CS554
Designs for Software and Systems

Architecture Evaluation (ATAM)

Lecture Given by

Ho-Jin Choi

Department of Computer Science

KAIST

Today's Agenda

- Architecture Description Languages
  - > Team assignment for further investigation
  - > Oral presentation – OP5
- Project 2
  - > ATAM – Architecture Tradeoff Analysis Method
  - > Oral presentations – OP4, OP6

Architecture Description Languages

- Architecture Description Languages (ADLs)
  - > Describe architectures precisely
  - > About a dozen ADLs
  - > Usually with supporting toolkits
- Basic ADL Requirements
  - > ADL's must provide support for:
    - » Composition
    - » Abstraction/Closure
    - » Reusability
    - » Configuration
    - » Heterogeneity
    - » Analysis
    - » First-Class Connectors

Architecture Description Languages

- A number of architecture description languages (ADLs) have been developed in the SE community
- Each language and supporting tool set
  - > Emphasizes different aspect of arch
  - > Is good for some things; bad for others
- Some examples
  - > Rapide – Events with simulation and animation
  - > UniCon – Emphasizing heterogeneity and compilation
  - > Wright – Formal specification of connector interactions
  - > Aesop – Style-specific arch design languages
  - > ArTek – TeKnowledge description language
  - > SADL – SRI language emphasizing refinement
  - > Meta-H – Architecture description for avionics domain
  - > C-2 – Architecture style using implicit invocation

Key Features of Modern ADLs

- System structure is defined separately from individual components
  - > Parts are "context independent"
  - > Supports hierarchical design
- Components have multiple interfaces
  - > Different points of interaction with their environment
- New kinds of connectors can be defined
  - > Need not be realizable directly in a primitive of an implementation language
  - > Have rich semantics
- Can express/analyze extra-functional properties
  - > Performance, reliability, etc.
- Support for architectural styles
  - > Reusable architectural patterns

Oral Presentation – OP5

- The task - Teamwork
  - > Choose one ADL from the list
    - » Teams must make different choices
  - > Survey literature and investigate
  - > Present in the class about the ADL of your choice
    - » 30 min presentation each team
- Schedule
  - > Two teams to present on Thursday, 19 November
  - > Remaining two teams to present on Thursday, 26 November

Project 2 (Recap.)

- Design of Software Architecture
  - Prioritized utility tree
    - Scenarios
  - Proposed architecture
    - Architectural views
  - Analysis of the architecture
    - Applying to ranked utility tree
  - Architectural evaluation
    - Risks, sensitivity points, tradeoffs
    - Key architectural decisions among alternatives

Oral Presentations – OP4, OP6

- Interim presentations for project P2
  - > In-class discussion to get feedback
  - > Not graded
  - > Present in the class
    - » 15 min presentation each team
- Schedule
  - > OP4 – Thursday, 12 November
  - > OP6 – Thursday, 3 December

Architecture Evaluation

- We identify architectural patterns or styles
  - > Architectural styles, idioms, families, frameworks, patterns
- Every architect should have a standard set of architectural styles in his/her repertoire
  - > It is important to understand the essential aspects of each style
    - » When to and when not to use them
  - > Examples
    - » Pipe and filters, objects, event-based systems, blackboards, interpreters, layered systems
- Choice of style can make a big difference in the properties of a system
  - > Analysis of the differences can lead to principled choices among alternatives

Architecture Evaluation

- **Why evaluate an architectural design?**
  - > All design involves tradeoffs.
  - > A software architecture is typically the first project artifact that embodies significant design decisions.
  - > We want a method that ensures that
    - » the right questions are asked . . . *early*
    - » risks and sensitivity points within the architecture for various attributes are identified
    - » tradeoff points are explicitly identified

Architecture Evaluation

- Determining whether an architecture satisfies its requirements often involves
  - > Being very explicit about what the requirements really are
  - > Understanding where one has to make tradeoffs between different designs
  - > Applying formal analysis where possible to determine the consequences of an architectural choice
  - > Mediating between desires of different stakeholders
- These goals can be achieved by an architectural evaluation process
- We will look at an example of one such process:
  - > ATAM

ATAM

- **ATAM**
  - > Architecture Tradeoff Analysis Method
    - » An architecture evaluation method
  - > Focuses on multiple quality attributes
  - > Illuminates points in the architecture where:
    - » Sensitivity to various attributes exists
    - » Quality attribute tradeoffs occur
  - > Generates a framework for ongoing quantitative/qualitative analysis
  - > Utilizes an architecture's vested stakeholders as authorities on the quality attribute goals

The ATAM Process

1) Collect system usage scenarios from various stakeholders

2) Collect requirements/constraints/environment

   » These are the requirements for which analyses will be performed

3) Describe architectural designs

   » Describe multiple, competing architectural options

4) Perform attribute-specific analyses

   » Analyze properties of each architecture option in isolation

5) Identify sensitivities

   » Determine the sensitivity of the various attributes to the available architectural design options

6) Identify tradeoffs

   » Determine which architectural elements are sensitive to multiple attributes (e.g. # of servers affects both the cost of the system and its overall reliability)

7) Repeat…

Factors Influencing Choice of Architecture

- Structure of the development organization
  - > May influence how project can partitioned
- Enterprise goals of the development organization
  - > Influences product-lines; use of standards
- Customer requirements
  - > Determines properties that system must have
- Architect’s experience
  - > Determines what are risky and safe architectures
- Technical environment
  - > Implementor skills; development environment and platform; legacy code

ATAM and Risks

- The point of an ATAM analysis is
  - > Not to provide precise analyses
  - > But to discover areas of high risk in the architecture
- We want to find trends
  - > Correlations between architectural parameters and measurable properties
- These areas can then be made the focus of risk mitigation activities
  - > E.g., further design, further analysis, prototyping

ATAM Benefits

- We have observed a number of benefits to performing ATAM analyses:
  - > Clarified requirements
  - > Improved architecture documentation
  - > Documented basis for architectural decisions
  - > Early problem identification
  - > Increased stakeholder communication

- And, obviously, improved architectures.

Scenarios

- ATAM uses use cases and scenarios
  > To articulate the specific quality attributes that are goals for the architecture
- Use cases and scenarios are short descriptions of putting the system:
  > through a particular use
  > through a particular change
  > through a particular attack
  > through a particular failure

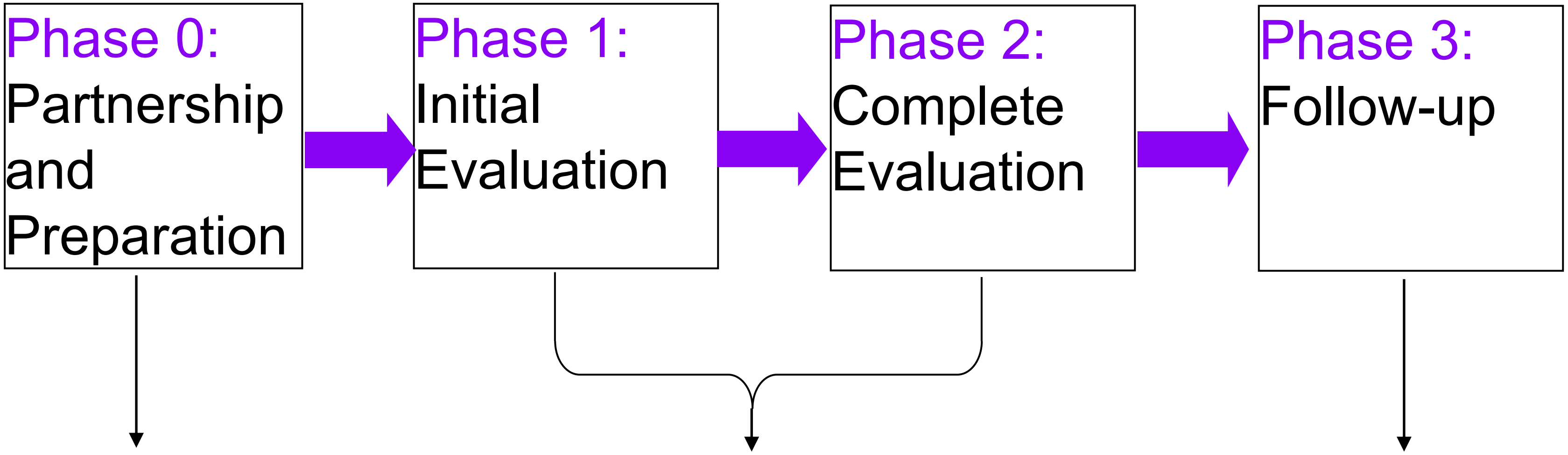
. . . from the perspective of one or more of the system's stakeholders.

Attribute Taxonomies

- To better understand each quality attribute, we need to develop attribute taxonomies.
  - > See Appendix of [KKC00] for some attributes
- For each attribute, the taxonomy describes
  - > The stimuli of interest
  - > The responses
  - > The quality attribute architectural parameters
- These provide a standard set of investigation paths to structure an analysis.

ATAM Phases

- **ATAM evaluations are conducted in four phases.**



Phase 0: Partnership and Preparation

Phase 1: Initial Evaluation

Phase 2: Complete Evaluation

Phase 3: Follow-up

Duration: varies
Meeting: primarily phone, email

Duration: 1.5 - 2 days each for Phase 1 and Phase 2
Meeting: typically conducted at customer site

Duration: varies
Meeting: primarily phone, email

ATAM Phase 0

- Phase 0: This phase precedes the technical evaluation.
  - > The customer and a subset of the evaluation team exchange understanding about the method and the system whose architecture is to be evaluated.
  - > An agreement to perform the evaluation is worked out.
  - > A core evaluation team is fielded.

ATAM Phase 1

- Phase 1: involves a small group of predominantly technically-oriented stakeholders

- Phase 1 is
  - > Architecture centric
  - > Focused on eliciting detailed architectural information and analyzing it
  - > Top down analysis

ATAM Phase 1 Steps

1. Present the ATAM
2. Present business drivers
3. Present architecture
4. Identify architectural approaches
5. Generate quality attribute utility tree
6. Analyze architectural approaches

Phase 1

7. Brainstorm and prioritize scenarios
8. Analyze architectural approaches
9. Present results

1. Present the ATAM

- The evaluation team presents an overview of the ATAM including:
  - ATAM steps in brief
  - Techniques
    - Utility tree generation
    - Architecture elicitation and analysis
    - Scenario brainstorming/mapping
  - Outputs
    - Architectural approaches
    - Utility tree and scenarios
    - Risks and “non-risks”
    - Sensitivity points and tradeoffs

2. Present Business Drivers

- ATAM customer representative describes the system’s business drivers including:
  - > Business context for the system
  - > High-level functional requirements
  - > High-level quality attribute requirements
    - » Architectural drivers – quality attributes that “shape” the architecture
    - » Critical requirements – quality attributes most central to the system’s success

3. Present Architecture

- Architect presents an overview of the architecture including:
  - > Technical constraints such as an OS, hardware, or middleware prescribed for use
  - > Other systems with which the system must interact
  - > Architectural approaches used to address quality attribute requirements
- Evaluation team begins probing for and capturing risks.

4. Identify Architectural Approaches

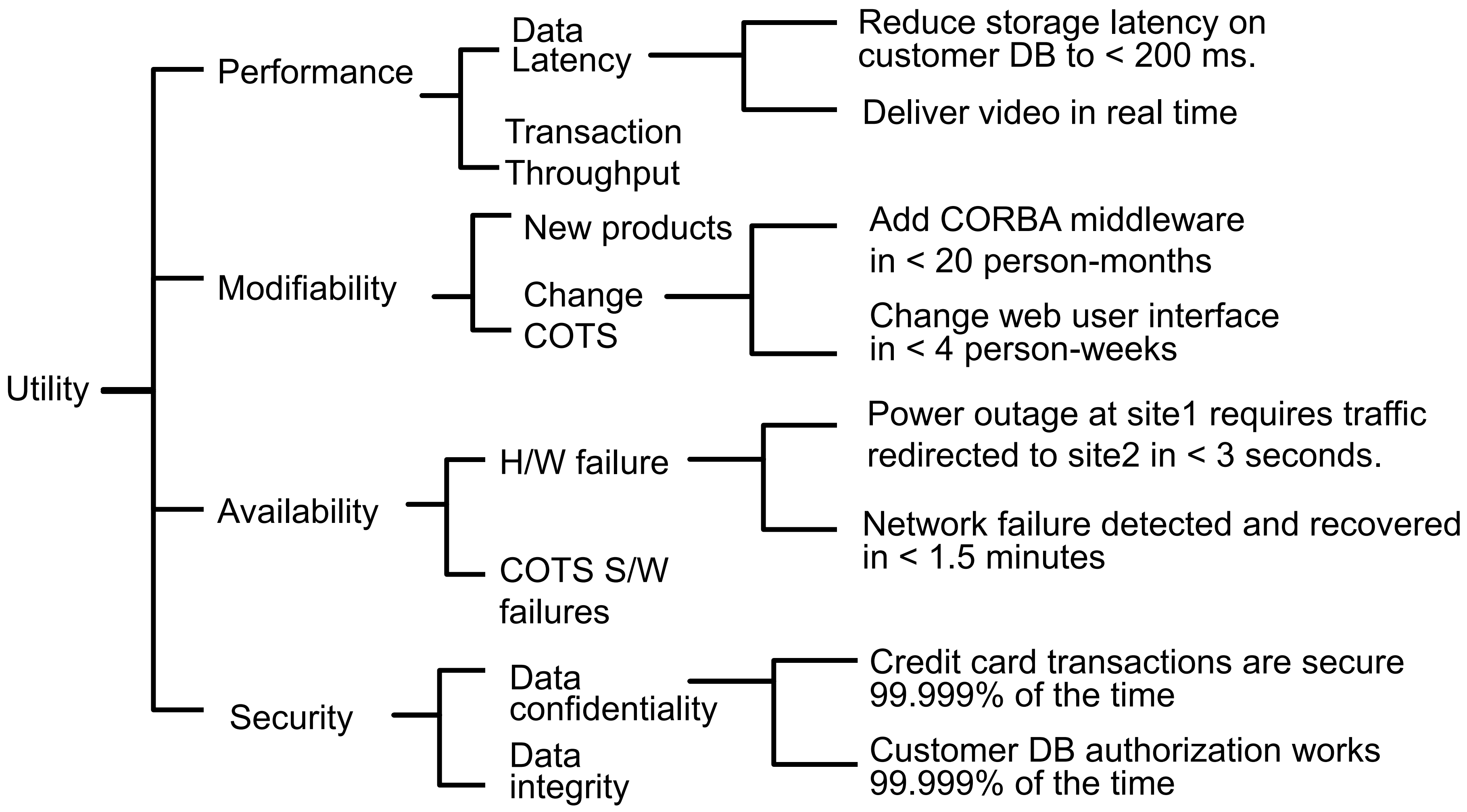
- The evaluators will begin to identify places in the architecture that are key to realizing quality attribute goals.
- Identify predominant architectural approaches such as:
  - > Client-server
  - > 3-tier
  - > Watchdog
  - > Publish-subscribe
  - > Redundant hardware

5. Generate Quality Attribute Utility Tree

- Identify, prioritize, and refine the most important quality attribute goals by building a utility tree.
  - > A utility tree is a top-down vehicle for characterizing the “driving” attribute-specific requirements.
  - > The most important quality goals are the high-level nodes (typically performance, modifiability, security, and availability).
  - > Scenarios are the leaves of the utility tree.

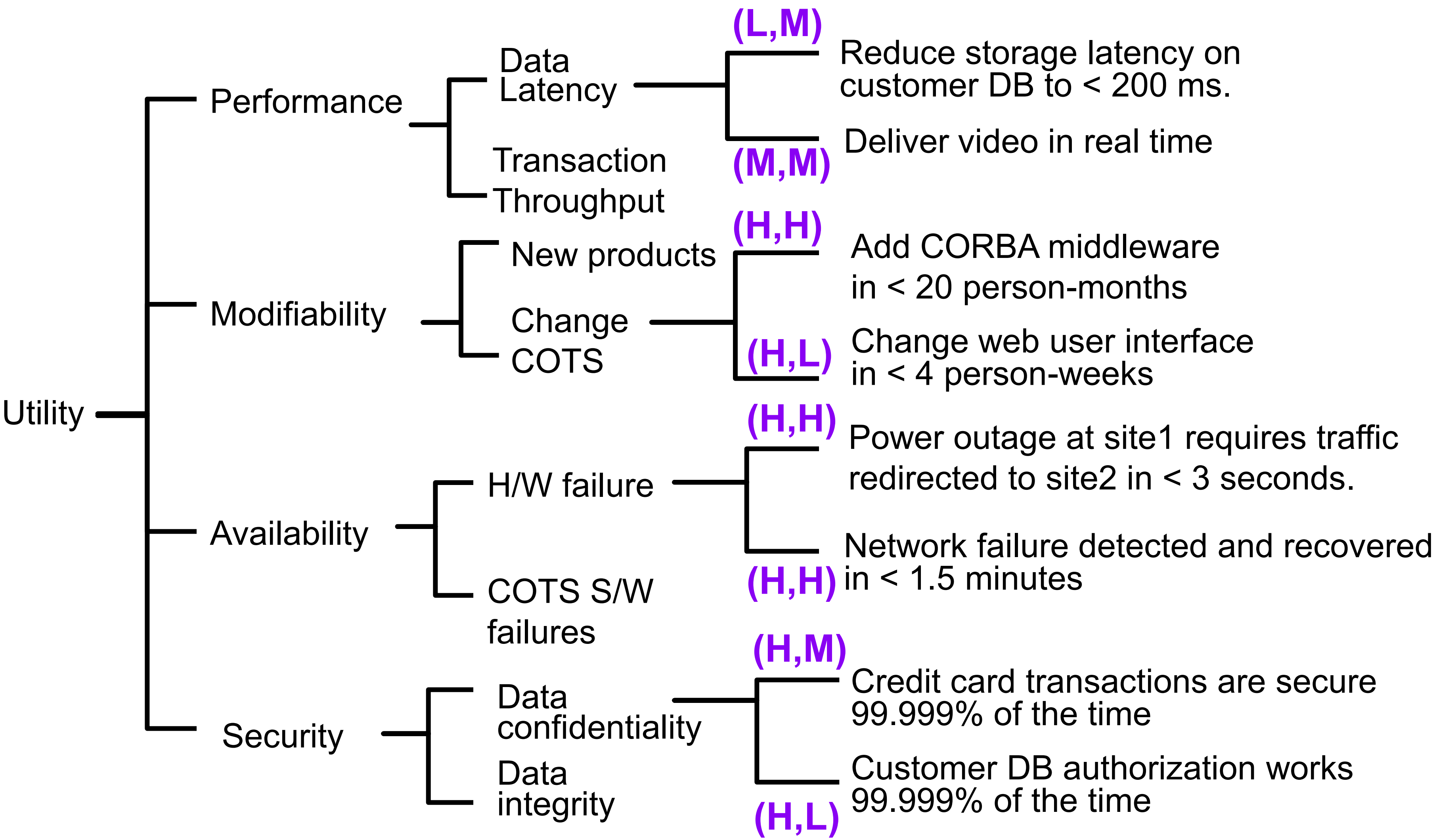
- Output: a characterization and a prioritization of specific quality attribute requirements.

Utility Tree Construction



- Utility
  - Performance
    - Data Latency
      - Reduce storage latency on customer DB to < 200 ms.
      - Deliver video in real time
    - Transaction Throughput
  - Modifiability
    - New products
    - Change COTS
      - Add CORBA middleware in < 20 person-months
      - Change web user interface in < 4 person-weeks
  - Availability
    - H/W failure
      - Power outage at site1 requires traffic redirected to site2 in < 3 seconds.
      - Network failure detected and recovered in < 1.5 minutes
    - COTS S/W failures
  - Security
    - Data confidentiality
      - Credit card transactions are secure 99.999% of the time
      - Customer DB authorization works 99.999% of the time
    - Data integrity

Utility Tree Construction



- Utility
  - Performance
    - Data Latency
      - (L,M) Reduce storage latency on customer DB to < 200 ms.
      - (M,M) Deliver video in real time
    - Transaction Throughput
  - Modifiability
    - New products
    - Change COTS
      - (H,H) Add CORBA middleware in < 20 person-months
      - (H,L) Change web user interface in < 4 person-weeks
  - Availability
    - H/W failure
      - (H,H) Power outage at site1 requires traffic redirected to site2 in < 3 seconds.
      - (H,H) Network failure detected and recovered in < 1.5 minutes
    - COTS S/W failures
  - Security
    - Data confidentiality
      - (H,M) Credit card transactions are secure 99.999% of the time
      - (H,L) Customer DB authorization works 99.999% of the time
    - Data integrity

Scenarios

- Scenarios are used to
  - > Represent stakeholders' interests
  - > Understand quality attribute requirements

- Scenarios should cover a range of
  - > Anticipated uses of (use case scenarios),
  - > Anticipated changes to (growth scenarios), or
  - > Unanticipated stresses (exploratory scenarios) to the system.

- A good scenario makes clear what the stimulus is that causes it and what responses are of interest.

Example Scenarios

- Use case scenario
  - > Remote user requests a database report via the Web during peak period and receives it within 5 seconds.
- Growth scenario
  - > Add a new data server to reduce latency in scenario 1 to 2.5 seconds within 1 person-week.
- Exploratory scenario
  - > Half of the servers go down during normal operation without affecting overall system availability.
- Scenarios should be as specific as possible.

Stimulus, *Response*, Environment

- Use case scenario
  - > Remote user requests a database report via the Web during peak period and *receives it within* 5 *seconds*.

- Growth scenario
  - > Add a new data server to reduce latency in scenario 1 to 2.5 seconds *within* 1 *person-week*.

- Exploratory scenario
  - > Half of the servers go down during normal operation *without affecting overall system availability*.

6. Analyze Architectural Approaches

- Evaluation team probes architectural approaches from the point of view of specific quality attributes to identify risks.
  - > Identify the architectural approaches
  - > Ask quality attribute specific questions for highest priority scenarios
  - > Identify and record risks and non-risks, sensitivity points and tradeoffs

Example Approach Elicitation

- Scenario: Detect and recover from HW failure of main switch
- Stimulus: A CPU fails
- Response: 0.999999 availability of switch

Architectural Approaches:	R	S	T
> Backup CPU(s)	x	x	
> Backup Data Channel	x	x	x
> Watchdog		x	
> Heartbeat		x	
> Failover Rerouting	x	x	

Example Approach Elicitation (cont'd)

- **Analysis:**
  - > Ensures no common mode failure by using different HW and OS
  - > Worst-case rollover is accomplished in 3 seconds
  - > Guaranteed to detect failure with 1 second
  - > Watchdog is simple and proven reliable

Quality Attribute Questions

- Quality attribute questions probe architectural decisions that bear on quality attribute requirements.
- These come from codified knowledge, such as ABASs.

- Performance
  - > How are priorities assigned to processes?
  - > What are the message arrival rates?

- Modifiability
  - > Are there any places where layers/facades are circumvented?
  - > What components rely on detailed knowledge of message formats?

Risks, Tradoffs, Sensitivities, Non-Risks

- A Risk is a potentially problematic architectural decision.
- Non-Risks are good architectural decisions that are frequently implicit in the architecture.
- A Sensitivity Point is a property of one or more components (and/or component relationships) that is critical for achieving a particular quality attribute response.
- A Tradeoff Point is a property that affects more than one attribute and is a sensitivity point for more than one attribute.

Risks and Tradeoffs

- Example Risk:
  - > "Rules for writing business logic modules in the second tier of your 3-tier architecture are not clearly articulated. This could result in replication of functionality thereby compromising modifiability of the third tier."

- Example Tradeoff:
  - > "Changing the level of encryption could have a significant impact on both security and performance."

Sensitivities and Non-Risks

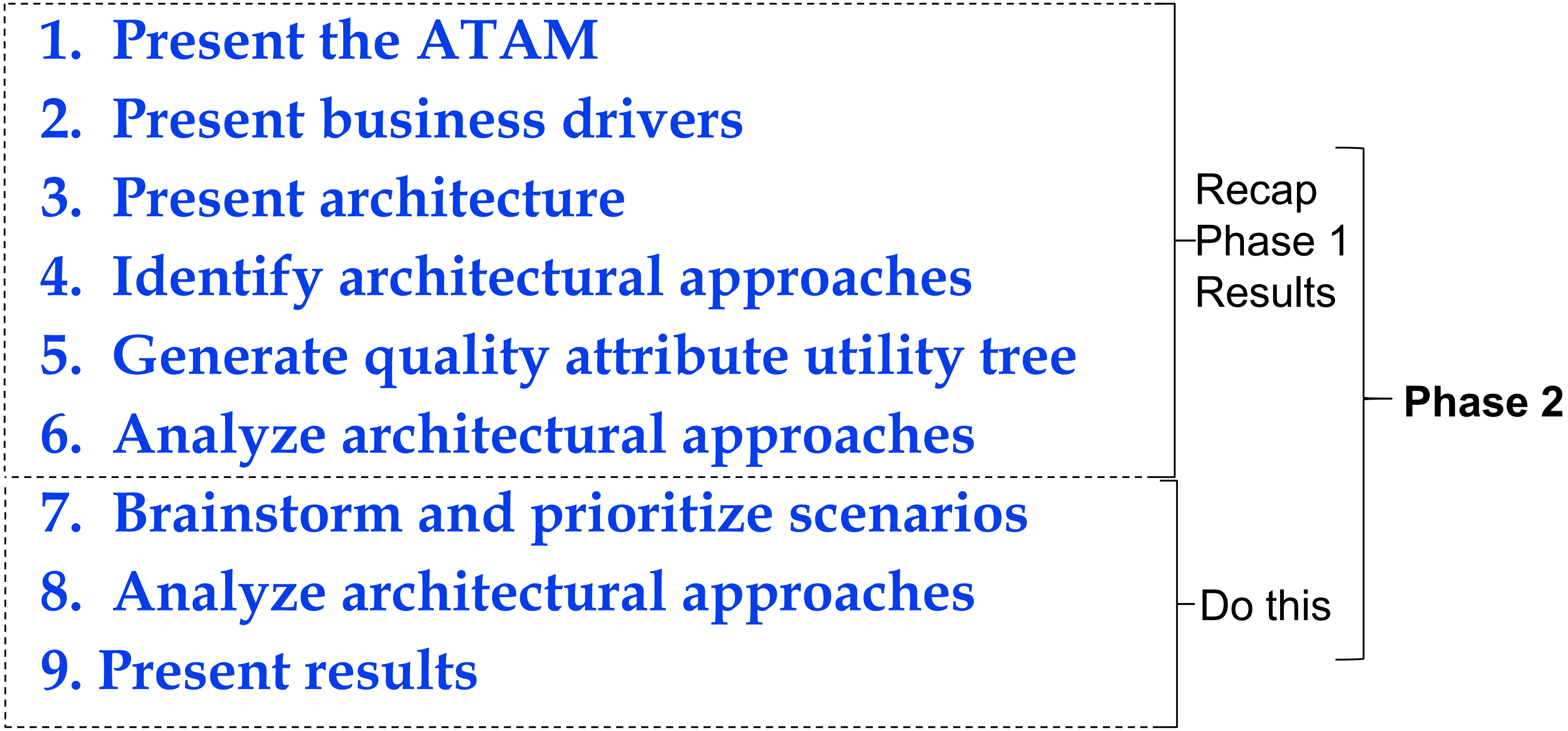
- Example Sensitivity:
  - > "The average number of person-days of effort it takes to maintain a system might be sensitive to the degree of encapsulation of its communication protocols and file formats."
- Example Non-Risk:
  - > "Assuming message arrival rates of once per second, a processing time of less than 30 ms, and the existence of one higher priority process, a 1 second soft deadline seems reasonable."

ATAM Phase 2

- Phase 2: involves a larger group of stakeholders

- Phase 2 is
  - > Stakeholder centric
  - > Focused on eliciting diverse stakeholder points of view and on verification of the Phase 1 results

ATAM Phase 2 Steps



1. Present the ATAM
2. Present business drivers
3. Present architecture
4. Identify architectural approaches
5. Generate quality attribute utility tree
6. Analyze architectural approaches
7. Brainstorm and prioritize scenarios
8. Analyze architectural approaches
9. Present results

7. Brainstorm and Prioritize Scenarios

- Stakeholders generate scenarios using a facilitated brainstorming process.
  - > Scenarios at the leaves of the utility tree serve as examples to facilitate the step.

- In phase 2, each stakeholder is allocated a number of votes roughly equal to 0.3 x #scenarios.

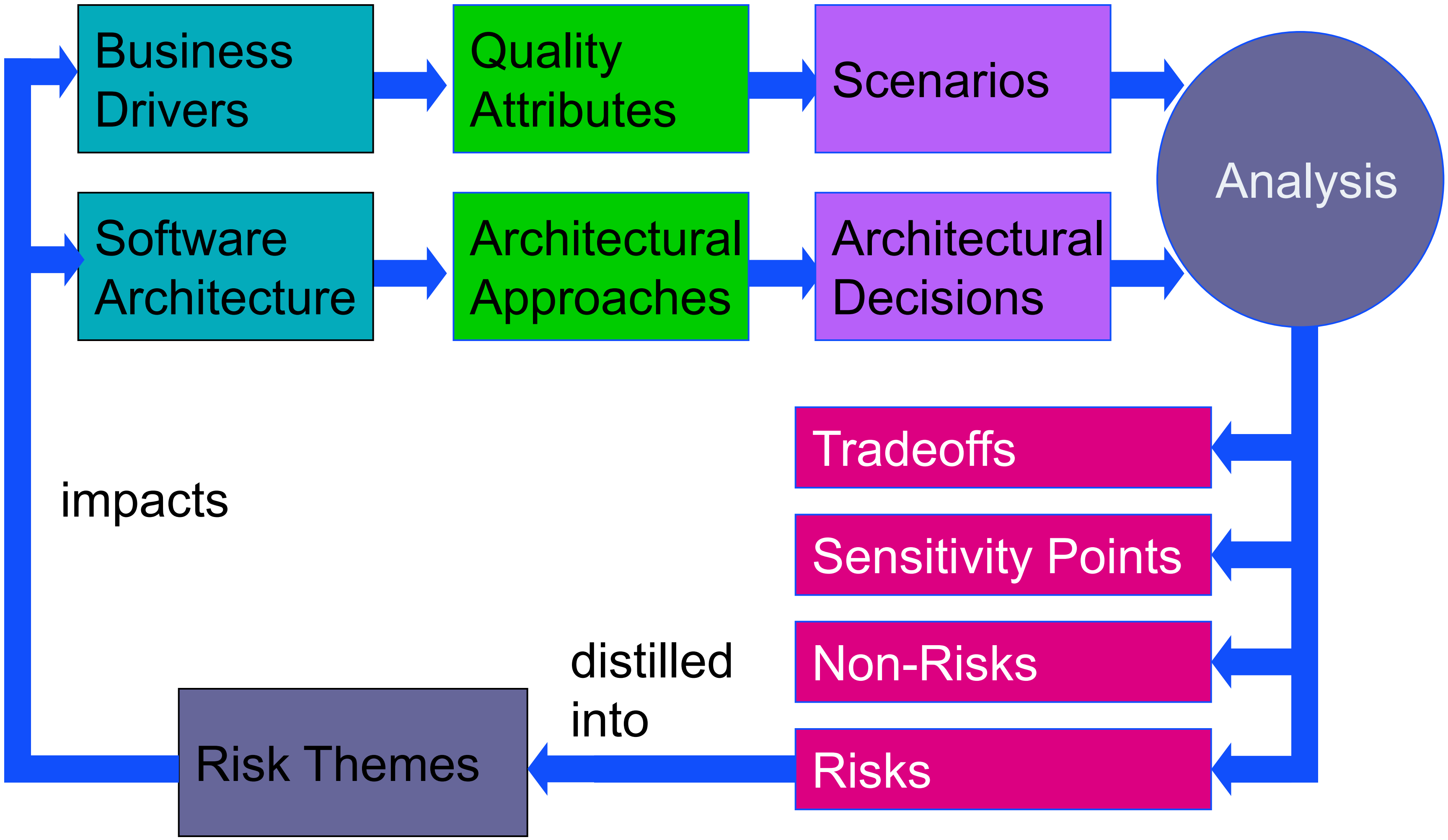
8. Analyze Architectural Approaches

- Identify the architectural approaches impacted by the scenarios generated in the previous step.
  - > This step continues the analysis started in step 6 using the new scenarios.
  - > Continue identifying risks and non-risks.
  - > Continue annotating architectural information.

9. Present Results

- Recapitulate all the steps of the ATAM and present the ATAM outputs:
  - > Architectural approaches
  - > Utility tree
  - > Scenarios
  - > Risks and non-risks
  - > Sensitivity points and tradeoffs
  - > Risk themes

Conceptual Flow of ATAM



ATAM Phase 3

- Phase 3: primarily involves producing a final report for the customer as well as reflecting upon the quality of the evaluation and the ATAM materials.
  - > If no final report was called for, then report writing is omitted.

The Final Report

- The evaluation team will typically create the final report which includes:
  - > Executive summary
  - > Description of ATAM
  - > Description of business goals and architecture
  - > List of Phase 1 and Phase 2 scenarios and utility tree
  - > Phase 1 and Phase 2 analysis: architectural approaches, decisions, risks, sensitivities, tradeoffs, and non-risks
  - > Risk Themes
  - > Next Steps

Rules of Engagement

- To complete everything and get the most from the evaluation we will need some rules of engagement:
  - > Stakeholder participation is essential.
  - > They should feel free to ask questions at anytime.
  - > Discussion is good, but the facilitator may need to cut some discussions short in the interest of time.
  - > Stakeholders should stay focused and limit side discussions.

Summary

- ATAM is a method for evaluating an architecture with respect to multiple quality attributes.
- It is an effective strategy for discovering the consequences of architectural decisions.
- The ATAM:
  - > Can be done early; can be done on legacy systems
  - > Is inexpensive
  - > Builds stakeholder confidence and buy-in
- The key to the method is looking for trends, not in making precise analyses.

Summary (cont'd)

The ATAM relies critically on

- Appropriate preparation by the customer
- Clearly-articulated quality attribute requirements
- Active stakeholder participation
- Active participation by the architect
- Familiarity with architectural approaches/styles and analytic models

Q & A

- **Questions?**