectures -2

Typically, the architect is asked to bring the following documentation to an evaluation, as a starting point:
- **a description of the logical or module structure, with a clear definition of the responsibilities of each component**
- **a description of the process structure**
- **a data flow diagram showing how the modules interact at run time**
- **a description of the interconnection mechanism(s) used to distribute data and control among architectural components at run time**
- **a description, if needed, of the "uses" structure, showing how potential subsets will be realized**

Step 4: Classify Scenarios

There are two classes of scenarios.
- Direct scenarios are those that can be executed by the system without modification.
- Indirect scenarios are those that require modifications to the system.

The classification depends upon both the scenario and the architecture.

The classification should have gradations.
- indirect: How hard is the change? (SAAM elicits this information.)
- direct: How hard is it to execute? (SAAM does not consider this.)

Step 5: Perform Scenario Evaluation

For each indirect scenario
- identify the components, data connections, control connections, and interfaces that must be added, deleted, or modified
- estimate the difficulty of modification

Difficulty of modification is elicited from the architect and is based on the number of components to be modified and the effect of the modifications.

A monolithic system will score well on this step, but not on next step.

Step 6: Reveal Scenario Interactions

When multiple indirect scenarios affect the same components, this could indicate a problem.
- could be good, if scenarios are variants of each other
  - change background color to green
  - change background color to red
- could be bad, indicating a potentially poor separation of concerns
  - change background color to red
  - port system to a different platform

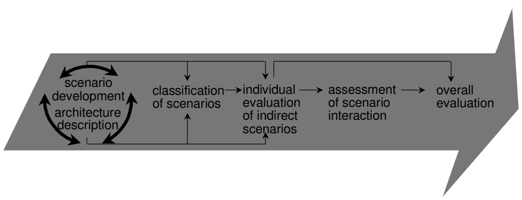
Step 7: Generate Overall Evaluation

Not all scenarios are equal.

The organization must determine which scenarios are most important.

Then the organization must decide as to whether the design is acceptable "as is" or if it must be modified.

Interaction of SAAM Steps



Two Examples of SAAM

Keyword in context (KWIC)
- **recall from lecture on architectural styles**
- **short overall demonstration of method**

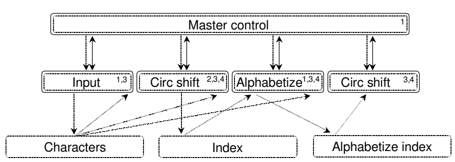
WRCS
- **demonstrates analysis method**
- **shows use of SAAM in reengineering context**

Example 1: KWIC

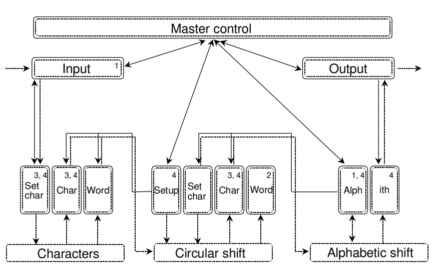
Two architectures: shared memory and abstract data type (ADT)

Four scenarios
1. **Operate in an incremental rather than a batch fashion. Program to accept one sentence at a time.**
2. **Program to eliminate entries beginning with "noise" words.**
3. **Change the internal representation of sentences (e.g., compressed or uncompressed).**
4. **Change the internal representation of the intermediate data structure (e.g., store shifted sentences or store pointers to them).**

Shared Memory KWIC



ADT KWIC



Evaluation of This Scenario Set

	Shared memory	Abstract data types
Scenario 1	0	0
Scenario 2	0	0
Scenario 3	-	+
Scenario 4	-	+
Contention	-	+

Example 2: SAAM Applied to Revision Control System

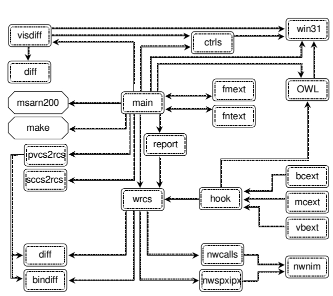
"WRCS" is a large, commercially-available revision control system.

No documented system architecture existed prior to the evaluation.

The purpose of the evaluation was to assess the impact of anticipated future changes.

Three iterations were required to develop a satisfactory representation, alternating between
- **development of scenarios**
- **representation of architecture**

Architectural Representation of WRCS



Scenarios Used in WRCS

User scenarios
- compare binary file representations
- configure the product's toolbar

Maintainer
- port to another operating system
- make minor modifications to the user interface

Administrator
- change access permissions for a project
- integrate with a new development environment

Scenario Classification

User scenarios
- compare binary file representations: indirect
- configure the product's toolbar: direct

Maintainer
- port to another operating system: indirect
- make minor modifications to the user interface: indirect

Administrator
- change access permissions for a project: direct
- integrate with a new development environment: indirect

Scenario Interactions

Each indirect scenario necessitated a change in some modules. This can be represented either tabularly or visually.

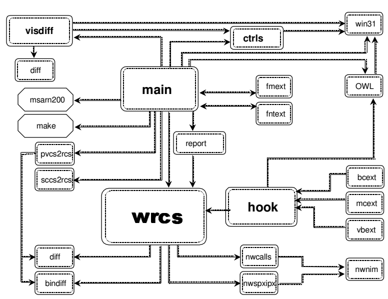
The number of scenarios that affected each module can be shown with a table or graphically, with a fish-eye view.

A fish-eye view uses size to represent areas of interest.

Scenario Interaction Table

Module	No. changes
main	4
wrcs	7
diff	1
bindiff	1
pvcs2rcs	1
sccs2rcs	1
nwcalls	1
nwspxipx	1
nwnlm	1
hook	4
report	1
visdiff	3
ctrls	2

Scenario Interaction Fish-Eye



Lessons Learned from WRCS

Granularity of architectural description

Interpretation of scenario interactions

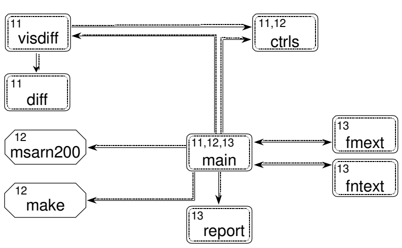
Proper Granularity of Architectural Description

The level of detail of architectural description is determined by the scenarios chosen.

The next slide shows what an architect thought was an appropriate level of detail.

Components are annotated with the numbers of indirect scenarios that affect them.

Original Representation of WRCS



What To Make of "Main" Scenario Interactions

Possibilities
- **Scenarios are all of the same class.**
- **Scenarios are of different classes, and "main" can be subdivided.**
- **Scenarios are of different classes and "main" cannot easily be subdivided.**

To differentiate between the last two options, see whether it is possible to divide the main module based on the three scenarios.

Lessons from SAAM -1

Direct scenarios provide a
- **first-order differentiation mechanism for competing architectures**
- **mechanism for eliciting and understanding structures of architectures (both static and dynamic)**

Lessons from SAAM -2

A SAAM evaluation produces
- **technical results: provides insight into system capabilities**
- **social results**
  - **forces some documentation of architecture**
  - **acts as communication vehicle among stakeholders**

Lessons from SAAM -3

It is important to have stakeholders present at evaluation meetings.
- **Stakeholders find it to be educational.**
- **Architectural evaluators may not have the experience to keep presenters "honest."**

SAAM and traditional architectural metrics
- **Coupling and cohesion metrics do not represent different patterns of use.**
- **High scenario interaction shows low cohesion.**
- **A scenario with widespread hits shows high coupling.**
- **Both are tied to the context of use.**
- **SAAM provides a means of sharpening the use of coupling and cohesion metrics.**

Architectural Reviews

Architectural reviews are a key step in architecture-based development. They include a method (such as SAAM), but the full practice includes more than that.

We will discuss
- cost/benefits of architectural reviews
- different architectural review techniques
- preconditions for an architectural review
- architectural review activities
- outputs from an architectural review

Costs of Architectural Reviews

There are three different types of costs of holding regular architectural reviews.
1. direct staff costs
2. organizational overhead
3. indirect staff costs

Examples of Direct Staff Costs - 1

AT&T
- 300 full-scale reviews done on projects of 700 staff-days or longer
- average cost per review: 70 staff days

Rational Software
- 30 reviews done on projects with at least 500 KSLOC each
- average cost per review: $50,000

Examples of Direct Staff Costs -2

SAAM evaluations
- **10-12 reviews done on projects ranging from 100 KSLOC to 1,000 KSLOC**
- **average cost per review: 14 to 20 staff-days**

Example of Organizational Overhead

Costs of establishing a corporate review unit
- **management overhead**
- **communication expenses**
- **staffing the unit**
- **relocating personnel to a central location**
- **training**

Example of Indirect Staff Costs

Using senior designers for evaluations instead of designing
- **loss of productivity (due to reassignment of superior designers)**
- **time spent training staff in review techniques**

Benefits of Architectural Reviews

Five different types of benefits result from holding architectural reviews.

1. **financial**
2. **forces preparation for review**
3. **early detection of problems**
4. **validation of requirements**
5. **improved architectures**

Financial Benefits of Reviews

AT&T estimated that each reviewed project saves 10% of total cost as a result of the review. Thus, a 70-staff-day review of projects of 700 staff-days pays for itself.

Consultants who perform architecture evaluations report 80% repeat business.

Forces Preparation for Review

Documentation/specifications must be provided, hence they must exist or be created.

Some reviews use standard questions, and the architect can prepare ahead to ensure that the architecture scores well.

Reviews make the criteria for evaluation explicit by prioritizing requirements or quality goals.

Early Detection of Problems

The problems that can be found by an architectural level inspection include

- unreasonable requirements
- performance problems
- problems associated with potential future modifications

The earlier in the life cycle that problems are found, the easier it is to fix them.

Validation of Requirements

Reviews put stakeholders in the same room with each other, often for the first time.

- uncovers conflicts and tradeoffs
- provides a forum for negotiated resolution of problems

It often results in the generation of new requirements or the clarification of existing requirements.

Improved Architectures

Development organizations anticipate types of questions raised at reviews and

- design architectures with questions in mind
- prepare documentation of the type needed at review
- give explicit consideration to qualities to be reviewed

Review Techniques

There are a variety of techniques for performing architectural reviews; each has a different cost and provides different information.

These techniques fall into one of two basic categories.

1. questioning techniques: applied to evaluate any aspect of an architecture for any given reason
2. measuring techniques: applied to answer questions about a specific quality

Questioning Techniques - 1

Scenario-based techniques

- describes a specific interaction between stakeholders and a system
- example: SAAM

Questionnaire-based techniques

- Some questions apply to all architectures (especially those in a single domain).
- Some questions ask about details of a specific architecture.
- Some ask about the development process.
  - "Is there a single architect?"
  - "How do you ensure conformance?"

Questioning Techniques - 2

Questionnaires are reused; they reflect more maturity or experience within a domain. Scenarios must be developed anew for each system.

Measuring Techniques - 1

Metrics: quantitative interpretations of observable measures
- complexity metrics identify areas where modifications may be difficult
- performance metrics suggest areas where bottlenecks may occur

Reviews tend to focus on
- results of metrics
- assumptions underlying the interpretation of the metrics (e.g., assumed event distribution)

Measuring Techniques - 2

Simulations, prototypes, experiments: domain-specific models of an architecture or performance model
- are expensive to create
- often exist as part of development anyway

May answer issues raised by questioning technique (e.g., "What evidence do you have that performance is adequate?")

Properties of Review Techniques

Review Technique		Generality	Level of Detail	Phase	What is Reviewed
Questioning techniques	Questionnaire	general	coarse	early	artifact, process
	Scenarios	system-specific	medium	middle	artifact
Measuring techniques	Metrics	general or domain-specific	fine	middle	artifact
	Prototype Simulation Experiment	domain-specific	varies	early	artifact

Review Process

An architectural review has three major elements.
1. preconditions: the set of necessary assets and conditions that must be in place before a review can be performed successfully
2. activities of the review itself
3. outputs: the report of review results

Review Preconditions

Understand the context of the review

Assemble the right people

Set organizational expectations and support

Prepare for review

Obtain representation of the architecture

Preconditions: Context - 1

Planned review
- part of normal development activity
- non-adversarial
- scheduled well in advance
- built into project's schedule and budget

Unplanned
- usually occurs when project is in trouble
- often devolves into finger-pointing
- can be painful for project already struggling

(Guess which one we recommend?)

Preconditions: Context -2

Timing of the review
- early "architecture discovery review"
  - is done after requirements are set, but before the architecture is firm
  - is used to understand implications of requirements on architecture
  - checks for requirements feasibility
  - prioritizes architectural goals
- full architectural review
  - is done when architectural documentation is available
  - is used to evaluate qualities of proposed architecture

Carnegie Mellon University
Software Engineering Institute

Preconditions: People

Representatives of project
- architect
- designers of major modules
- stakeholders' representatives

Review team members (ideally, a separate unit)
- at least one domain expert included
- someone to handle logistics (e.g., meeting rooms, supplies)
- librarian to organize documentation
- "apprentice reviewer" for growth

Carnegie Mellon University
Software Engineering Institute

Preconditions: Review Team -1

The review team must be assembled so that
- members are perceived as impartial, objective, and respected
- members devote their full-time effort to the review
- members are fluent in architecture and architectural issues
- it is located close to artifacts being examined

Preconditions: Review Team -2

The team must have access to

- **applicable domain knowledge (perhaps via consultants to the team)**
- **design documents**
- **review criteria**
- **support staff**

Preconditions: Organizational Expectations -1

Expectations must be set.

A *contract* between the review team and the sponsor of the review must be established.

- **Who will be told what upon completion?**
- **What will (not) be the review criteria?**
- **What/who will be made available to team?**
- **What follow-up is expected?**
- **How long will the review take?**

Preconditions: Organizational Expectations -2

Organizational culture and support must be considered.

- **Are reviews part of the standard project life cycle?**
- **Is serving on a review team considered good for one's career?**
- **Will the organization allow its superior designers time off from their projects to serve on review teams?**
- **Is there a standing review organization (even though members may rotate on and off)?**

Preconditions: Read-Ahead Material

For project being reviewed
- **questionnaires**
- **scenarios (These are created as part of the review, but time should be allowed for project members to digest the information.)**

For review team
- **description of architecture**
- **rationale for architecture**

Ideally, material will address the review questions. (Passing this test because the questions were known in advance is not cheating, but good engineering!)

Preconditions: Ranked Quality and Behavioral Requirements

Plan to examine three to five quality and behavioral requirements.

A particular representation medium (language) for the architecture is not necessary. As long as the information can be extracted (using project members as necessary), the form of the representation is not important.

Review Activities - 1

Evaluate
- **“Run” the scenarios and ask the questions, or answer the items in the checklist.**
- **Perform the experiments.**
- **Execute the prototypes or simulations.**

Record
- **critical issues highlighted in the review**
- **comments of project members on these issues**

Review Activities -2

Rank the issues
- **project-threatening**
- **major**
- **minor**

Each issue raised should be addressed in terms of
- **changing the architecture**
- **relaxing the requirements**

Review Activities -3

Warning signs
- **architecture forced to match organization**
- **top-level components number over 25**
- **one requirement drives entire design**
- **architecture depends on alternatives in the operating system**
- **choice of software components is dictated by hardware personnel**
- **redundancy not needed for reliability**
- **design is exception driven**
- **no identifiable architect**

Output from Review

Set of ranked issues
- **supporting data**
- **contained in formal report**
- **feedback to project**

Enhanced system documentation

Set of scenarios for future use

Identification of potentially reusable components

Estimation of costs and benefits

Lecture Summary

Have a formal review with external reviewers.

Time the review to best advantage.

Hold a planned (rather than unplanned) review.

Choose the appropriate review technique.

Create a review contract.

Limit the number of qualities to be reviewed.

Insist on having a system architect present.

Discussion Questions -1

1. **How are architectures currently reviewed in your enterprise? What are the good points of that practice? What are ways in which it could be improved?**
2. **AT&T's data suggest that an architecture review saves a typical project about 10 percent. This is an organization that builds systems in a fairly mature domain, using questionnaires that the architects have access to beforehand. Why would a review save anything (after all, every project knows what is going to be asked of it)? Discuss this apparent paradox.**

Discussion Questions -2

3. **How much do you think institutionalized architecture reviews would save in your organization? Why?**
4. **Try to write a checklist for an architecture review for a system in your organization's domain. Write one for artifact review and one for process review.**

Discussion Questions -3

5. The distinction between direct and indirect scenarios is a coarse way to measure architectural support for a feature. As we have seen, a direct scenario may be so cumbersome as to be effectively indirect. On the other hand, an indirect scenario may be trivially supported by changing a parameter and recompiling. Propose a finer-grained way to categorize scenarios that takes these considerations into account.

6. Produce a set of scenarios for an architecture in your organization.

Discussion Questions -4

7. Attendees at an architecture review naturally bring their own agendas to the table, and these agendas often show up in the scenarios they propose. Consider this actual scenario from a recent architecture evaluation using SAAM: Produce a system for the least cost and shortest time possible. What quality attribute of the architecture were they trying to measure? What else were they trying to address? How would you handle this scenario if you were running the evaluation?