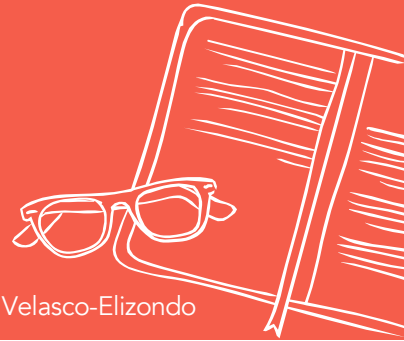


# Software Architecture

## Session 4: Architectural Requirements







# Objectives

- a) Recall the concept of software requirement and software requirement types
- b) Understand the concept of architectural requirements and architectural requirements types
- c) Recognize some methods to elicit architectural requirements





# Outline

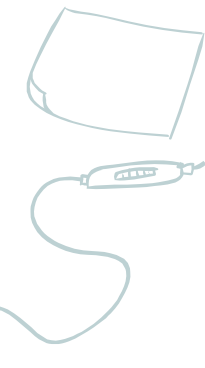


- 
1. Software Requirement
  2. Architectural Requirements
  3. Related Methods
  4. Summary
- 



# Outline

- 
1. Software Requirement
  2. Architectural Requirements
  3. Related Methods
  4. Summary
- 



# What is a software requirement?

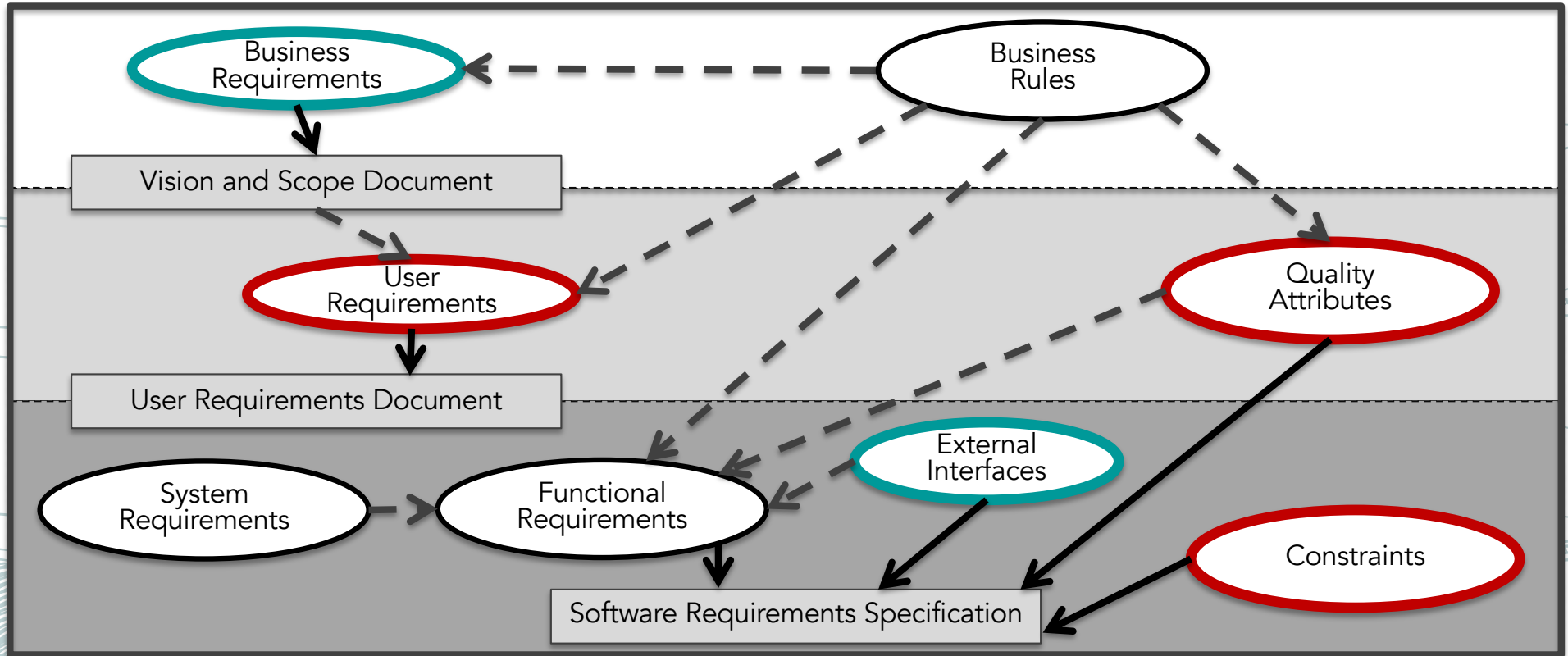




# Software Requirement

- A **property** that a software product must have to provide value to a stakeholder.
- A **specification** of what should be implemented.
- Anything that **drives design choices.**

# Requirements Types



\* From K.E. Wiegers. 2013. *Software Requirements* (3 ed.). Microsoft Press, Redmond, WA, USA.



# Requirements Types

## Business Requirements

Business requirements describe **why the organization is implementing the system** (i.e. the objectives the organization hopes to achieve).

Examples?



# Requirements Types

## User Requirements

- Describe **tasks** or **goals** that specific classes of **users must be able to perform** with the product.
- Describe **what the user will be able** to do with the system.

Examples?

# Requirements Types

## Quality Attributes

- Augment the description of the product's functionality by describing the product's characteristics in various dimensions that are important either to users or to developers.
- These characteristics include attributes such as usability, portability, integrity, efficiency, or robustness.

Examples?



# Requirements Types

## Constraints

Impose restrictions on the choices available to the developer for design and construction of the product.

Examples?

# Requirements Types



## External Interfaces

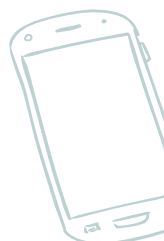
- Describe the connections between your system and the rest of the universe.
- It is important to identify: user, software, Hardware and communications.

Examples?



# Outline

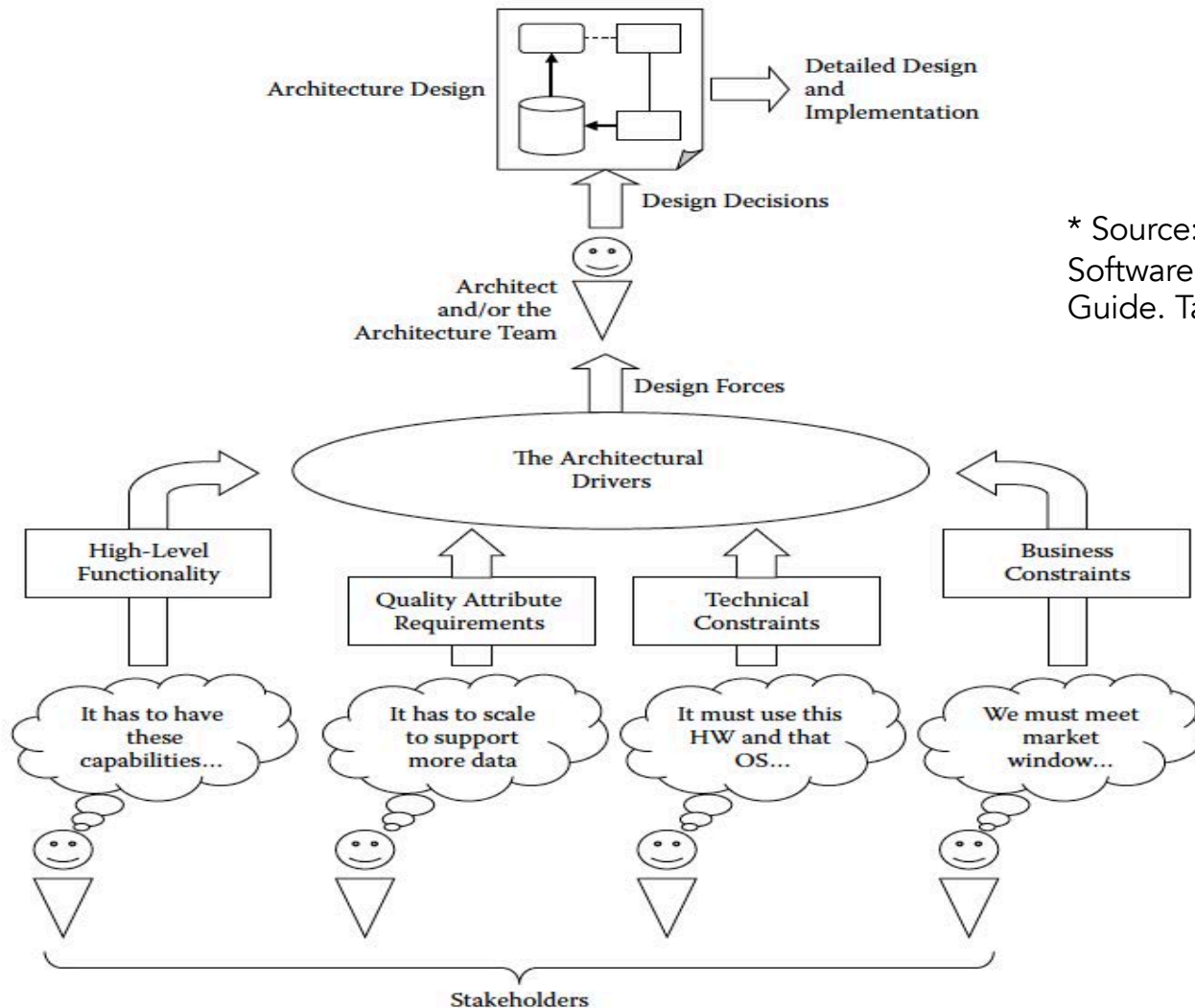
- 
- 
1. Software Requirement
  2. Architectural Requirements
  3. Related Methods
  4. Summary





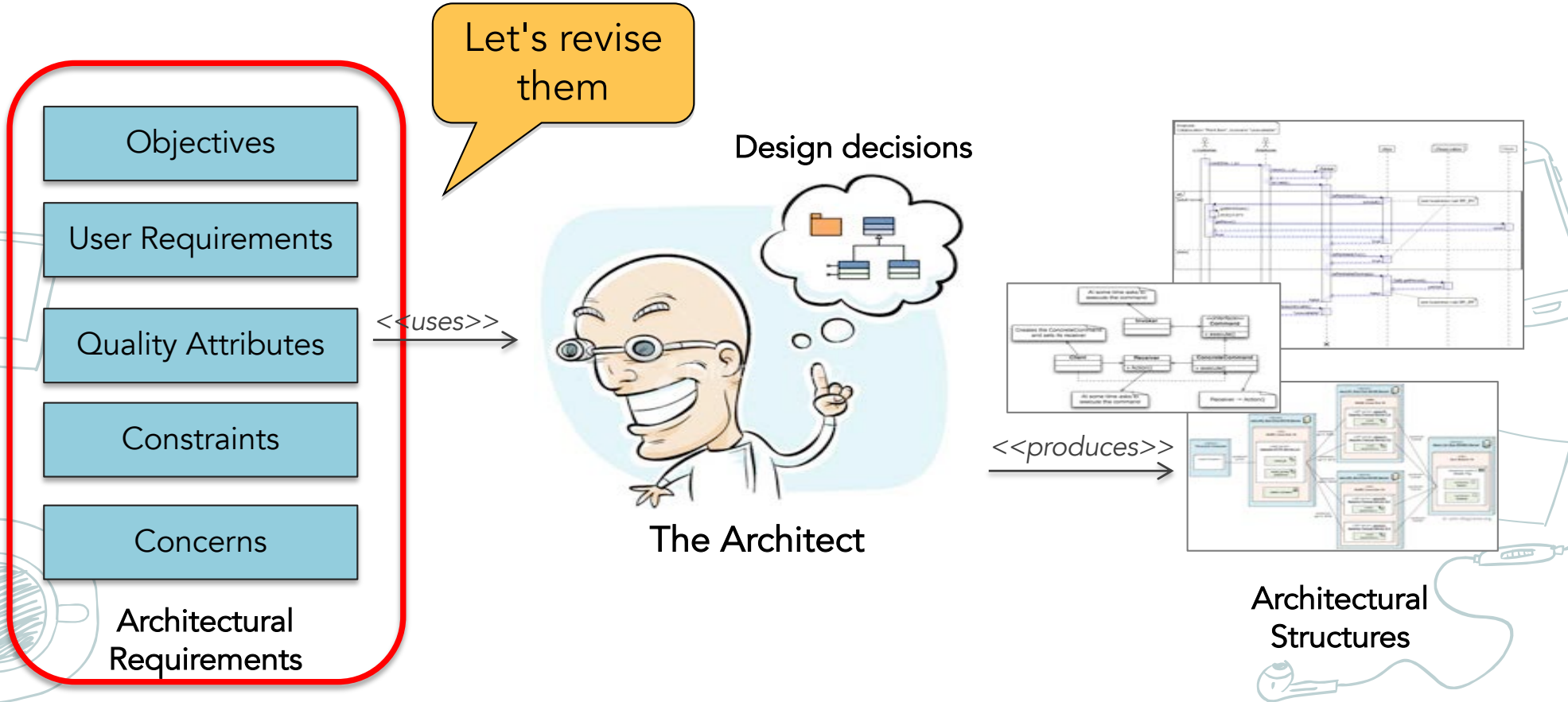
# Architectural Requirements

- A.k.a. **architectural drivers**.
- They are **not all** of the requirements for a system.
- They are those requirements that are most influential to the architect making **early design decisions**.



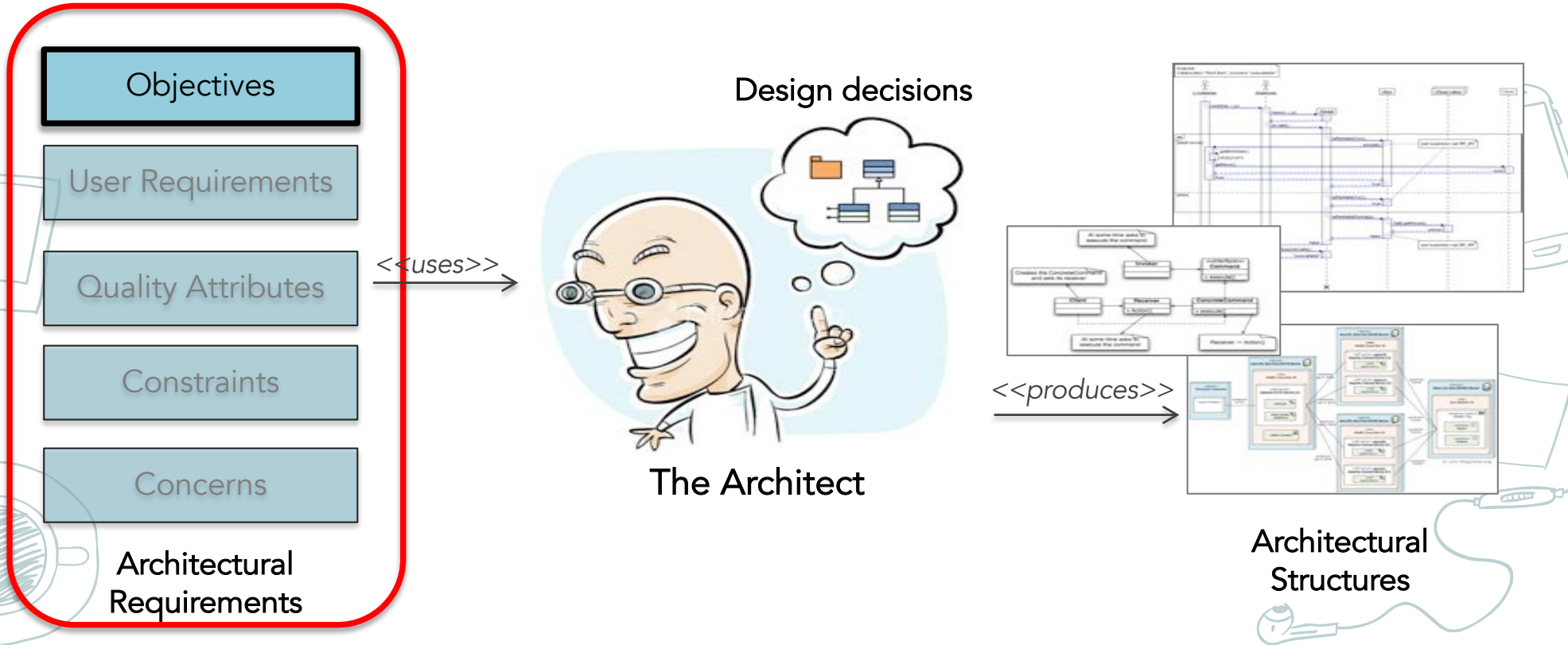
\* Source: Anthony J. Lattanze (2008). Architecting Software Intensive Systems: A Practitioner's Guide. Taylor and Francis/Auerbach.

# Software Architecture Development Cycle





# Software Architecture Development Cycle



# Why I am doing this architecture design?





# Why I am doing this architecture design?

a) Supporting a project proposal.

Such an architecture would **not be very detailed** but with sufficient detail that the units of work are **understood** and hence may be **estimated**.

# Why I am doing this architecture design?

## b) Creating an exploratory prototype ...

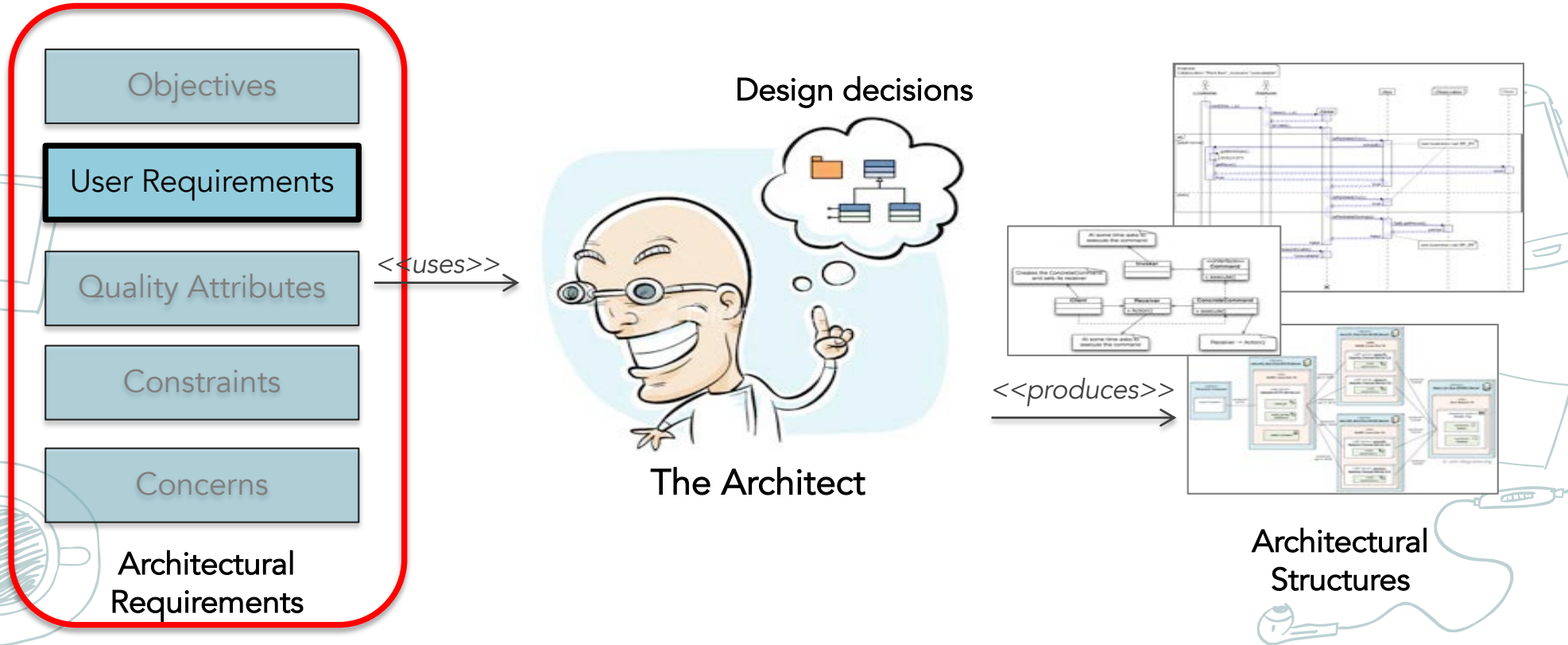
- to explore the domain
- to explore new technology
- to place something executable in front of a customer to get rapid feedback
- to explore some quality attributes

# Why I am doing this architecture design?

## c) Creating an architecture during a software development project

- for an entire or portion of a new system
- for portion of an existing system that is being refactored or replaced.

# Software Architecture Development Cycle





# User Requirements

- Describe **high level tasks** that the users must be able to perform with the system.
- Popular approaches to specify them are **use cases** and **user stories**.





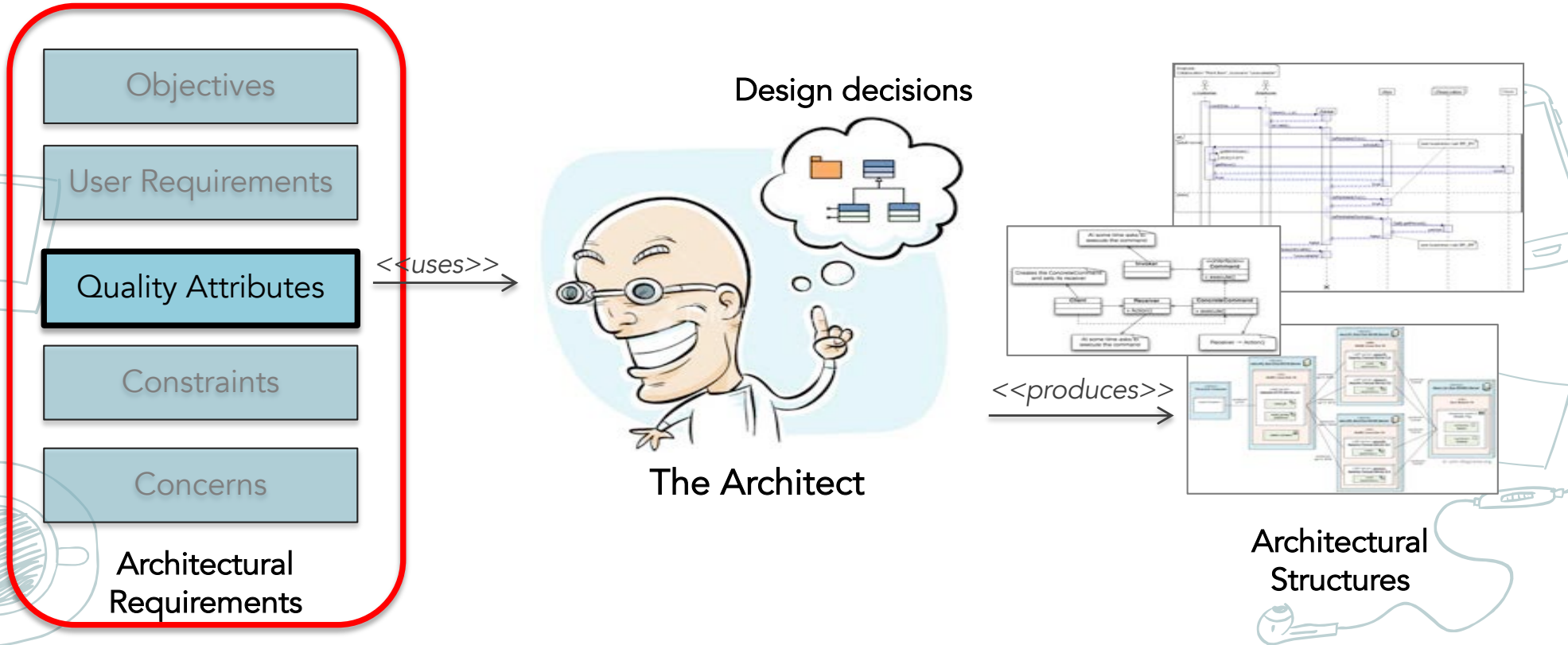
# User Requirements

A criteria to consider a user requirement as architectural includes:

- a) it is critical to achieve the **business goals** that motivate the development of the system.
- b) it implies a high level of **technical difficulty**
- c) it requires the **interaction** of many architectural elements.



# Software Architecture Development Cycle





# Quality Attributes

- Properties of a system that indicate how well it satisfies the needs of its stakeholders.
- While user requirements describe what the system must do, quality attribute requirements describe **how it must do it.**

# Quality Attributes

External quality	Brief description
Availability	The extent to which the system's services are available when and where they are needed
Installability	How easy it is to correctly install, uninstall, and reinstall the application
Integrity	The extent to which the system protects against data inaccuracy and loss
Interoperability	How easily the system can interconnect and exchange data with other systems or components
Performance	How quickly and predictably the system responds to user inputs or other events
Reliability	How long the system runs before experiencing a failure
Robustness	How well the system responds to unexpected operating conditions
Safety	How well the system protects against injury or damage
Security	How well the system protects against unauthorized access to the application and its data
Usability	How easy it is for people to learn, remember, and use the system
Internal quality	Brief description
Efficiency	How efficiently the system uses computer resources
Modifiability	How easy it is to maintain, change, enhance, and restructure the system
Portability	How easily the system can be made to work in other operating environments
Reusability	To what extent components can be used in other systems
Scalability	How easily the system can grow to handle more users, transactions, servers, or other extensions
Verifiability	How readily developers and testers can confirm that the software was implemented correctly

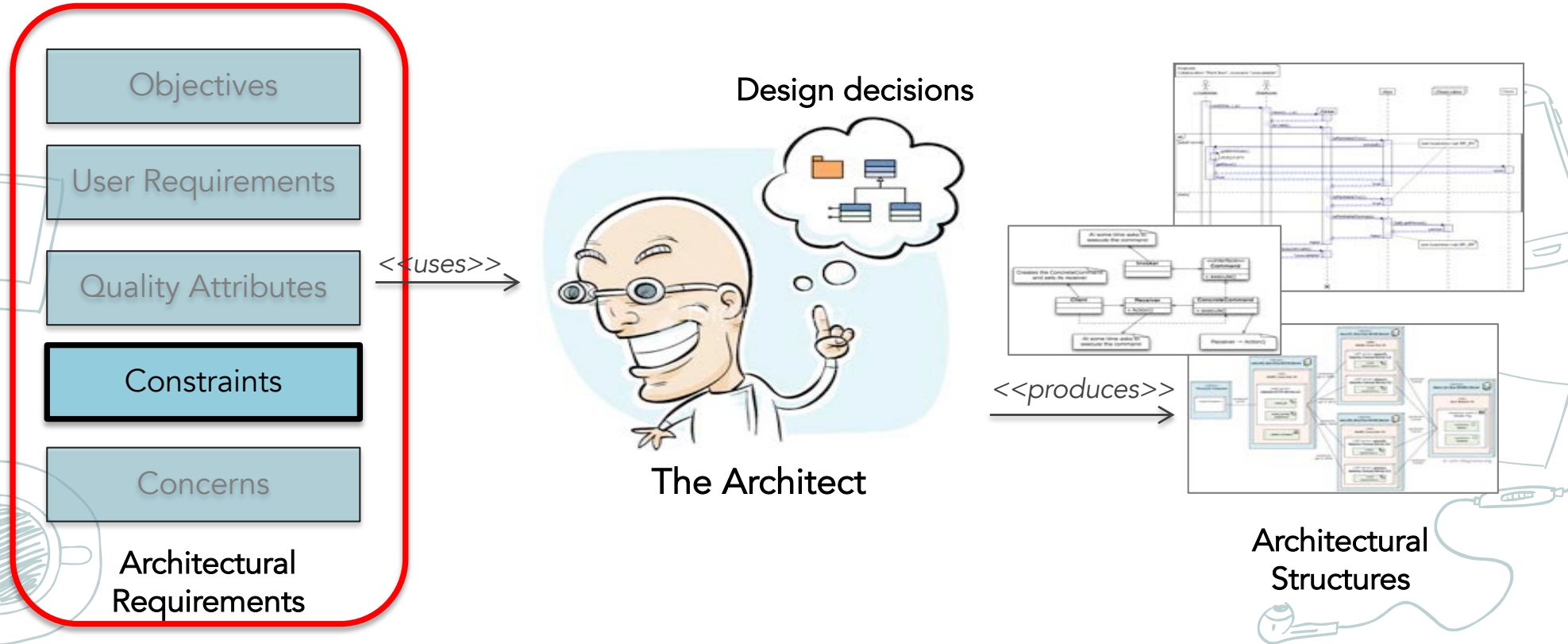


# Quality Attributes

A criteria to consider a quality attribute as architectural includes:

- a) it is critical to achieve the **business goals** that motivate the development of the system.
- b) it implies a high level of **technical difficulty**

# Software Architecture Development Cycle





# Technical Constraints

- They are mandatory **technical decisions** that **are already made** for the architect before he or she begins the process.
- They involve, for example, the use of specific hardware, software products, operating systems, legacy elements, etc.

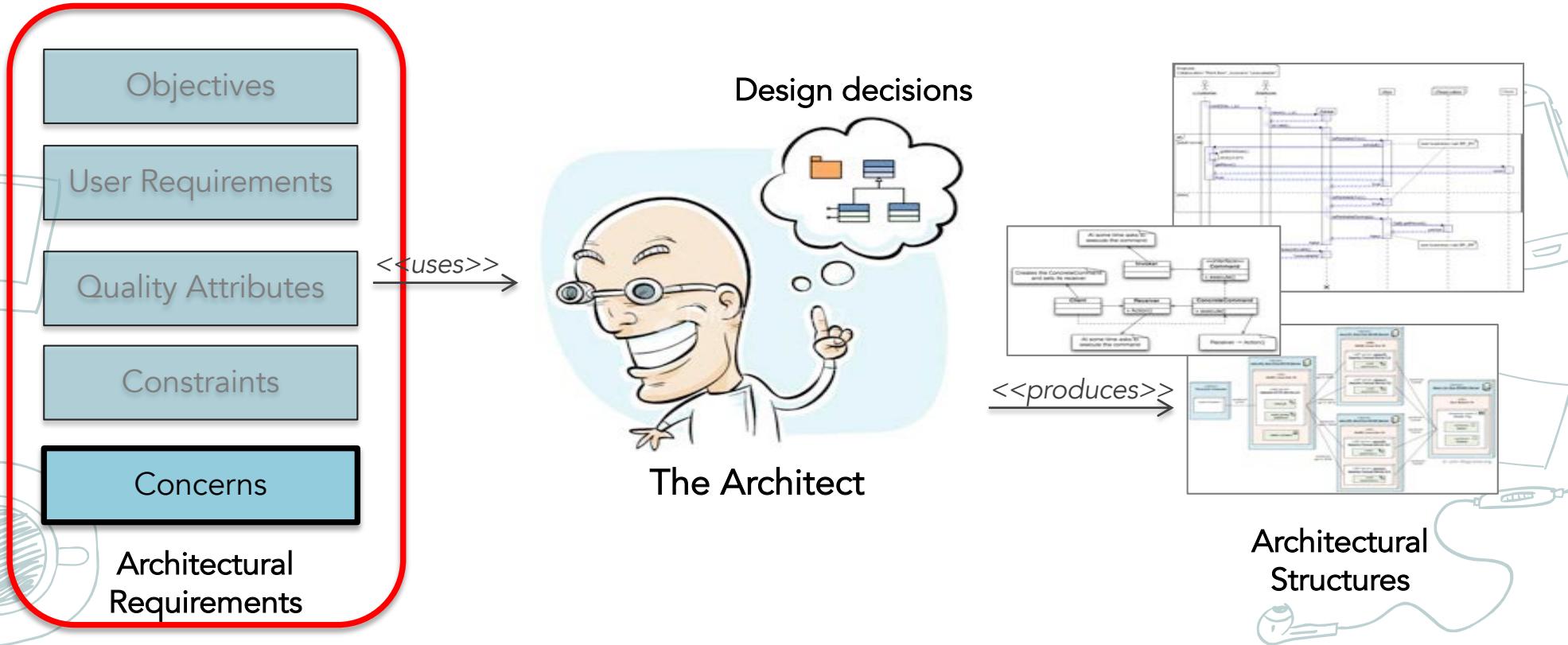




# Business Constraints

- They are mandatory decisions that are already made for the architect before he or she begins the process.
- They **imply the use** of specific technical approaches, solutions, or products.
- Examples often involve: budget, effort, human resources.

# Software Architecture Development Cycle





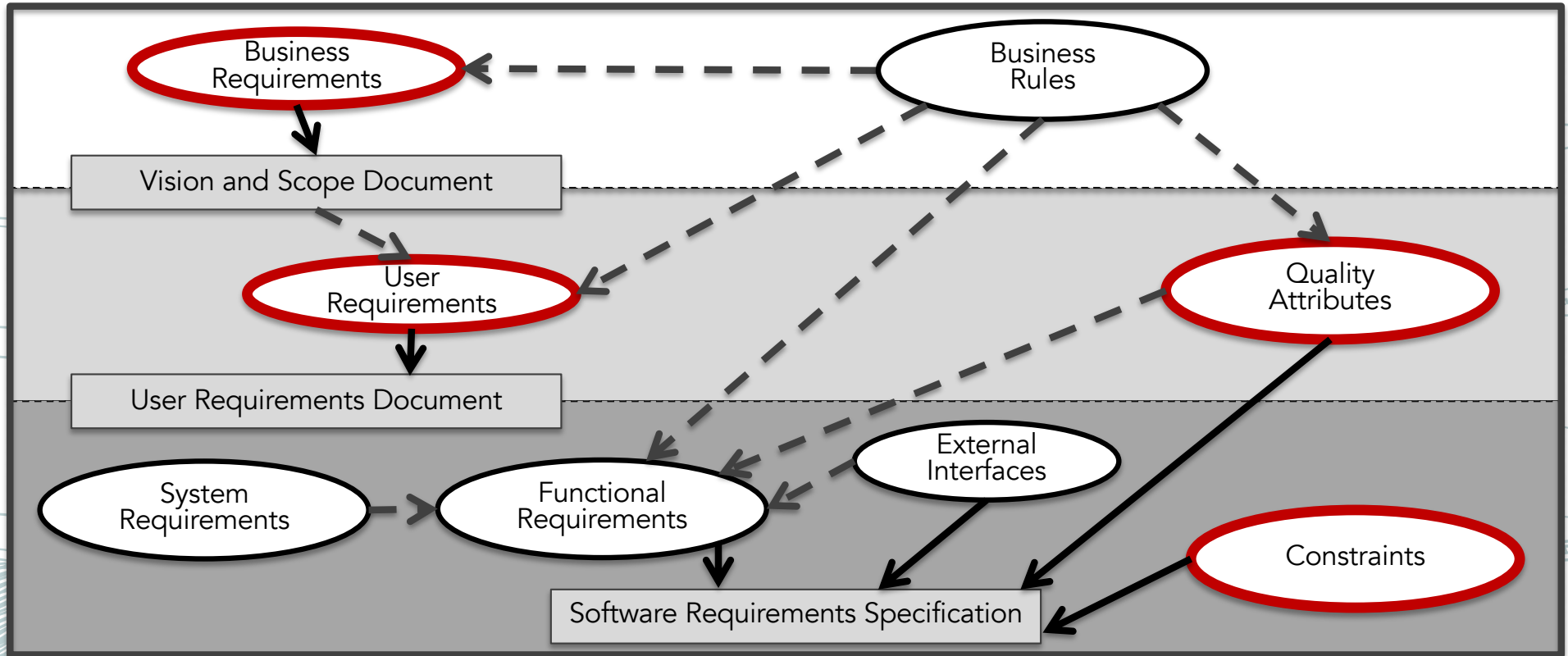


# Concerns

They are additional aspects that need to be considered but which are not expressed as traditional requirements:

- establishing an **initial overall system** structure
- identifying **structures** to support primary **functionality**
- identifying **structures** to support **quality attributes**
- allocation of **modules** to **development teams**
- analyzing **trade-offs** and selecting **technologies** to support



# Requirements Types

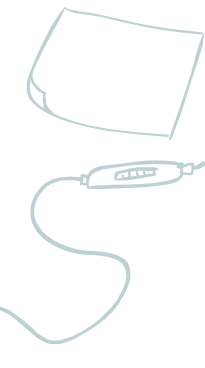
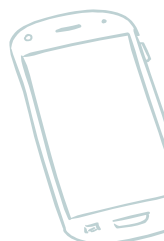


\* From K.E. Wiegers. 2013. *Software Requirements* (3 ed.). Microsoft Press, Redmond, WA, USA.



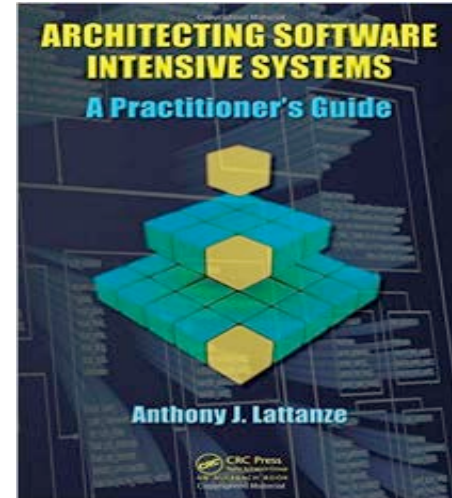
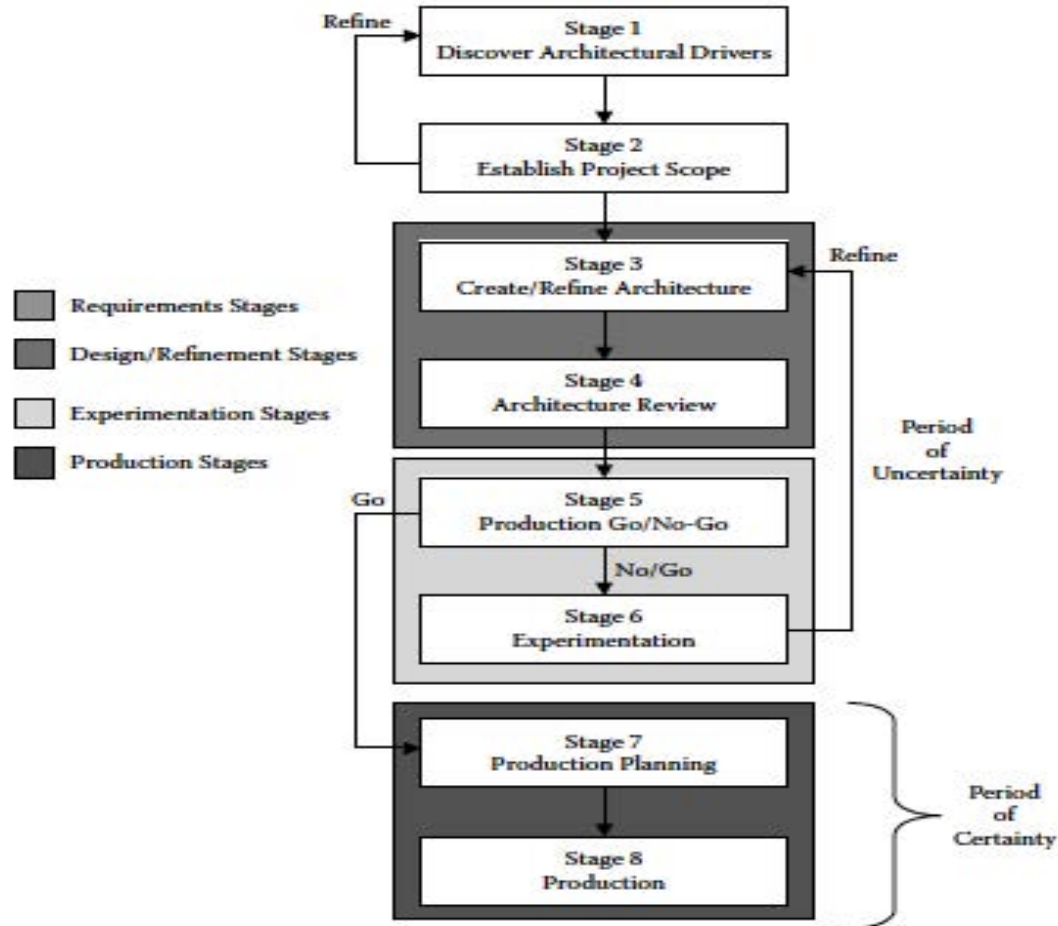
# Outline

- 
- 
1. Software Requirement
  2. Architectural Requirements
  3. Related Methods
  4. Summary



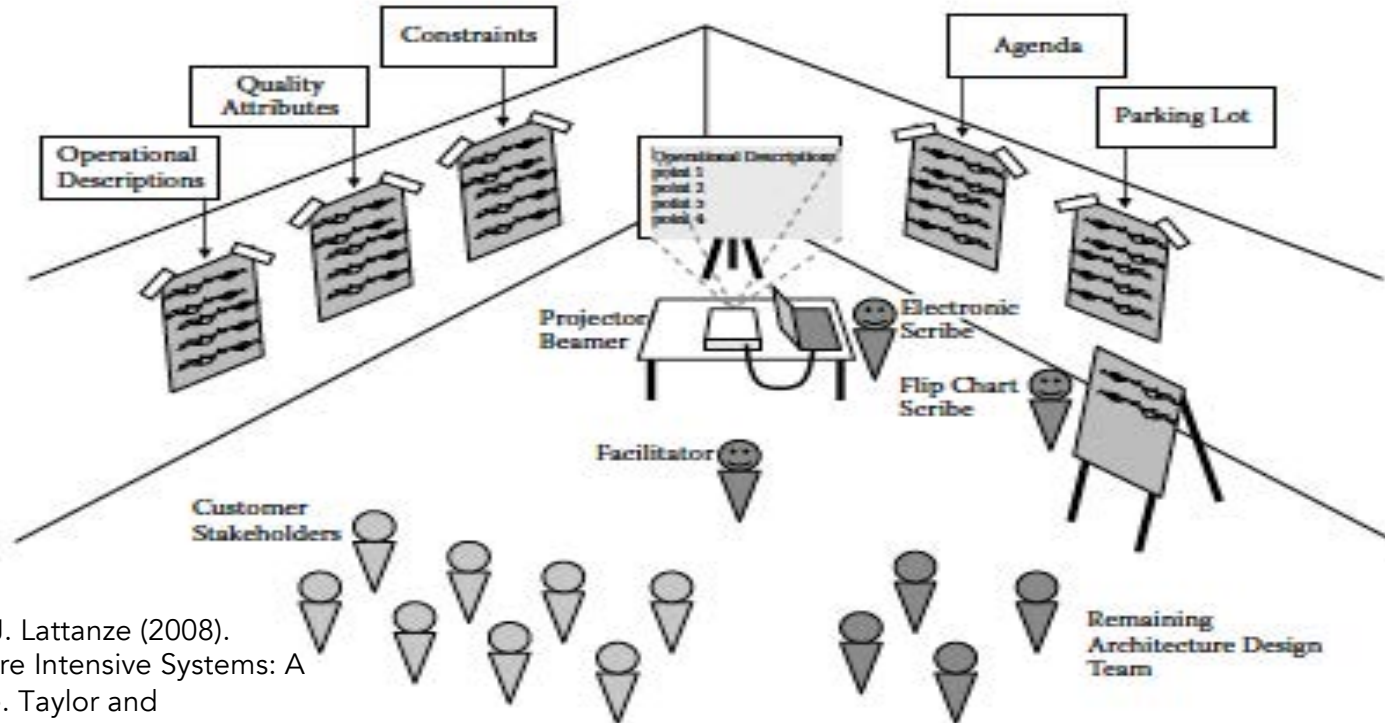
# ACDM

## Architecture centric design method



\* Source: Anthony J. Lattanze (2008). Architecting Software Intensive Systems: A Practitioner's Guide. Taylor and Francis/Auerbach.

# ACDM



\* Source: Anthony J. Lattanze (2008).  
Architecting Software Intensive Systems: A  
Practitioner's Guide. Taylor and  
Francis/Auerbach.

**Figure 8.3** Graphical depiction of an architecture drivers elicitation workshop showing the room setup, attendees, equipment, and artifacts.


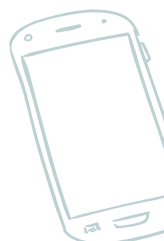
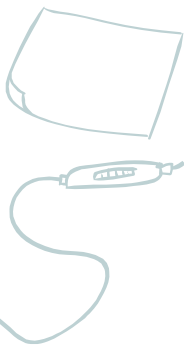



# QAW

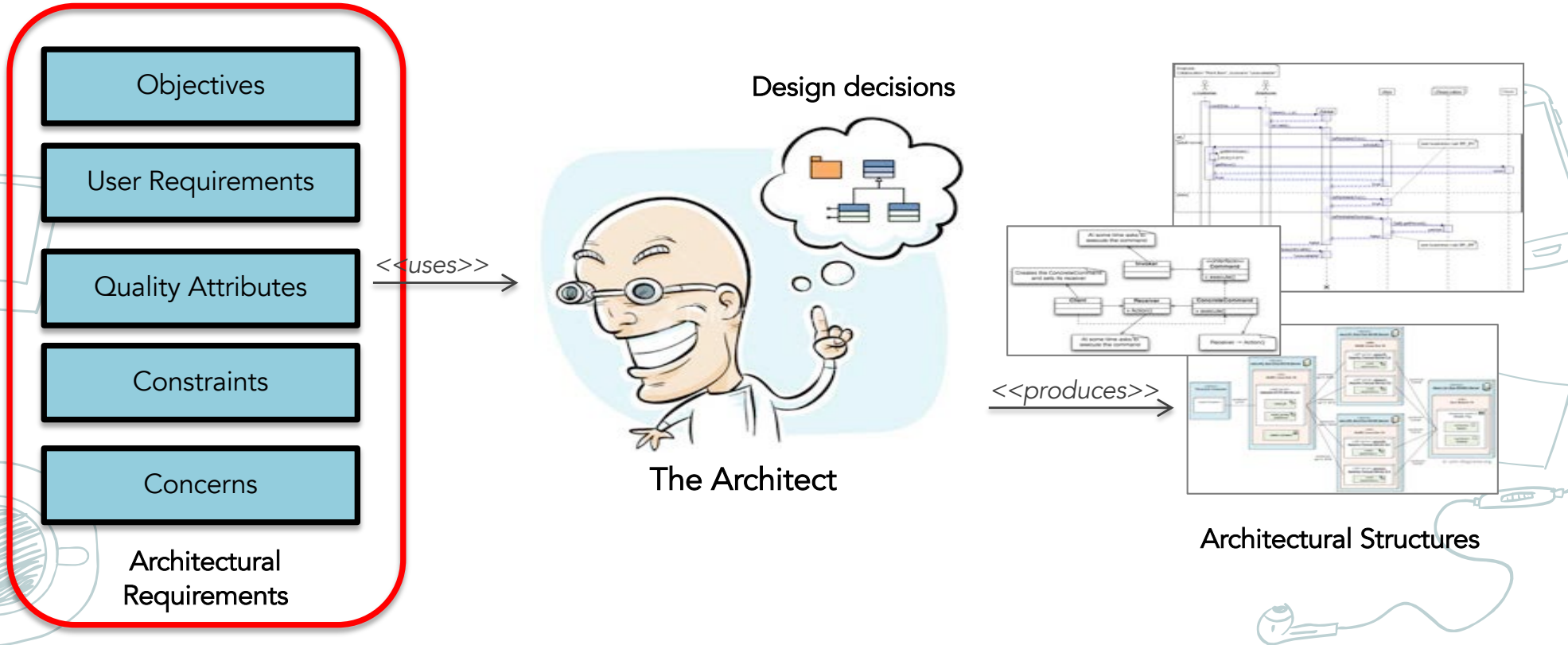
- Proposed by the Software Engineering Institute, SEI.
- It is a facilitated method that engages a diverse group of system stakeholders early in the life cycle to discover the driving quality attributes of a software-intensive system.



# Outline

- 
- 
- 
- 
1. Software Requirement
  2. Architectural Requirements
  3. Related Methods
  4. Summary

# Software Architecture Development Cycle





# Questions? Comments?

