Conclusion Thoughts

Now at last I want to conclude the main thoughts that I want you to know through beginning your future in software architecture especially if you are just beginning your study in this field or just beginning your first day in your job as an architect:

- First: *qualities* are the basic part for any individual, company, and organization that need to start building their products, because this will make the competition between products or even between the people whose responsibility is building the products in the market. It is the main target to building any system.
- Second: knowing *the patterns and tactics*. As beginning with the first thought, the quality is the basic thing that all must focus on, and the second thought that you must know is that gaining the quality can be done through other stages in the life cycle. Here the focus is on software architecture stage because the title of this book is going around this stage. And because I talked especially on software architecture and its relation to building high-quality products, then you need to know the architectural patterns and their tactics to use the most appropriate of them to have the high-quality products and also make a trade-off between qualities in the same product.
- Third: *stakeholders*. Through years ago I read many books and article papers and attend conferences on software architecture; through that I saw the important advice from professionals; and this advice says: know your stakeholders. Knowing the stakeholder is a very important part to any architect because you as an architect can extract the goals that they want from the product that you can go through this road to reach the quality.
- Fourth: *evaluation.* It *is* very important to any architecture because it avoid architectures from disaster. To put it in a different way, if you were building a house, you wouldn't proceed without carefully looking at the blueprints before building began. It is also the right thing to know the appropriate method according to the quality of the product as what said through this book but in general ATAM method is used as an evaluation method. Also you must know that the evaluation will tell you that the architecture is suitable according with one goal (or set of

L. Khalid, *Software Architecture for Business*,
https://doi.org/10.1007/978-3-030-13632-1

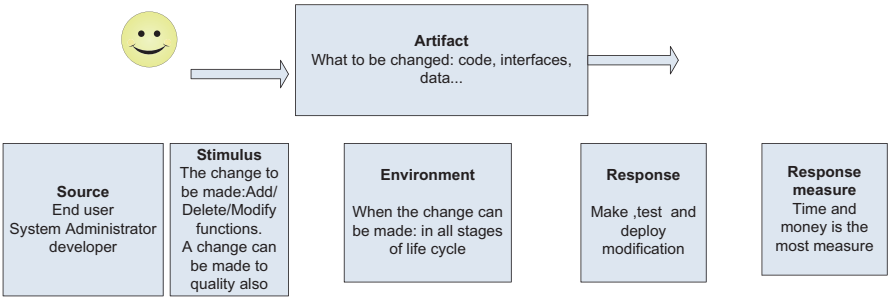
goals) and according to other qualities there is a problem, sometimes one goal is important than another sometimes a conflict occurs between qualities, the manager of the project will have the decision to make if the architecture evaluates good in some areas and not evaluated very well in other area.

- Also you showed that SAAM method focuses on modifiability quality in its different forms (such as portability, subsetability, and variability) and functionality, while ARID method provides a deep understanding about the suitability of part of the architecture to be used by developer to complete their responsibilities. That is why architecture evaluation methods can be choosing according to the qualities that are related to that architecture.
- In short, architecture evaluation produces better architectures.

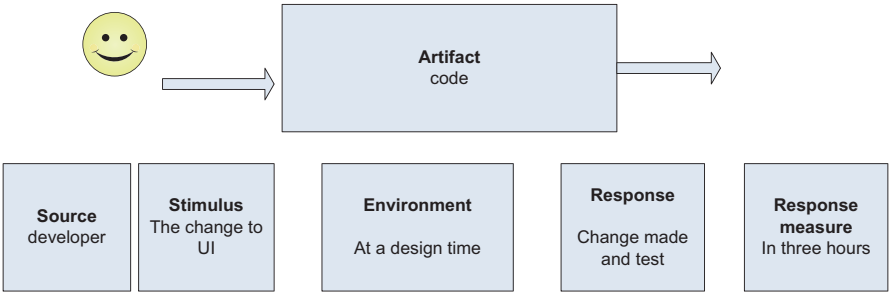
Finally I hope you have the basic stone to start your journey in software architecture and its relation to building high-quality products.

Appendix A

General Scenario for Modifiability

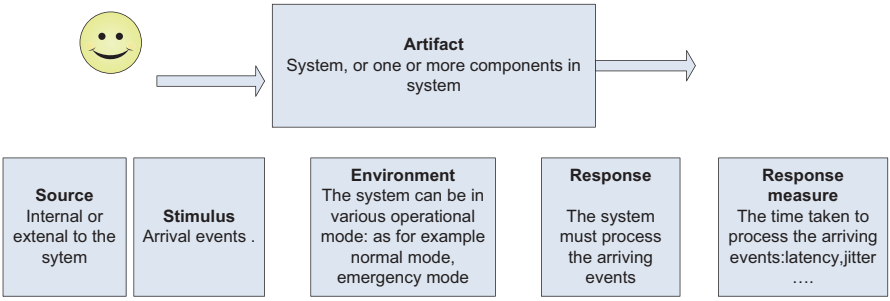


Concrete Scenario for Modifiability

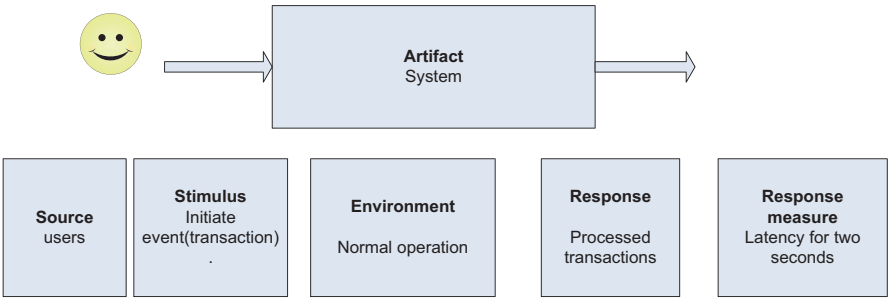


L. Khalid, *Software Architecture for Business*,
https://doi.org/10.1007/978-3-030-13632-1

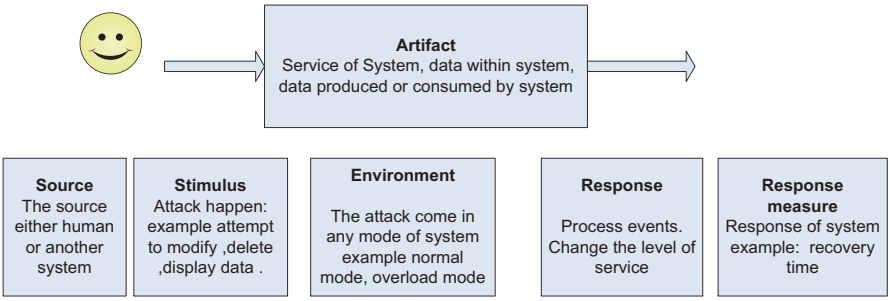
Role of General Scenario for Performance



Concrete Scenario for Performance



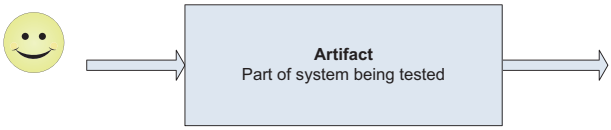
General Scenario for Security



Concrete Scenario for Security

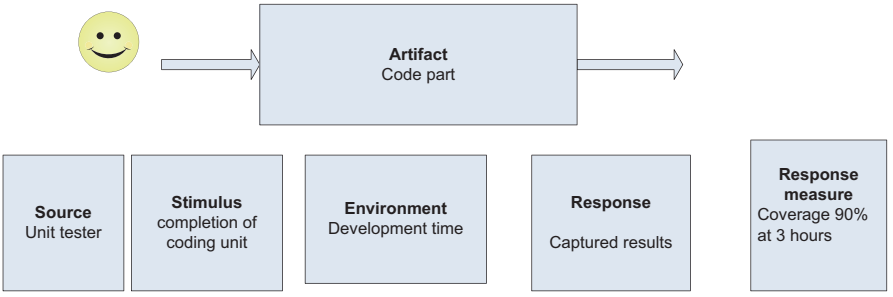
Source	Stimulus	Environment	Response	Response measure
Distinguished employee from remote place	Attempts to modify pay rate .	Normal operation	System maintains audit trail	Resore correct data within a day

General Scenario for Testability

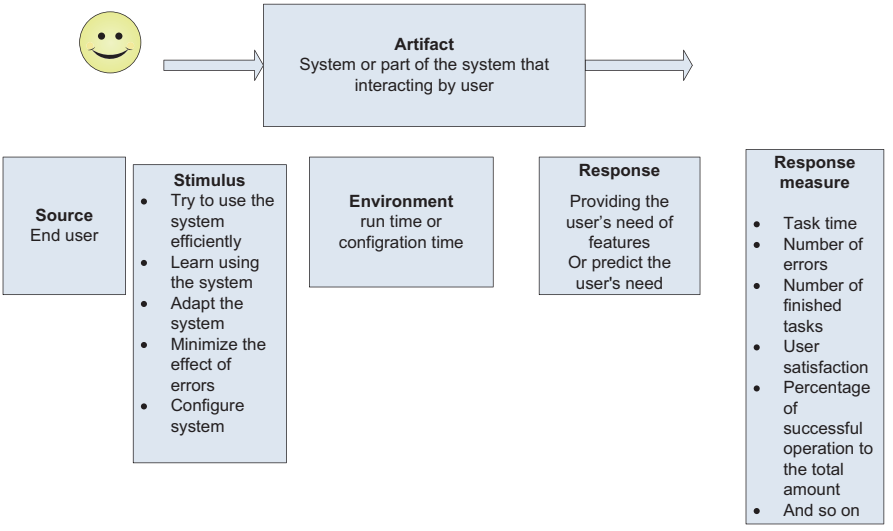


Source	Stimulus	Environment	Response	Response measure
Could be human or automated tester	Set of tests through completion of coding increment example: service	Happen at: development time ,compile time, deployment time ,run time, design time, integration time	Controlled the system to perform test and the result can be observed	• Effort to find a fault. • effort o achieve a coverage %, • Probability of fault being revealed by next stage, • Time to carry out tests, • Reduction in risks • Time to prepare test environment • Effort to detect fault And so on

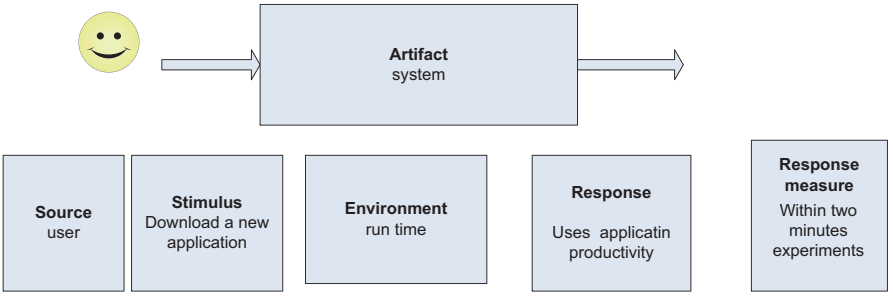
Concrete Scenario for Testability



General Scenario for Usability



Concrete Scenario for Usability



Appendix B

	Software structure	Element types	Relation	Useful for
Module structure	Decomposition	Module	Is a sub module of	Resource allocation and project structuring and planning; information hiding, encapsulation; configuration control
	Uses	Module	Uses	Engineering subsets; engineering extensions
	Layered	Layer	Requires the correct presence of; uses the services of; provides abstraction to	Incremental development; implementing systems on top of "virtual machines"
	Class	Classes, objects	Is an instance of; shares access methods of	In object-oriented design systems
C&C structure	Service	Service, registry, others	Run concurrently with, etc.	Scheduling analysis, performance analysis
	Concurrency	Process, thread	Can run in parallel	Identifying locations where resource contention exists, where threads may fork, join, be created or be killed
Allocation structure	Deployment	Components, hardware elements	Allocated to; migrates to	Performance, availability, security analysis
	Implementation	Modules, file structure	Stored in	Configuration control, integration, test activities
	Work assignment	Modules, organizational unit	Assigned to	Project management, best use of expertise, management of commonality

Architectural structure for the system

L. Khalid, *Software Architecture for Business*,
https://doi.org/10.1007/978-3-030-13632-1

Appendix C

ATAM (Architecture Trade-off Analysis Method)

First of all, ATAM gets its name because it reveals on how well an architecture satisfies the quality goals, and the most important thing is how trade-off between qualities has been used for over a decade to evaluate software architectures and draws its technique from three areas:

- The architectural style concepts
- The quality attributes analysis communities
- SAAM method the predecessor method to the ATAM

ATAM steps can be separated into four main groups.
Presentation: exchange the information through the presentation which includes:

1. Present the ATAM: evaluation leader introduce the method to the participate.
2. Present the business driver: the project manager or customer describes the business goals that motivated the development and then what is the main architectural driver (such as time to market).
3. Present the architecture: here, the role of architect is to describe the architecture and focus his attention on how it addresses the business driver.

Investigation and analysis

4. Identify the architectural approaches: the approaches of architecture are identified by architect but not analyzed.
5. Generate the utility tree: the quality attribute of system is elicited and specified down to the level of scenarios, stimulus, and response and prioritized.
6. Analyze the architectural approach: based on the high priority in previous step, the architectural approaches that address the scenario will be analyzed. Here, risk, sensitivity point, and non-risk trade-off will be identified in this step.

L. Khalid, *Software Architecture for Business*,
https://doi.org/10.1007/978-3-030-13632-1

Testing

7. Brainstorming and prioritize scenario: the set of scenario that elicited earlier will be prioritized through voting process involving all the stakeholders.
8. Analyze the architectural approaches: in this step, reiteration of the activities in step 6 occurs, but high-ranked scenarios from step 7 are used. Those scenarios are supposed to be the test cases to confirm the analysis performed thus far.

Reporting

9. Present the result based on the information collected during the steps, and then the results will be presented to the stakeholders.

These steps can be carried out through the following phases:

Phase	Activity
0	Preparation
1	Evaluation from steps 1 to 6
2	Evaluation from steps 7 to 9
3	Process improvement and delivery

According to quality, ATAM is not oriented to any specific type of quality

SAAM (the Software Architecture Analysis Method)

It is a simple method and good place to start if this is the first time you evaluate and architecture, *especially if you work on modifiability and functionality*

The output tangible results from SAAM evaluation are:

- Mapping onto scenarios that represent possible future changes to the system
- Understanding the functionality of the system

SAAM steps are:

1. *Develop scenario*: these scenarios represent tasks related to different roles of stakeholders; these scenarios are all brainstorming exercise, and also they are collected in two or more elicitations.
2. *Describe architecture*: describing the architectures should be through notations that will be understand by parties and must specify data components with related connections and system's computation, and all these will take in the form of natural language or some other formal specification.
3. *Classify/prioritize scenarios*: in SAAM, scenarios are classified into direct and indirect scenarios. Prioritization is done by choosing the most important scenarios and that is done by stakeholders through voting process.
4. *Individually evaluate indirect scenario*: chosen scenarios are mapped onto the architectural description. According to direct scenarios, the architect shows how

these scenarios are executed by architecture. On the other hand, in indirect scenarios, the architect should describe how the architecture needs to be changed in order to accommodate the scenarios.

5. *Assess scenario interaction*: interaction scenarios are called to the indirect scenarios that require changes to a single component of architecture. These types of scenarios are important for two reasons: first reason is it depicts allocation of functionality to the product's design. Second important reason is that it can expose the architecture not documented to the right level of structural decomposition.
6. *Create overall evaluation*: at the final stage, a weight is assigned to each scenario for the reason of showing the related importance to the success of system. This weight usually attaches to the business goals that support the scenarios.

Notes

- Step 1 and 2 done on interleaved way or on several iterations.
- There are two important concepts in these iterations, and those are direct and indirect scenarios. Scenarios represent tasks relevant to different roles such as developer, maintainer, customer, etc.

Direct scenarios are those scenarios that are specified by the architecture through the execution of the system. Such types of scenarios are increasing the understanding of the architecture to the stakeholders and allow the systematic exploration of their architectural qualities. On the other hand, there is *an indirect scenario* which defines as that requires a modification to the architecture to be satisfied. It is those scenarios that play as a central role to the measurements of the degree to which an architecture can hold evolutionary changes that are important to the stakeholder. The indirect scenarios measure the suitability for continuing use throughout the lifetime of the family.

ARID (Active Reviewers for the Intermediate Design)

It is a method that used to evaluate the architecture partially or in the intermediate designs when all the architecture passes through. This method lies at the intersection between ATAM and ADRs (active design reviewers) methods. It consists of nine steps going through two phases.

Phase 1: Rehearsal

In this phase, a meeting between the lead designer and review facilitator to prepare for exercise.

Step 1: identify the reviewers—the reviewers of the ARID are the design's stakeholders.

Step2: prepare the design briefing—the designers organize a brief explanation of the design. The goal is to present the design in a good so that members can use it in a sufficient way.

Step3: prepare the seed scenarios—here, designer and the review facilitator present a set of seed scenarios; the aim is roughly a dozen scenarios.

Step 4: prepare the materials—this step is the preparation to phase 2 by preparing copies of the presentation, scenarios, and review agenda to the reviewers during phase 2.

Phase 2: Review

Step 5: present ARID—the explanation of the steps of ARID to the participants is done by review facilitator.

Step 6: present the design—the suitability of the design is the goal of this step. The lead designer gives an overview presentation with examples.

Step 7: brainstorm and prioritize scenarios—this session is just for brainstorming and prioritizes scenarios. Voting process is done through process, and the most voted scenarios received are used to test the design for testability.

Step 8: apply the scenarios—when received the most voted scenarios, the facilitator ask reviewers to expertise the code that uses the design services to solve the problem posed in the scenario whenever the group went to the wrong direction they will be stopped to get the group moving again by providing whatever information is supposed it will be necessary.

Step 9: summarize—recounting the list of issues is done by facilitator; ask the participants for their opinions and thank them for their participations.

Note

ARD method relies on actively engaging reviewers by assigning them review tasks that are structured; it is used to evaluate detailed design of coherent units of software for example modules or components.

According to quality, ARID is used for suitability of the design approach.

Index

L. Khalid, *Software Architecture for Business*,
https://doi.org/10.1007/978-3-030-13632-1