Architecture-Related Technology

**Problem**
- Prevailing technology and technology trends can both enable and be hostile to sound architecture practices
- Architecture practices are often labor intensive and error prone
  - Automated support can help.

**Approach**
- Scrutinize technology and technology trends through the lens of architecture-centric development and provide guidance and support
  - SOA, from a quality attribute point of view
  - Service Level Agreements
- Identify technology gaps related to architecture practices and provide guidance and build prototype tools
  - ArchE, an architectural design assistant
Evaluating a Service Oriented Architecture (SOA)

Improved interoperability is a prominent benefit of SOA but other important quality attributes can be impacted.

Our goal:
- Determine how an SOA approach affects the application of the Architecture Tradeoff Analysis Method (ATAM).
- "Put on quality attribute glasses."
- Provide practical information for evaluating a system that uses an SOA approach

Our approach:
- Identify common SOA design decisions
- Discuss how these variants impact system quality attributes
- Illustrate ideas by partially evaluating a publicly available system that uses an SOA approach

Service Oriented Architectural Style

[Diagram showing the architectural style with Service User(s), Service Registry, Service Provider(s), and Service broker]
Service level agreements (SLAs)

Survey the state-of-practice for ensuring quality of service by contract in an SOA context

- Which qualities can be expressed in a service level agreement?
- Describe the current state of the practice:
  - formalizing SLA statements
  - monitoring via the infrastructure
  - visualizing violations
  - standard availability and maturity

Offer guidelines to organizations that work with external services providers

Technical report will be available later this summer.

ArchE

ArchE is an architectural design assistant developed by the SEI to help architects explore architectural designs driven by quality attributes

ArchE shows the architect proposals for improving the current architecture, and allows him to decide on the best alternative

ArchE relies on the outputs of reasoning frameworks

Versions:
- ArchE 2.1 (released July 2007)
- ArchE 3.0 (released April 2008)
ArchE Supports Multiple Quality Attributes

A Reasoning Framework encapsulates the knowledge needed to enable ArchE (or a designer) to reason about a specific quality attribute.

Allows for extension of quality attribute knowledge within ArchE by plugging in a new reasoning framework.

Reduces interactions (dependencies) among quality attributes.

One of the research questions is the extent to which interactions among quality attributes reasoning frameworks can be reduced.
Reasoning Frameworks within ArchE – 1

A reasoning framework within ArchE

1. Translates from architecture description to quality attribute model – we call this “Interpretation”
2. Evaluates quality attribute scenarios in terms of the model – we call this “Evaluation”
3. Proposes tactics to improve architecture.

Two inputs into a reasoning framework within ArchE

1. Current architecture
2. Relevant quality attribute scenarios

Outputs:

1. Evaluation of current architecture with respect to the quality attribute scenarios
2. List of potential tactics to modify the architecture, if at least one scenario is currently unmet

Reasoning Frameworks within ArchE – 2

Requires a clear definition of the architectural elements, relations, and properties that can influence a quality attribute.

- The “Interpretation” extracts this information from an architecture and creates a quality attribute model from it

Requires the existence of a “Formula” to do calculations with the model to provide some information about the fulfillment of the quality attribute

- That is what the “Evaluation” does

Requires a clear definition of possible changes to the architecture to make it better fulfill the quality attribute

- This is what “Tactics” are for
Example: Performance Reasoning Framework – 1

The performance of an architecture depends on the assignment of functionality to tasks. One of the typical measures for performance is Latency – the time it takes to finish a task.

Therefore the following information must be available:

- Performance scenarios
  - Have period and deadline
- Scenario to responsibility assignments
- Responsibility properties
  - Execution time
  - Is the responsibility reentrant?

Example: Performance Reasoning Framework – 2

Assigning each responsibility an execution time is job of architect. There is no way ArchE can know initial values.

Constructing model from architecture description is as follows:

- Each performance scenario becomes a task
- The period specified for the scenario becomes the period of this task
- The response measure becomes the task deadline
- Each responsibility has an execution time
- Responsibilities assigned to a scenario become responsibilities assigned to the task
- Responsibilities not assigned to a performance scenario are assigned to an additional, low priority task (background task)
- Shared responsibilities become shared resources
- A shared resource has an execution time for each task that uses this resource
Example: Performance Reasoning Framework – 3

Architecture

Period 100
Scenario 1
Time 20
Resp. A

Period 140
Scenario 2
Time 20
Resp. B

Period 160
Scenario 3

Performance Model

Task 1
Time 100
20ms

Task 2
Time 30
20ms

Task 3
Time 30
20ms

Shared Resource A

Shared Resource B

Interpretation

ArchE RF Interface (ArchE 3.0)

The more reasoning frameworks ArchE has available, the broader its capabilities will be. The ArchE-RF Interface is a collaborative infrastructure to help third parties to contribute their own reasoning frameworks to ArchE.
ArchE Main Loop

ArchE runs a search algorithm whenever the user makes a change. Each reasoning framework responds to ArchE interaction commands, implementing:

- **Self description** (manifesto)
- **Create initial design**
- **Analyze**
- **Suggest tactics**
- **Describe tactics**
- **Apply tactics**

**Development Steps**

The ArchE RF-Interface provides hooks and default classes that keep you from knowing the internals of the communications between ArchE and the reasoning frameworks. You can focus on implementation tasks such as definition of scenarios, analysis of the architecture, transformation of the architecture via tactics, etc.
ArchE – Current RF Examples

We have implemented reasoning frameworks for modifiability and real-time performance

Resources

Downloading ArchE:

- http://www.sei.cmu.edu/architecture/arche.html
- http://www.sei.cmu.edu/architecture/arche_research.html

Related publications:

- Using ArchE in the Classroom: One Experience (CMU/SEI-2007-TN-001). McGregor, John D.; Bachmann, Felix; Bass, Len; Bianco, Philip; Klein, Mark