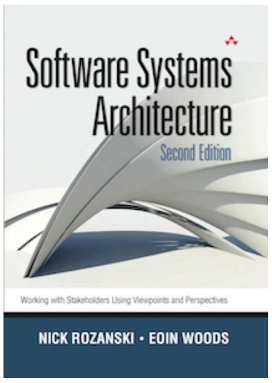
Requirements Engineering and Software Architecture

Fundamentals of Software Architecture

Emmanuel Letier
http://letier.cs.ucl.ac.uk/

Reference

N. Rozanski and E. Woods, *Software Systems Architecture : Working With Stakeholders Using Viewpoints and Perspectives, 2nd Ed.,* Addison-Wesley, 2012 [Chapters 1 to 14]


Software Systems Architecture
Second Edition
Working with Stakeholders Using Viewpoints and Perspectives
NICK ROZANSKI • EOIN WOODS

http://www.viewpoints-and-perspectives.info/

Software Architecture

(Woods & Rozanski, *Software Systems Architecture*, 2011)

A software system's architecture is *the set of principal design decisions* made about the system. It includes decisions about the system's

- externally visible behaviours (functionalities) and quality properties (e.g. performance, availability)
- static structures (its internal design elements and their arrangement)
- dynamic structures (its run-time elements and their interactions)
- implementation and evolution principles

3

Why care about software architecture?

Every system has an architecture, whether or not it is documented and understood

A good architecture makes it easier to

- satisfy the functional and quality requirements
- understand how the software work
- analyze the software properties
- test the software
- maintain and evolve the software

A bad architecture makes all these things much harder, and sometimes impossible

4

Architecture influences qualities

- A same set of functional requirements can be implemented using many different architectures
  – e.g. a small online auction system vs ebay
- Different architectures have different qualities
  – Performance
  – Evolvability
  – Availability
  – Security
  – Cost
- A good architecture is one that successfully addresses the concerns of its stakeholders and, when those concerns are in conflict, balances them in a way that is acceptable to the stakeholders

Architecture as guide rails

- An architecture imposes constraints on what the system's internal elements can and must do

  E.g.
  – components must use a given component for authentication
  – components of layer i can only use components of layer i+1
- Architecture constraints act as guide rails: developers' freedom is purposefully restricted in order to facilitate
  – satisfaction of quality requirements
  – coordination between developers
  – understanding and analysis of the whole system
  – testing and debugging

Architecture descriptions as a guidebook

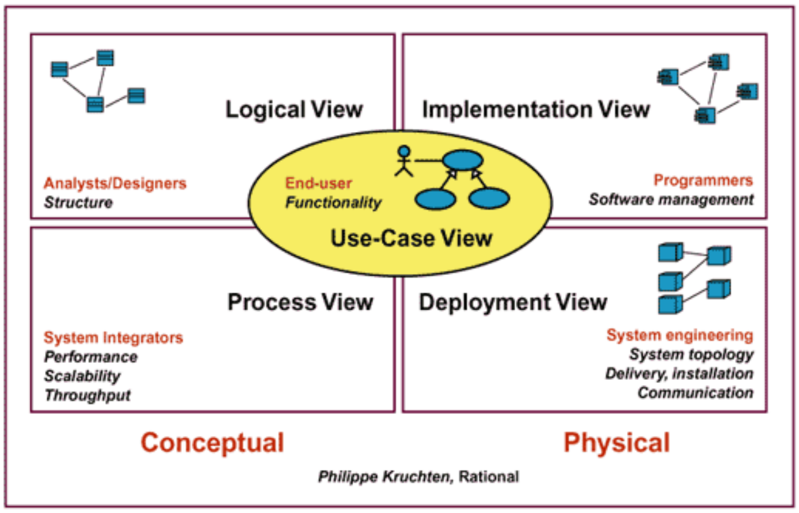
An architecture description is like a guidebook for people who need to work on or with the software code base: clients, developers (current and future), testers, sysadmins, ...

A good architectural description is one that effectively and consistently communicates key aspects of the architecture to the appropriate stakeholders

Different sections (viewpoints) address different stakeholders' concerns

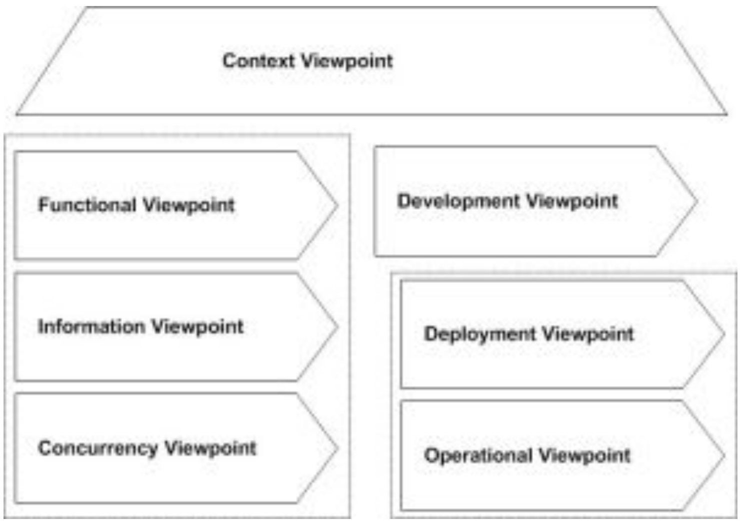
7

Kruchten's 4+1 Architecture Viewpoints (1995)



8

Rozanski and Woods Viewpoints



http://www.viewpoints-and-perspectives.info/home/viewpoints/

Role of the Software Architect

Four main responsibilities

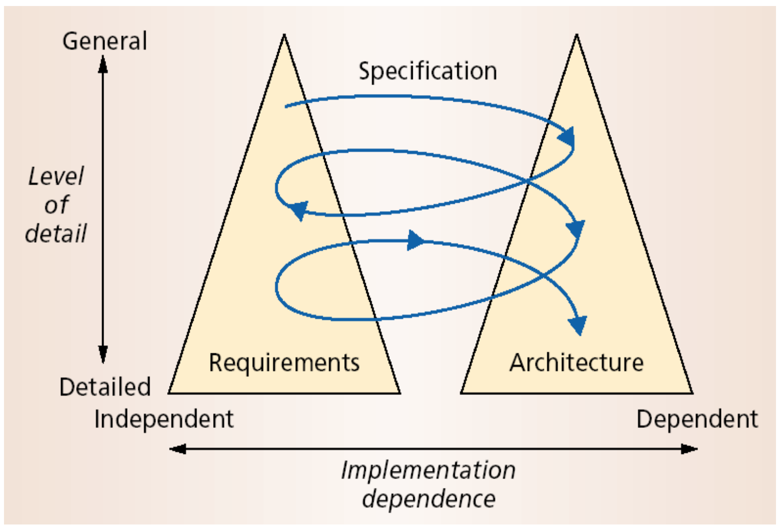
1. identify and engage stakeholders
2. understand and capture their concerns
3. create and take ownership of the architectural description
4. take a leading role in the realisation of the architecture

Rozanski and Woods, Software Systems Architecture, Addison Wesley, 2012

The Architecture Definition Process

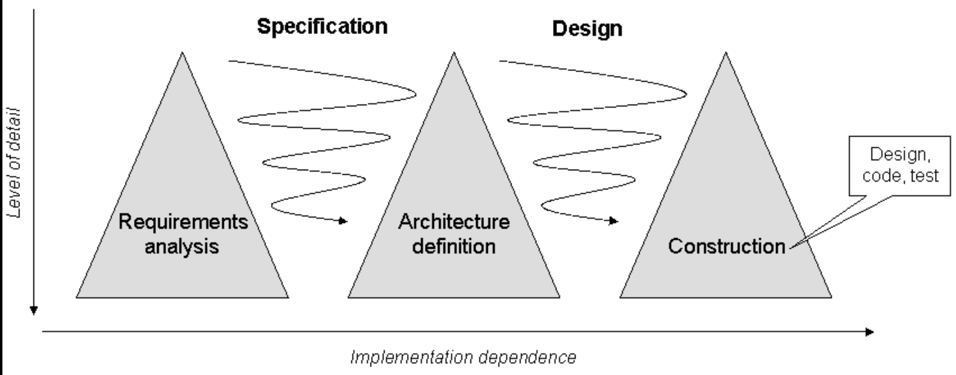
11

The Twin Peaks Model



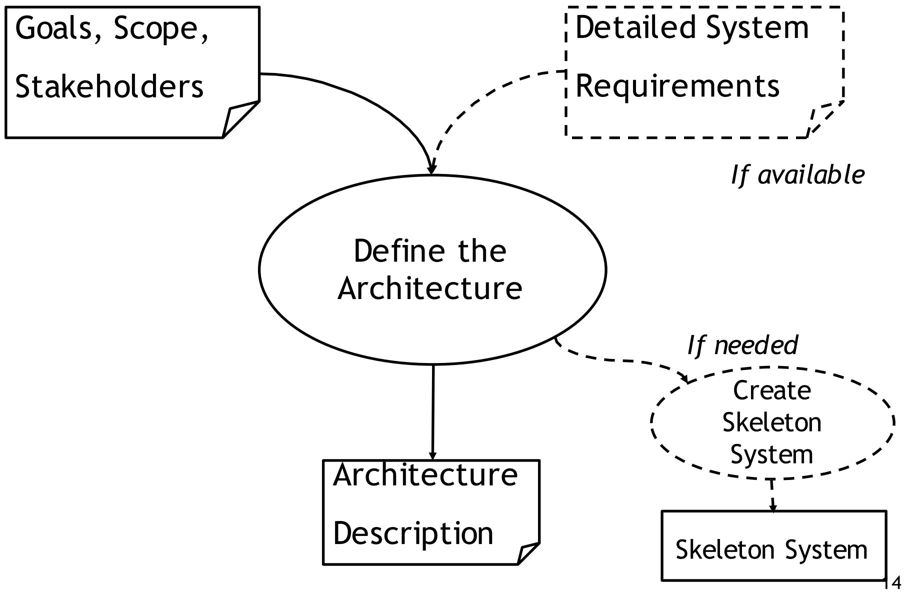
Bashar Nuseibeh, Weaving together requirements and architectures, *IEEE Computer*, 2001

The Three Peaks Model

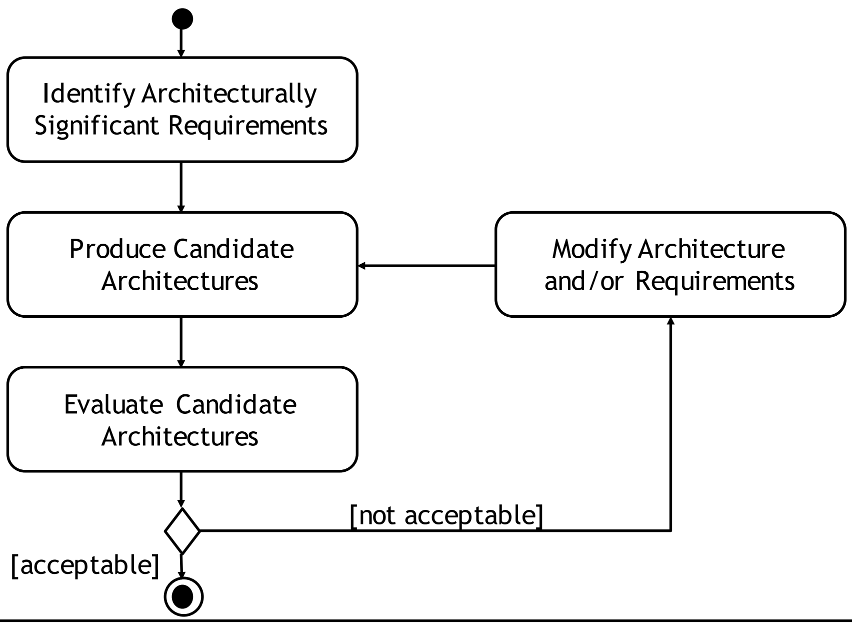


Rozanski, Nick & Woods, Eoin. Software Systems Architecture. Addison-Wesley, 2005.

Context of the Architecture Definition Process



Architecture Definition Process


Identify Architecturally Significant Requirements
Produce Candidate Architectures
Modify Architecture and/or Requirements
Evaluate Candidate Architectures
[not acceptable]
[acceptable]

1. Identify Architecturally Significant Requirements

Architecturally significant requirement (ASR): a requirement that has a significant impact on architectural decisions; the outcome of the architectural decisions would be very different if the requirement was missing or different

By contraposition, a requirement is **not** architecturally significant if the presence or absence of this requirement does not affect the outcome of architectural decisions

Exercise

Which of these requirements for an air pollution monitoring system are architecturally significant?

1. The system shall be able to display air temperature in Celsius or Fahrenheit according to the user preference
2. The system shall record air pollution levels at all 10,000 monitoring stations every 0.1 seconds
3. User Story: search for monitoring stations by postcode

    As a visitor on the air pollution website
    I want to find monitoring stations close to my postcode
    So that I can view past and current pollution levels close to where I live

17

ASR Characteristics

- During requirements elaboration, we can only *predict* whether a requirement will be architecturally significant or not. Predicting architectural significance is difficult and requires judgement and expertise.
- ASR are those than can "break" an architecture
    - A missing core functionality that requires major changes to the existing architecture
    - A missing quality requirement that cannot be met with the existing architecture

18

Heuristics: Likely ASR are those that refer to …

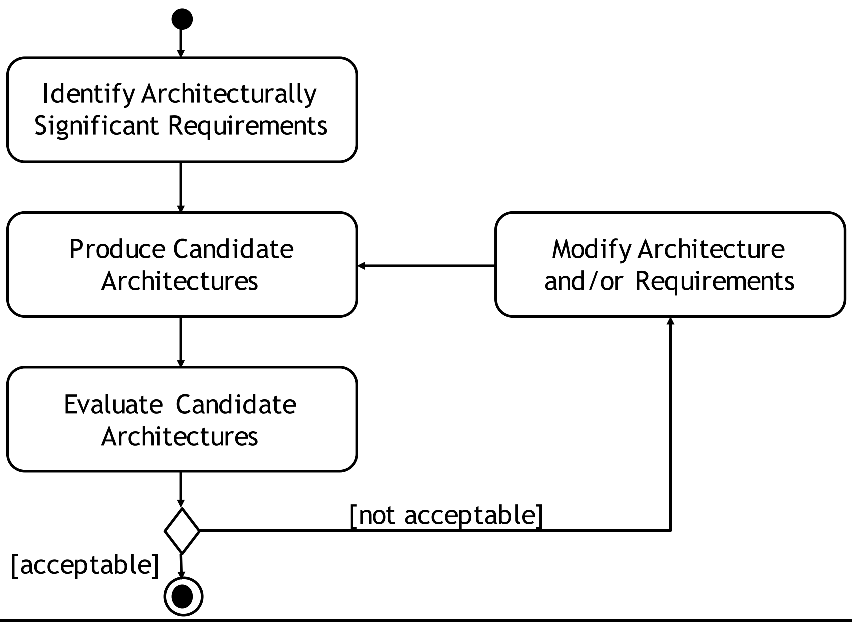
Chen et al. Characterizing Architecturally Significant Requirements. IEEE Software, 2013

1. Core features
   - Features essential to the project's main goals
2. Quality requirements
   - Performance, availability, security, evolution, etc.
3. Constraints
   - Budget and schedule constraints
   - Legacy systems
   - Implementation and technology constraints
4. Application environment
   - Internet, corporate network, embedded hardware, virtual machines, mobile devices, etc.
   - Systems running in different environments often have vastly different architectures

Observations

- For a given system, only a small subset of requirements are architecturally significant
- In many requirements documents, ASR tend to be neglected, described vaguely, or hidden within other requirements

Architecture Definition Process


Identify Architecturally Significant Requirements
Produce Candidate Architectures
Modify Architecture and/or Requirements
Evaluate Candidate Architectures
[not acceptable]
[acceptable]

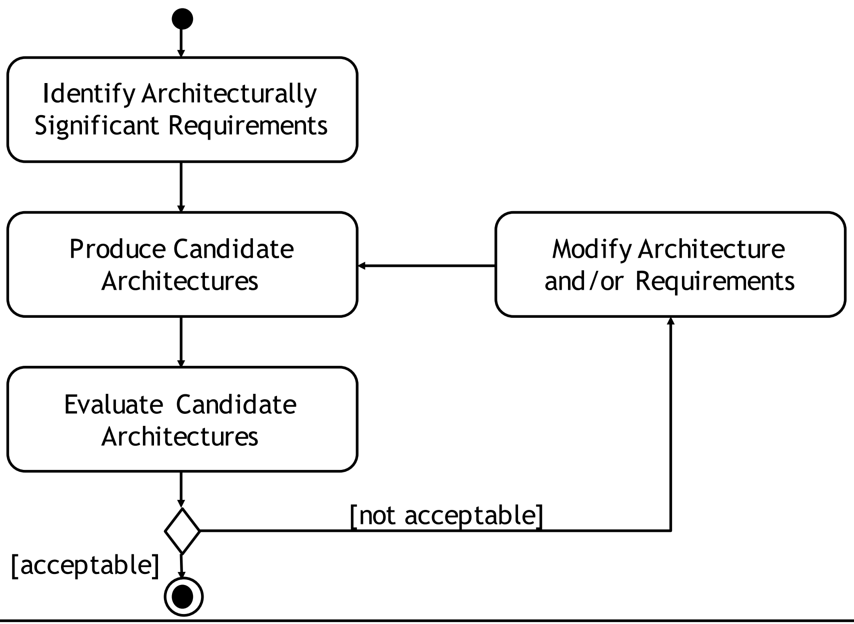
21

2. Produce Candidate Architectures

- Decompose the system into functional elements (a.k.a. components) with well-defined responsibilities
- Identify architectural choices and create models for one or more candidate architectures

22

Architecture Definition Process


Identify Architecturally Significant Requirements
Produce Candidate Architectures
Modify Architecture and/or Requirements
Evaluate Candidate Architectures
[not acceptable]
[acceptable]

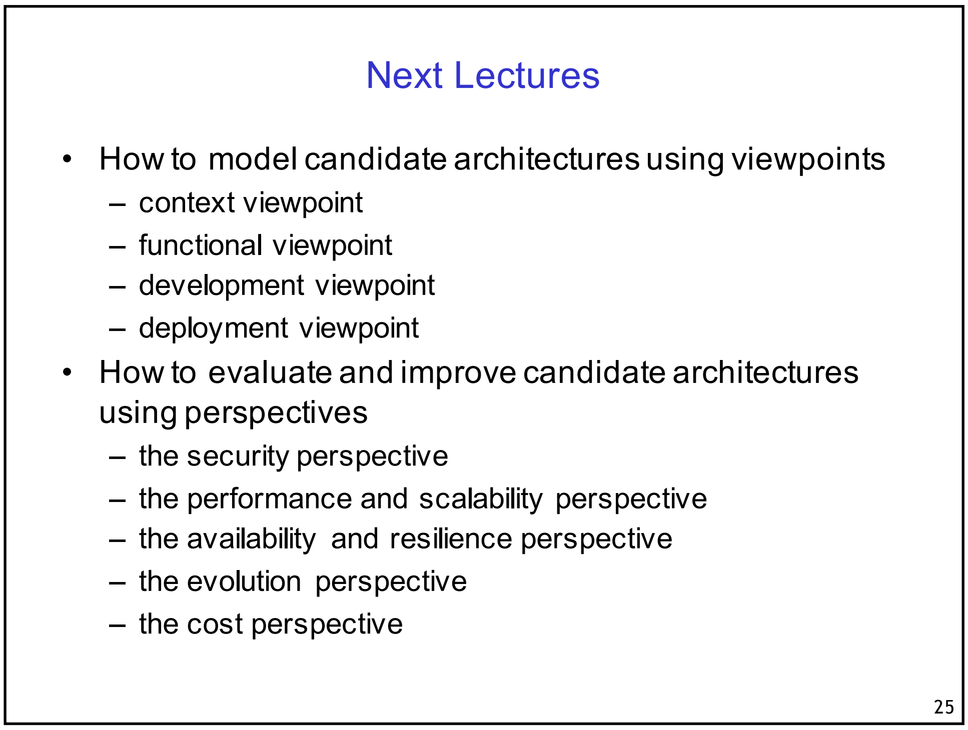
23

3-4. Evaluate and Rework Architecture

Architectural perspectives =

- guidelines for defining perspective-specific requirements (e.g. performance, availability, security, evolution, etc.)
- techniques for evaluating architecture against these perspective-specific requirements
- architecture tactics for modifying an architecture to satisfy the perspective-specific requirements

24



Next Lectures

- How to model candidate architectures using viewpoints
  - context viewpoint
  - functional viewpoint
  - development viewpoint
  - deployment viewpoint
- How to evaluate and improve candidate architectures using perspectives
  - the security perspective
  - the performance and scalability perspective
  - the availability and resilience perspective
  - the evolution perspective
  - the cost perspective

25