Requirements and Architecture Evaluation

Parinya Ekparinya

Parinya.Ekparinya@gmail.com

Software Architecture and Design

2021 Semester 1

Outline

- Architecturally Significant Requirement (ASR)
- Utility Tree
- Architecture Tradeoff Analysis Method (ATAM)
  - Participants
  - Output
  - Phases
  - 9 Evaluation steps
- Lightweight Architecture Evaluation

Architecturally Significant Requirement (ASR)

Architecturally Significant Requirement (ASR)

- ASR is a requirement that will have a profound effect on the architecture—that is, the architecture might well be dramatically different in the absence of such a requirement.
- You cannot hope to design a successful architecture if you do not know the ASRs.
- ASRs often, but not always, take the form of quality attribute (QA) requirements— the performance, security, modifiability, availability, usability, and so forth, that the architecture must provide to the system.
- An obvious location to look for candidate ASRs is in the requirements documents or in user stories. After all, we are looking for requirements, and requirements should be in requirements documents.

Gathering ASRs from Requirements Documents

- The architect must begin while the requirements are still in flux. The QA requirements are quite likely to be up in the air when the architect starts work.
- Many projects don't create or maintain the kind of requirements document that professors in software engineering classes or authors of software engineering books love to prescribe.
- Even where they exist, requirements documents often fail an architect in two ways.
  - First, most requirements specifications is focused on the required features and functionality of a system. We rarely see adequate capture of QA requirements in practice.
  - Second, much of what is useful to an architect is not in even the best document. ASRs often derive from business goals in the development organization itself.
- Although requirements documents won't tell an architect the whole story, they are an important source of ASRs.

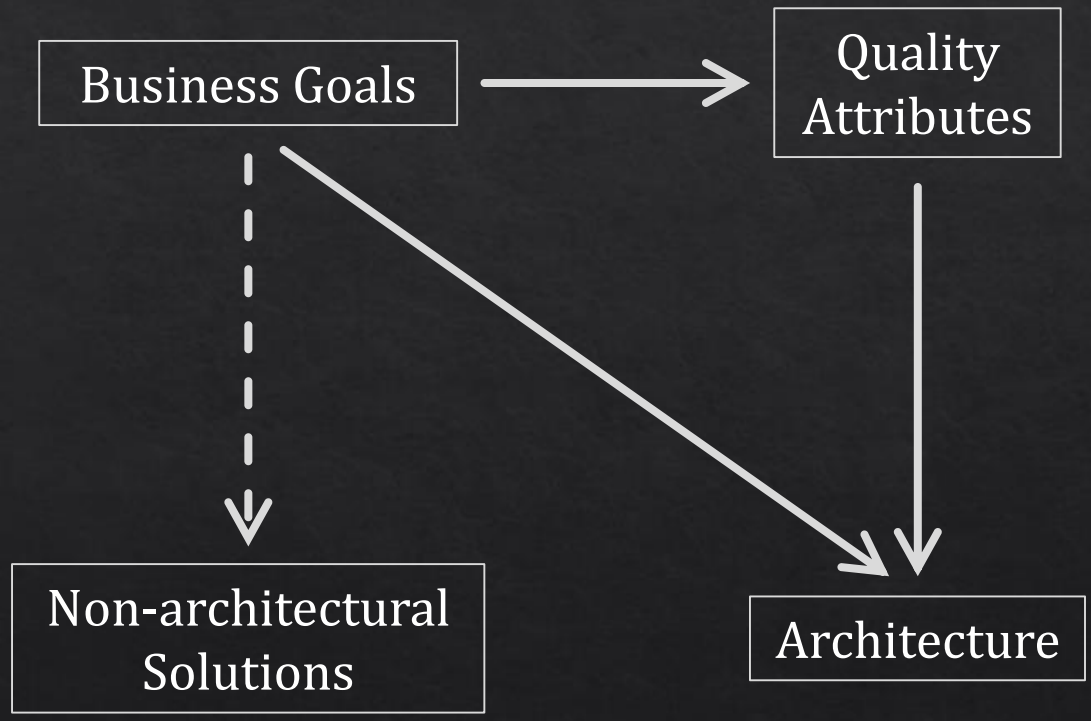
Gathering ASRs by Interviewing Stakeholders

- Architects are often called upon to help set the QA requirements for a system.
- Interviewing the relevant stakeholders is the surest way to learn what they know and need.
- Stakeholders often have no idea what QAs they want in a system.
- Architects often have very good ideas about what QAs are exhibited by similar systems.
- Architects can usually provide quick feedback as to which QAs are going to be straightforward to achieve and which are going to be problematic or even prohibitive.

Gathering ASRs by Understanding the Business Goals

There are three possible relationships between business goals and an architecture:

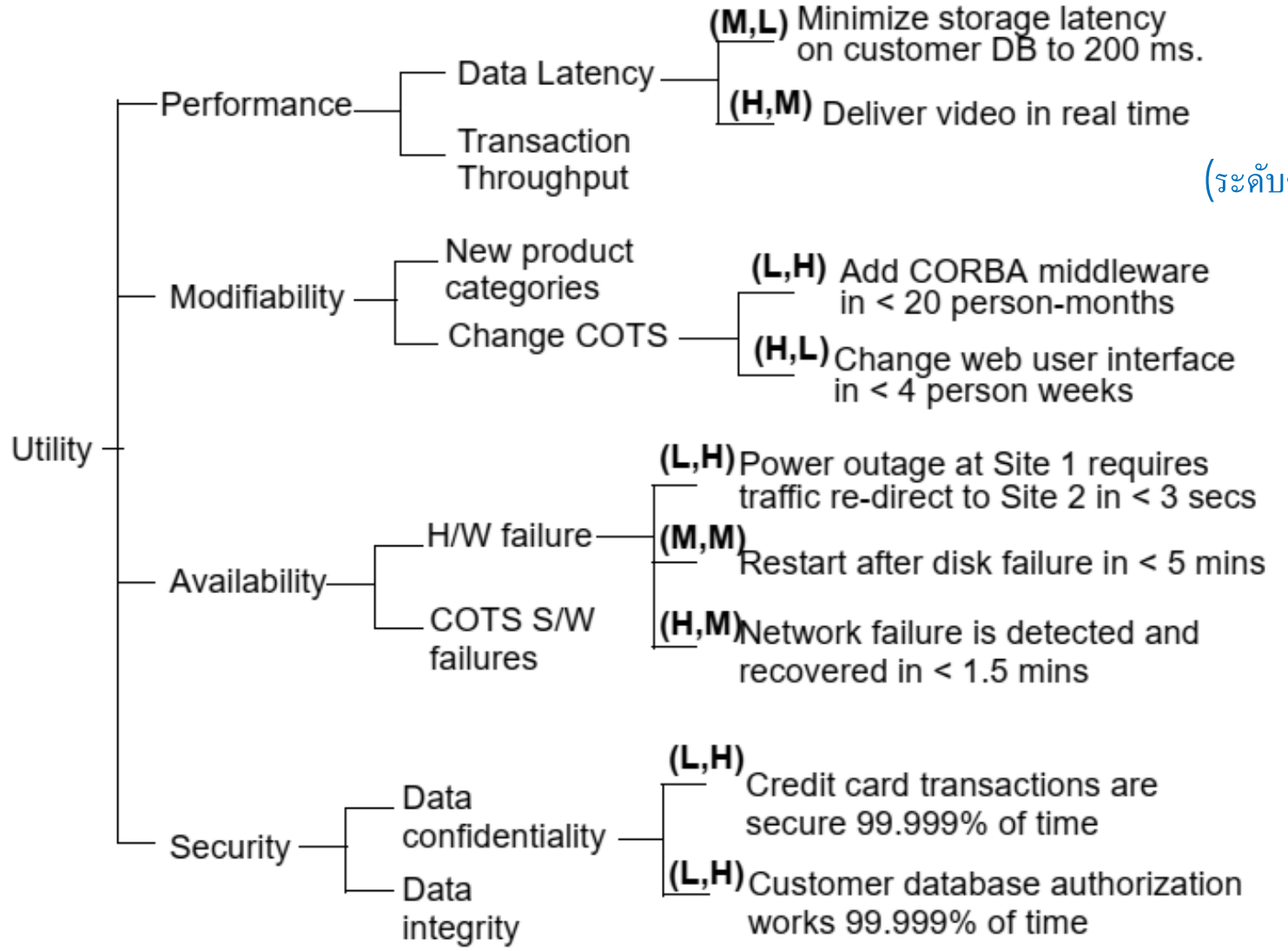
1. Business goals often lead to quality attribute requirements
2. Business goals may directly affect the architecture without precipitating a QA requirement at all
3. No influence at all



Utility Tree

Utility Tree

- Architects can use a construct called a utility tree to record ASRs in one place.
- A utility tree begins with the word “utility” as the root node. We then elaborate this root node by listing the major QAs that the system is required to exhibit.
- Under each quality attribute, record a specific refinement of that QA. For example, performance might be decomposed into “data latency” and “transaction throughput.” Or it might be decomposed into “user wait time” and “time to refresh web page.”
- Once the ASRs are recorded and placed in the tree, you can now evaluate them against the two criteria we listed above: the importance of each node to the success of the system and the degree of perceived risk posed by the achievement of this node.
- You can use any scale you like, but we find that a simple “H” (high), “M” (medium), and “L” (low) suffice for each criterion.



(ระดับความสำคัญ,ระดับความเสี่ยง)

Quality Attribute	Attribute Refinement	ASR
Performance	Transaction response time	A user updates a patient's account in response to a change-of-address notification while the system is under peak load, and the transaction completes in less than 0.75 second. (H,M)
		At peak load, the system is able to complete 150 normalized transactions per second. (M,M)
	Throughput	A user updates a patient's account in response to a change-of-address notification while the system is under double the peak load, and the transaction completes in less than 4 seconds. (L,M)
Usability	Proficiency training	A new hire with two or more years' experience in the business becomes proficient in Nightingale's core functions in less than 1 week. (M,L)
		A user in a particular context asks for help, and the system provides help for that context, within 3 seconds. (H,M)
	Normal operations	A hospital payment officer initiates a payment plan for a patient while interacting with that patient and completes the process without the system introducing delays. (M,M)
Configurability	User-defined changes	A hospital increases the fee for a particular service. The configuration team makes the change in 1 working day; no source code needs to change. (H,L)

Architecture Tradeoff Analysis Method (ATAM)

Architecture Tradeoff Analysis Method (ATAM)

- ATAM was developed by the Software Engineering Institute at the Carnegie Mellon University.
- The ATAM is designed so that:
  - Evaluators need not be familiar with the architecture or its business goals.
  - The system need not yet be constructed.
  - There may be a large number of stakeholders.

Participants in the ATAM

1. **The evaluation team**
   - This group is external to the project whose architecture is being evaluated.
   - They need to be recognized as competent, unbiased outsiders with no hidden agendas.
2. **Project decision makers**
   - People who have the authority to mandate changes to it, such as the project manager.
   - A cardinal rule of architecture evaluation is that the architect must willingly participate.
3. **Architecture stakeholders**
   - Stakeholders have a vested interest in the architecture performing as advertised.
   - Stakeholders include developers, testers, integrators, maintainers, performance engineers, users, builders of systems interacting with the one under consideration, and others.

ATAM Evaluation Team Roles

Role	Responsibilities
Team Leader	Sets up the evaluation; coordinates with client, making sure client's needs are met; establishes evaluation contract; forms evaluation team; sees that final report is produced and delivered (although the writing may be delegated)
Evaluation Leader	Runs evaluation; facilitates elicitation of scenarios; administers scenario selection/prioritization process; facilitates evaluation of scenarios against architecture; facilitates on-site analysis
Scenario Scribe	Writes scenarios on flipchart or whiteboard during scenario elicitation; captures agreed-on wording of each scenario, halting discussion until exact wording is captured
Proceedings Scribe	Captures proceedings in electronic form on laptop or workstation: raw scenarios, issue(s) that motivate each scenario (often lost in the wording of the scenario itself), and resolution of each scenario when applied to architecture(s); also generates a printed list of adopted scenarios for handout to all participants
Questioner	Raises issues of architectural interest, usually related to the quality attributes in which he or she has expertise

Outputs of the ATAM

1. A concise presentation of the architecture.
2. Articulation of the business goals.
3. Prioritized quality attribute requirements expressed as quality attribute scenarios.
4. A set of risks and nonrisks.
5. A set of risk themes.
6. Mapping of architectural decisions to quality requirements.
7. A set of identified sensitivity and tradeoff points.

Phases of the ATAM

Phases of the ATAM

Phase	Activity	Participants	Typical Duration
0	Partnership and preparation	Evaluation team leadership and key project decision makers	Proceeds informally as required, perhaps over a few weeks
1	Evaluation	Evaluation team and project decision makers	1–2 days followed by a hiatus of 1–3 weeks
2	Evaluation (continued)	Evaluation team, project decision makers, and stakeholders	2 days
3	Follow-up	Evaluation team and evaluation client	1 week

Phase 0: Partnership and Preparation

- The evaluation team leadership and the key project decision makers meet to work out the details of the exercise.
- The project representatives brief the evaluators about the project so that the team can be supplemented by people who possess the appropriate expertise.
- The two groups agree on:
  - logistics, such as the time and place of meetings, who brings the flipcharts
  - a preliminary list of stakeholders (by name, not just role)
  - when the final report is to be delivered and to whom
  - formalities such as a statement of work or nondisclosure agreements (NDA)
- The evaluation team examines the architecture documentation.

Phase 1 & Phase 2: Evaluation

- Prior to the start of phase 1, the evaluation team will have studied the architecture documentation and will have a good idea of what the system is about.
- The ATAM analysis phases (phase 1 and phase 2) consist of 9 steps.
- Steps 1 – 6 are carried out in phase 1 with the evaluation team and the project's decision makers: typically, the architecture team, project manager, and project sponsor.
- Steps 7 – 9 are carried out in phase 2 with all stakeholders present.

More details later.

Phase 3: Follow-up

- The evaluation team produces and delivers a written final report.
- The report first circulated to key stakeholders to make sure that it contains no errors of understanding.
- After the review is complete the report is delivered to the person who commissioned the evaluation.

Steps of the Evaluation in Phase 1

Step 1: Present the ATAM

- The evaluation leader:
  - presents the ATAM to the project representatives.
  - explains the process that everyone will be following.
  - sets the context and expectations for the remainder of the activities.
  - describes the ATAM steps in brief and the outputs of the evaluation.

Step 2: Present the Business Drivers

- The project representatives as well as the evaluation team members needs to understand the context for the system and the primary business drivers motivating its development.
- A project decision maker (ideally the project manager or the system's customer) presents a system overview from a business perspective.
- The presentation should describe the following:
  - The system's most important functions
  - Any relevant technical, managerial, economic, or political constraints
  - The business goals and context as they relate to the project
  - The major stakeholders
  - The architectural drivers (that is, the architecturally significant requirements)

Step 3: Present the Architecture

- The lead architect (or architecture team) makes a presentation describing the architecture at an appropriate level of detail. The “appropriate level” depends on several factors:
  - how much of the architecture has been designed and documented
  - how much time is available
  - the nature of the behavioral and quality requirements
- In this presentation the architect covers technical constraints such as operating system, hardware, or middleware prescribed for use, and other systems with which the system must interact.
- Most important, the architect describes the architectural approaches (or patterns, or tactics) used to meet the requirements.

Step 4: Identify Architectural Approaches

- The ATAM focuses on analyzing an architecture by understanding its approaches.
- The architect is asked to explicitly name the patterns and tactics used, but the evaluation team may as well spot ones not mentioned. For example:
  - A layered pattern tends to bring portability to a system, possibly at the expense of performance.
  - A publish-subscribe pattern is scalable in the number of producers and consumers of data.
  - The active redundancy tactic promotes high availability.
- The evaluation team simply catalogs the patterns and tactics that have been identified.
- The list is publicly captured by the scribe for all to see and will serve as the basis for later analysis.

Step 5: Generate Quality Attribute Utility Tree

- A quality attribute utility tree makes the requirements concrete by defining precisely the relevant quality attribute requirements that the architects were working to provide.
- Broad goals in step 2 such as “modifiability” or “high throughput” or “ability to be ported to a number of platforms” establish important context and direction.
- However, they are not specific enough to let us tell if the architecture suffices. Modifiable in what way? Throughput that is how high? Ported to what platforms and in how much time?
- In this step, the evaluation team works with the project decision makers to identify, prioritize, and refine the system’s most important quality attribute goals. These are expressed as scenarios, as we have already learned.

Step 6: Analyze Architectural Approaches

- The evaluation team examines the highest-ranked scenarios (as identified in the utility tree) one at a time.
- The architect is asked to explain how the architecture supports each one.
  - Evaluation team members—especially the questioners—probe for the architectural approaches that the architect used to carry out the scenario.
  - The evaluation team documents the relevant architectural decisions and identifies and catalogs their risks, nonrisks, sensitivity points, and tradeoffs.
- The analysis is not meant to be comprehensive. The key is to elicit sufficient architectural information to establish some link between the architectural decisions and the quality attribute requirements.

Examples of Scenario Walkthrough in Step 6

- The frequency of heartbeats affects the time in which the system can detect a failed component. Some assignments will result in unacceptable values of this response—these are risks.
- The frequency of heartbeats determines the time for detection of a fault. Higher frequency leads to improved availability but will also consume more processing time and communication bandwidth (potentially leading to reduced performance). This is a tradeoff.
- The number of simultaneous database clients will affect the number of transactions that a database can process per second. Thus, the assignment of clients to the server is a sensitivity point with respect to the response as measured in transactions per second.

Hiatus

- The evaluation team summarizes what it has learned and interacts informally (usually by phone) with the architect during a hiatus of a week or two.
- More scenarios might be analyzed during this period, if desired, or questions of clarification can be resolved.

Steps of the Evaluation in Phase 2

The Start of Phase 2

- Additional stakeholders join the evaluation.
- Step 1 is repeated so that the stakeholders understand the method and their roles.
- Then the evaluation leader recaps the results of steps 2 through 6, and shares the current list of risks, nonrisks, sensitivity points, and tradeoffs.

Step 7: Brainstorm and Prioritize Scenarios (1)

- In this step, the evaluation team asks the stakeholders to brainstorm scenarios that are operationally meaningful with respect to the stakeholders' individual roles. Such as:
  - A maintainer will likely propose a modifiability scenario
  - A user will probably come up with a scenario that expresses useful functionality or ease of operation
  - A quality assurance person will propose a scenario about testing the system or being able to replicate the state of the system leading up to a fault.
- The purpose of scenario brainstorming is to take the pulse of the larger stakeholder community: to understand what system success means for them.
- Scenario brainstorming works well in larger groups, creating an atmosphere in which the ideas and thoughts of one person stimulate others' ideas.

Step 7: Brainstorm and Prioritize Scenarios (2)

- Once the scenarios have been collected, they must be prioritized, for the same reasons that the scenarios in the utility tree needed to be prioritized: the evaluation team needs to know where to devote its limited analytical time.
- First, stakeholders are asked to merge scenarios they feel represent the same behavior or quality concern.
- Then stakeholders vote for those scenarios they feel are most important.
  - Each stakeholder is allocated a number of votes equal to 30 percent of the number of scenarios,1 rounded up. So, if there were 40 scenarios collected, each stakeholder would be given 12 votes.
  - These votes can be allocated in any way that the stakeholder sees fit: all 12 votes for 1 scenario, 1 vote for each of 12 distinct scenarios, or anything in between.
- The list of prioritized scenarios is compared with those from the utility tree exercise.

Step 8: Analyze Architectural Approaches

- After the scenarios have been collected and prioritized in step 7, the evaluation team guides the architect in the process of carrying out the highest ranked scenarios.
- The architect explains how relevant architectural decisions contribute to realizing each one.
- Ideally this activity will be dominated by the architect's explanation of scenarios in terms of previously discussed architectural approaches.
- In this step the evaluation team performs the same activities as in step 6, using the highest-ranked, newly generated scenarios.
- Typically, this step might cover the top five to ten scenarios, as time permits.

Step 9: Present Results

- In step 9, the evaluation team groups risks into risk themes, based on some common underlying concern or systemic deficiency.
- For each risk theme, the evaluation team identifies which of the business drivers listed in step 2 are affected. For example:
  - A group of risks about inadequate or out-of-date documentation might be grouped into a risk theme stating that documentation is given insufficient consideration.
  - A group of risks about the system's inability to function in the face of various hardware and/or software failures might lead to a risk theme about insufficient attention to backup capability or providing high availability.
- The collected information from the evaluation is summarized and presented to stakeholders. This takes the form of a verbal presentation with slides.

Lightweight Architecture Evaluation

Lightweight Architecture Evaluation

- Although one attempts to use time in an ATAM exercise as efficiently as possible, it remains a substantial undertaking.
- Investing this amount of time only makes sense on a large and costly project, where the risks of making a major mistake in the architecture are unacceptable.
- Bass et al. have developed a Lightweight Architecture Evaluation method, which may take place in a single day, or even a half-day meeting.
- A Lightweight Architecture Evaluation may be carried out entirely by members internal to the organization.
- The lower level of scrutiny and objectivity may not probe the architecture as deeply, but this is a cost/benefit tradeoff that is entirely appropriate for many projects.

A Typical Agenda for Lightweight Architecture Evaluation (1)

Step	Time	Notes
1: Present the ATAM	0 hrs	The participants are familiar with the process. This step may be omitted.
2: Present Business Drivers	0.25 hrs	The participants are expected to understand the system and its business goals and their priorities. Fifteen minutes is allocated for a brief review to ensure that these are fresh in everyone's mind and that there are no surprises.
3: Present Architecture	0.5 hrs	Again, all participants are expected to be familiar with the system and so a brief overview of the architecture, using at least module and C&C views, is presented and 1 to 2 scenarios are traced through these views.
4: Identify Architectural Approaches	0.25 hrs	The architecture approaches for specific quality attribute concerns are identified by the architect. This may be done as a portion of step 3.

A Typical Agenda for Lightweight Architecture Evaluation (2)

Step	Time	Notes
5: Generate Quality Attribute Utility Tree	Variable 0.5 hrs to 1.5 hrs	Scenarios might exist: part of previous evals, part of design, part of requirements elicitation. If you've got 'em, use 'em and make them into a tree. Half hour. Otherwise, it will take longer. A utility tree should already exist; the team reviews the existing tree and updates it, if needed, with new scenarios, new response goals, or new scenario priorities and risk assessments.
6: Analyze Architectural Approaches	2 to 3 hrs	This step—mapping the highly ranked scenarios onto the architecture—consumes the bulk of the time and can be expanded or contracted as needed.

A Typical Agenda for Lightweight Architecture Evaluation (3)

Step	Time	Notes
7: Brainstorm and Prioritize Scenarios	0 hrs	This step can be omitted as the assembled (internal) stakeholders are expected to contribute scenarios expressing their concerns in step 5.
8: Analyze Architectural Approaches	0 hrs	This step is also omitted, since all analysis is done in step 6.
9: Present Results	0.5 hrs	At the end of an evaluation, the team reviews the existing and newly discovered risks, nonrisks, sensitivities, and tradeoffs and discusses whether any new risk themes have arisen.
Total	4 to 6 hrs	

There is no final report, but (as in the regular ATAM) a scribe is responsible for capturing results, which can then be distributed and serve as the basis for risk remediation.

Lightweight Architecture Evaluation: Pros & Cons

Pros:

- An entire Lightweight Architecture Evaluation can be prosecuted in less than a day—perhaps an afternoon.
- This version of evaluation is inexpensive, easy to convene, and relatively low ceremony, so it can be quickly deployed whenever a project wants an architecture quality assurance sanity check.

Cons:

- The evaluation team, being internal, is typically not objective, and this may compromise the value of its results—one tends to hear fewer new ideas and fewer dissenting opinions.
- The results will depend on how well the assembled team understands the goals of the method, the techniques of the method, and the system itself.