# Software Architecture Concepts

Prof. Dr. Everton Cavalcante

http://www.dimap.ufrn.br/~everton/

## Goals

- To introduce the fundamental concepts about software architectures
- To highlight the central role played by software architectures in the development of software systems

# 5 Fundamental Questions

## Objetivos

What is (and what is not) software architecture?

Is there space for Software Architecture when adopting agile methodologies?

Why software architectures are important?

What is the relationship among software requirements and architectures?

What is the role of a software architect?

## O que é (e o que não é) arquitetura de *software*?

What is (and what is not) software architecture?

Is there space for Software Architecture when adopting agile methodologies?

Why software architectures are important?

What is the relationship among software requirements and architectures?

What is the role of a software architect?

## Software architectures

Software architectures are within the context of
software-intensive systems

**Software-intensive system**

Any system where software essentially influences the design, construction, deployment, and evolution of the system as a whole

IEEE Std 1471-2000. **IEEE Recommended Practice for Architectural Description of Software-Intensive Systems.** New York, NY, USA: IEEE, October 2000

**IEEE Std 1471-2000**

**IEEE Recommended Practice for Architectural Description of Software-Intensive Systems**

## Software Architecture as an engineering discipline

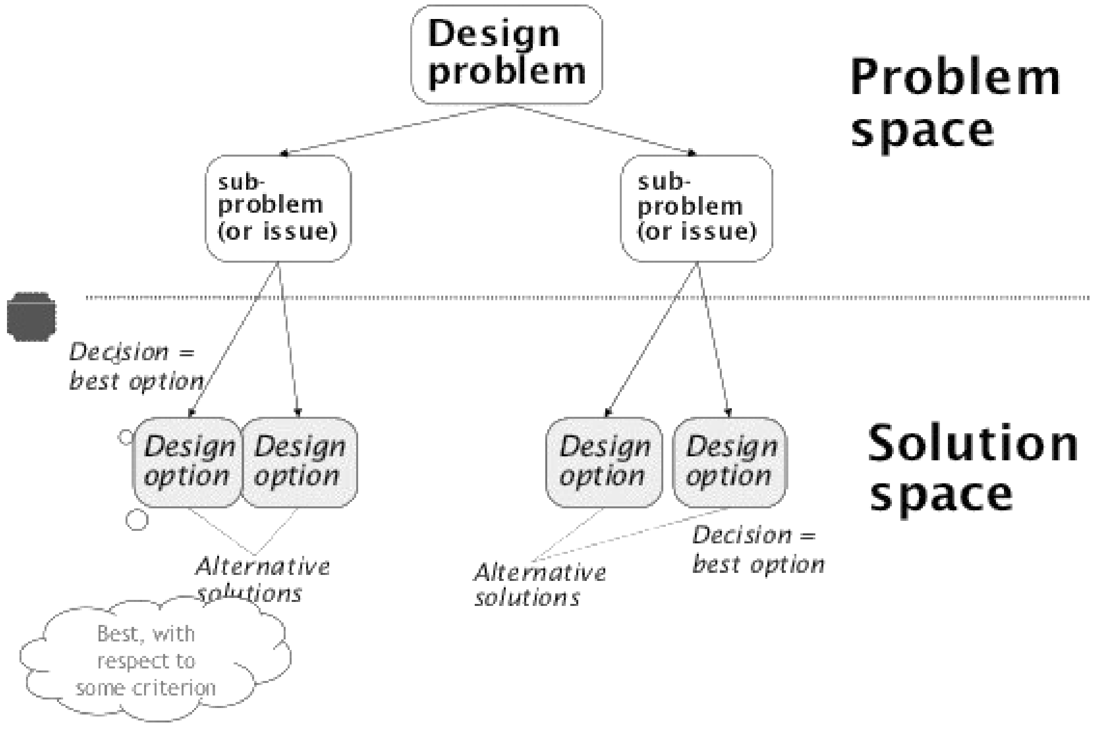
- Software Architecture is essential for the development of software-intensive systems enabling to reason about system properties very early in the development lifecycle
- Understanding structure, behavior, and properties of complex software systems
- Systematic approach to design software systems encompassing principles, theories, methods, techniques, and tools supporting it

## Software Architecture as an engineering discipline

The issue is on how to organize a system to simultaneously

- make the suitable decisions
- provide the required functionalities (functional requirements)
- guarantee the required quality of service (non-functional requirements)

## Software Architecture as an engineering discipline



Architecting is largely about making decisions

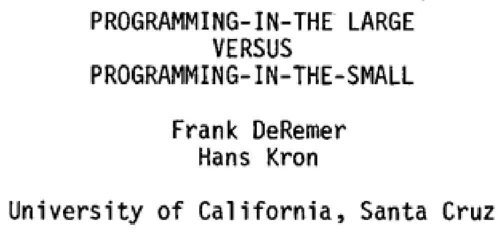
– If the decisions are not the right ones, they potentially will impact how the system operates and/or result in failures of many types

– If one starts with the wrong architecture, the software is necessarily going to be unsuccessful and failed

## Software Architecture as an engineering discipline

**1960s**
Comparison between software design and (civil) architecture drawn to manage complexity

**Mid-1970s**
The idea of "Programming-in-the-Large" (organizing a system as a collection of modules) lays the initial bricks for the need of architecting software



PROGRAMMING-IN-THE LARGE
VERSUS
PROGRAMMING-IN-THE-SMALL

Frank DeRemer
Hans Kron

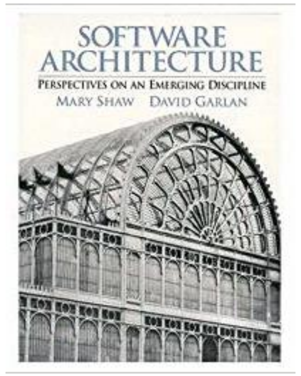
University of California, Santa Cruz

**Mid-1980s**
Early attempts to capture and explain software architectures, but with imprecision and disorganization

# Software Architecture as an engineering discipline

**Early 1990s**
The term “software architecture” becomes prevalent as a Software Engineering discipline

**1996**
First book on Software Architecture (by Mary Shaw and David Garlan)



**2000**

**IEEE Std 1471-2000**

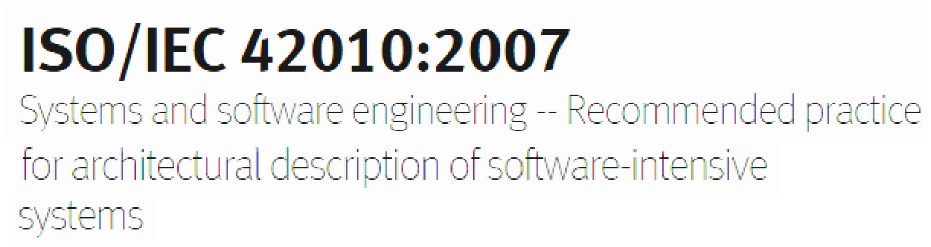
**IEEE Recommended Practice for Architectural Description of Software-Intensive Systems**

## Software Architecture as an engineering discipline

2000s

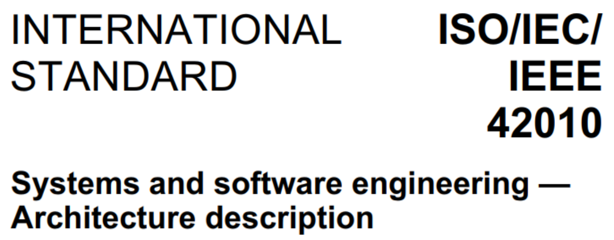
Several books, journals, and conferences focusing on Software Architecture arise

2007


**ISO/IEC 42010:2007**

Systems and software engineering -- Recommended practice for architectural description of software-intensive systems

2011


INTERNATIONAL STANDARD

**ISO/IEC/ IEEE 42010**

**Systems and software engineering — Architecture description**

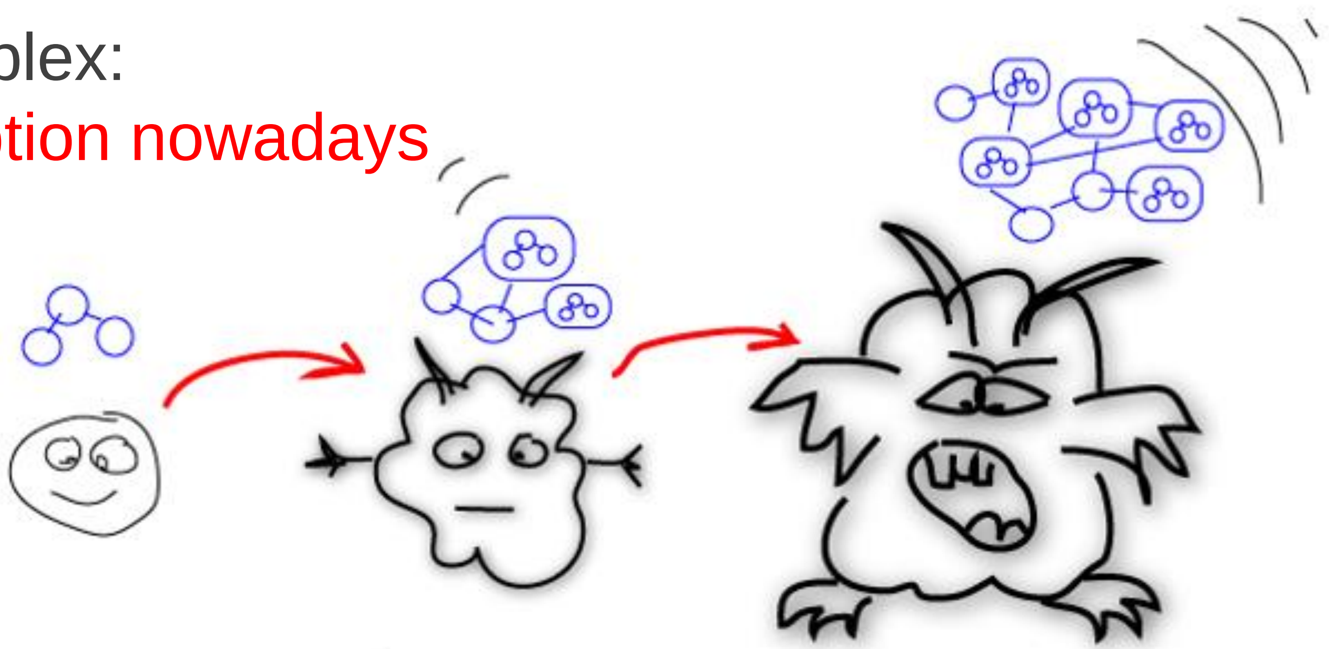
## Software Architecture as an engineering discipline

Software Architecture is closely related to complexity management

– Inherent complexity of software development

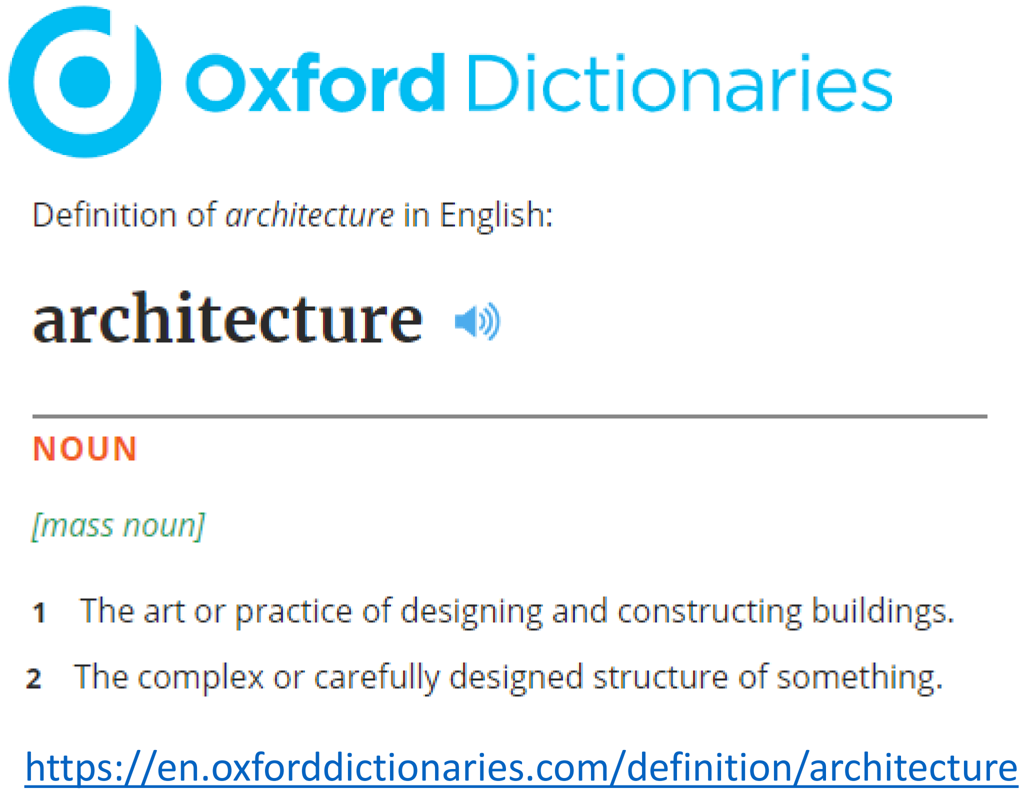
– Demand for high-quality systems

– Software systems are large, complex:
this is the norm rather than exception nowadays



COMPLEXITY

## Software architecture as an artifact


Oxford Dictionaries

Definition of *architecture* in English:

**architecture**

NOUN

*[mass noun]*

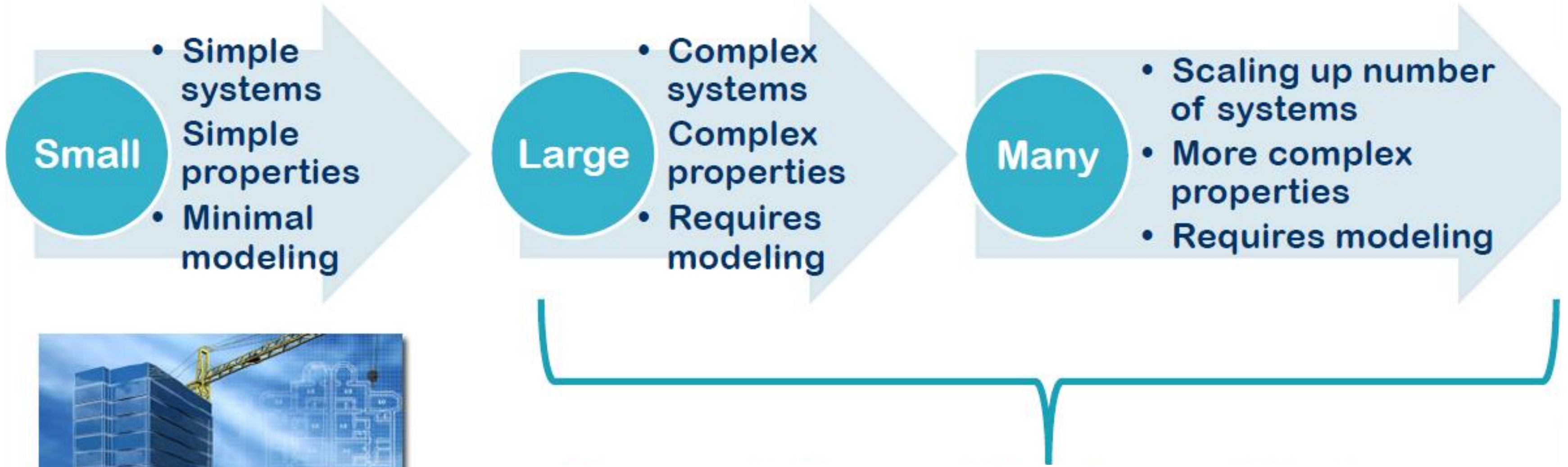
1 The art or practice of designing and constructing buildings.

2 The complex or carefully designed structure of something.

https://en.oxforddictionaries.com/definition/architecture

The need for architecting arises in a shift from simple/small things to complex/large things, thus becoming a real concern

## Software architecture as an artifact



**The needs for modeling the architecture increases with the complexity of the system in terms of its parts, their relationships and the properties to be enforced**

## Software architecture as an artifact

The architecture of a software system is a metaphor, analogous to the architecture of a building as they are similar (though not identical), requiring

– fulfilling customers' needs, expectations, and exigences

– blueprints and multiple perspectives

– specialized activities

– good quality materials in construction

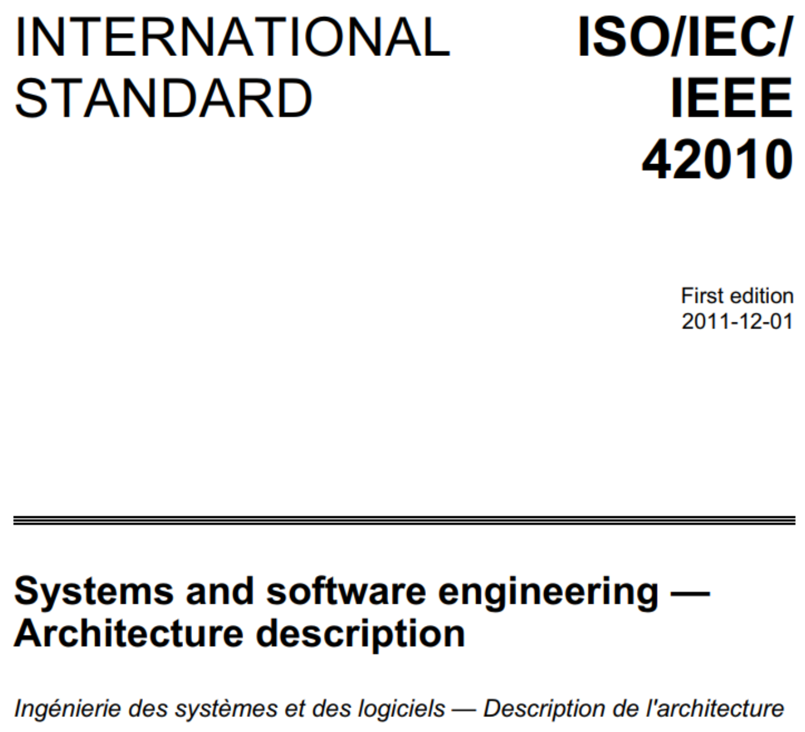
– being useful at the end

## Software architecture as an artifact

**Software architecture**

The fundamental conception of a system in its environment embodied in its **elements**, **relationships**, and in the **principles of its design and evolution**

ISO/IEC/IEEE 42010. **Systems and software engineering – Architecture description.** Geneva, Switzerland: ISO, December 2011



INTERNATIONAL STANDARD

**ISO/IEC/ IEEE 42010**

First edition
2011-12-01

**Systems and software engineering — Architecture description**

*Ingénierie des systèmes et des logiciels — Description de l'architecture*

## Software architecture as an artifact

A GENERAL DEFINITION OF SWA

Set of principal design decisions about the system

© Alessandro Orso

Blueprint of a software system

- Structure
- Behavior
- Interaction
- Non-functional properties

## Software architecture as an artifact

### Software architectures materialize important concerns

| | |
|---|---|
| Structure | organization of elements and their relationships for executing the functionalities while satisfying the properties of the system |
| Behavior | detailed specification of activities allocated to the elements towards providing the functionalities of the system |
| Properties | properties (typically related to non-functional requirements and constraints to be enforced) to be satisfied by the system before and after its construction |
| Evolution | directions on how to modify (maintain/ evolve) the system in future once it has built |
| Decisions | designers' intentions and knowledge about system structure and behavior thereby providing a defense against design decay as a system ages |

## Software architecture as an artifact

– Software architectures are high-level abstract artifacts that select certain details and suppresses others
(e.g. coding or internal implementation)

– Every software system has a software architecture

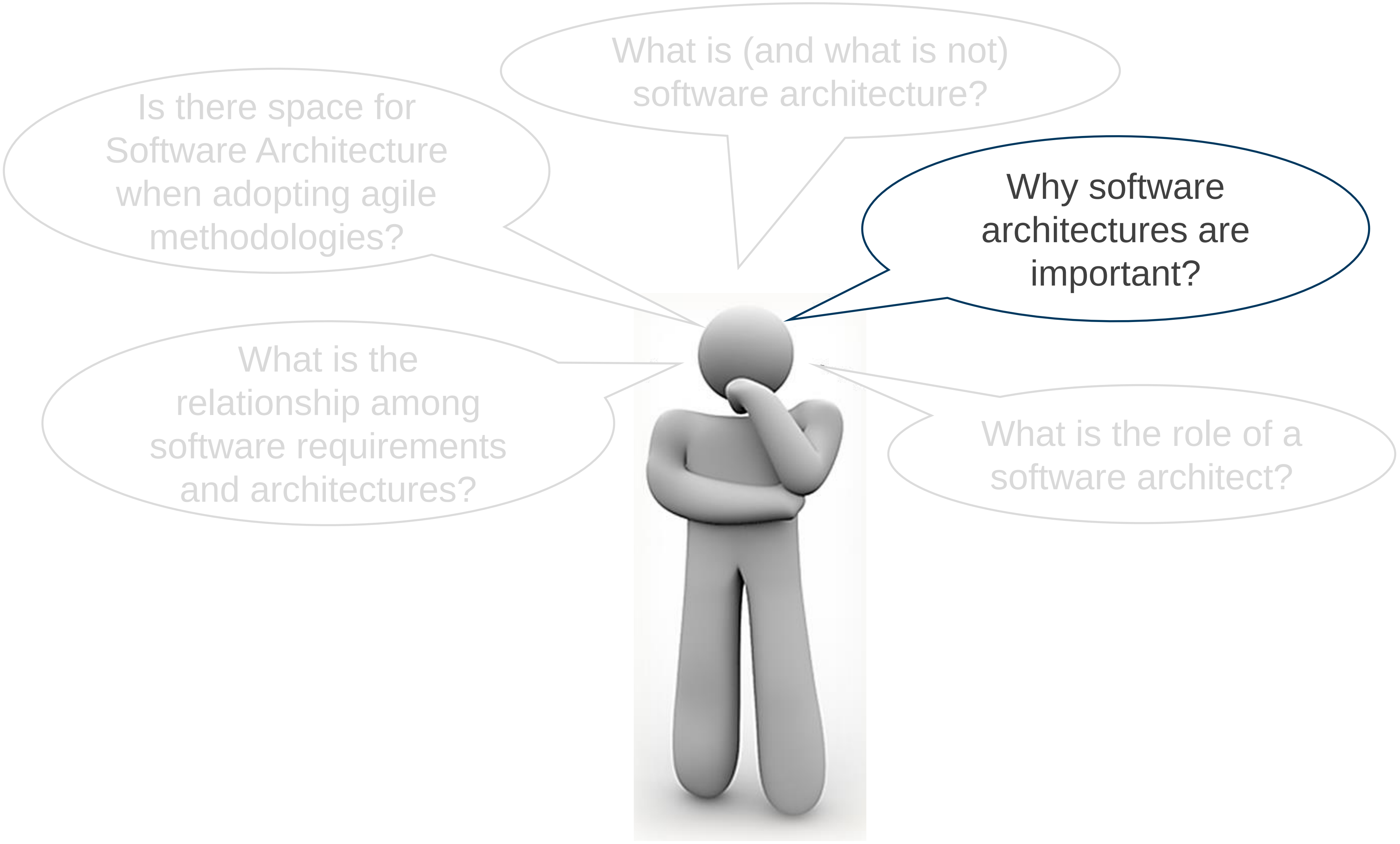
- even if it has not been explicitly conceived since it is independent from its representation

– Software architectures have a temporal aspect

- Design decisions are made, unmade, and changed over a system's lifetime

A system could be successful and yet poorly architected

## Por que arquiteturas de *software* são importantes?


What is (and what is not) software architecture?
Is there space for Software Architecture when adopting agile methodologies?
Why software architectures are important?
What is the relationship among software requirements and architectures?
What is the role of a software architect?

## The importance of software architectures

– Software architectures are recognized to be the backbone of any successful software-intensive system as they contribute to the achievement of both business goals and quality requirements
  - Poor architecture can make these aims fail

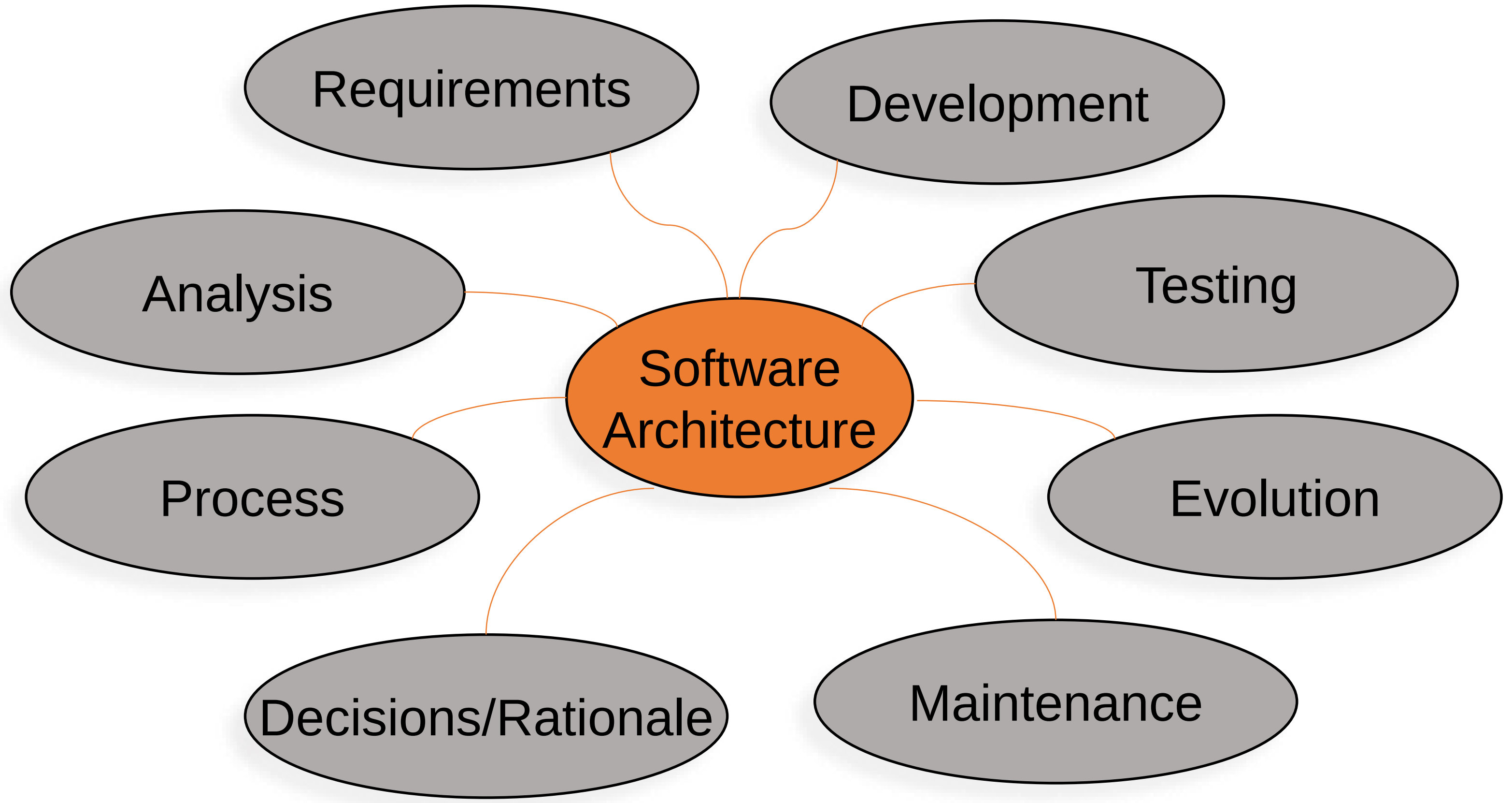
– Software architectures are manageable, meaningful abstractions of the system under development

## The importance of software architectures

Applied throughout the software life cycle, good architectural practice has the potential of

- increasing the understandability of the system and the development process used to create it
- facilitating the communication with stakeholders, contributing to a system that better fulfills their needs
- ensuring the satisfaction of requirements
- reducing the overall cost and risk of the software development process even without building the software itself

## The importance of software architectures



© Thais Batista

## The importance of software architectures

A software architecture can be used as a relevant artifact in

– requirements specification

– system design supported by early decisions that may impact system development, deployment, and maintenance

– system analysis to verify if it fulfills the stakeholders’ needs even before its construction

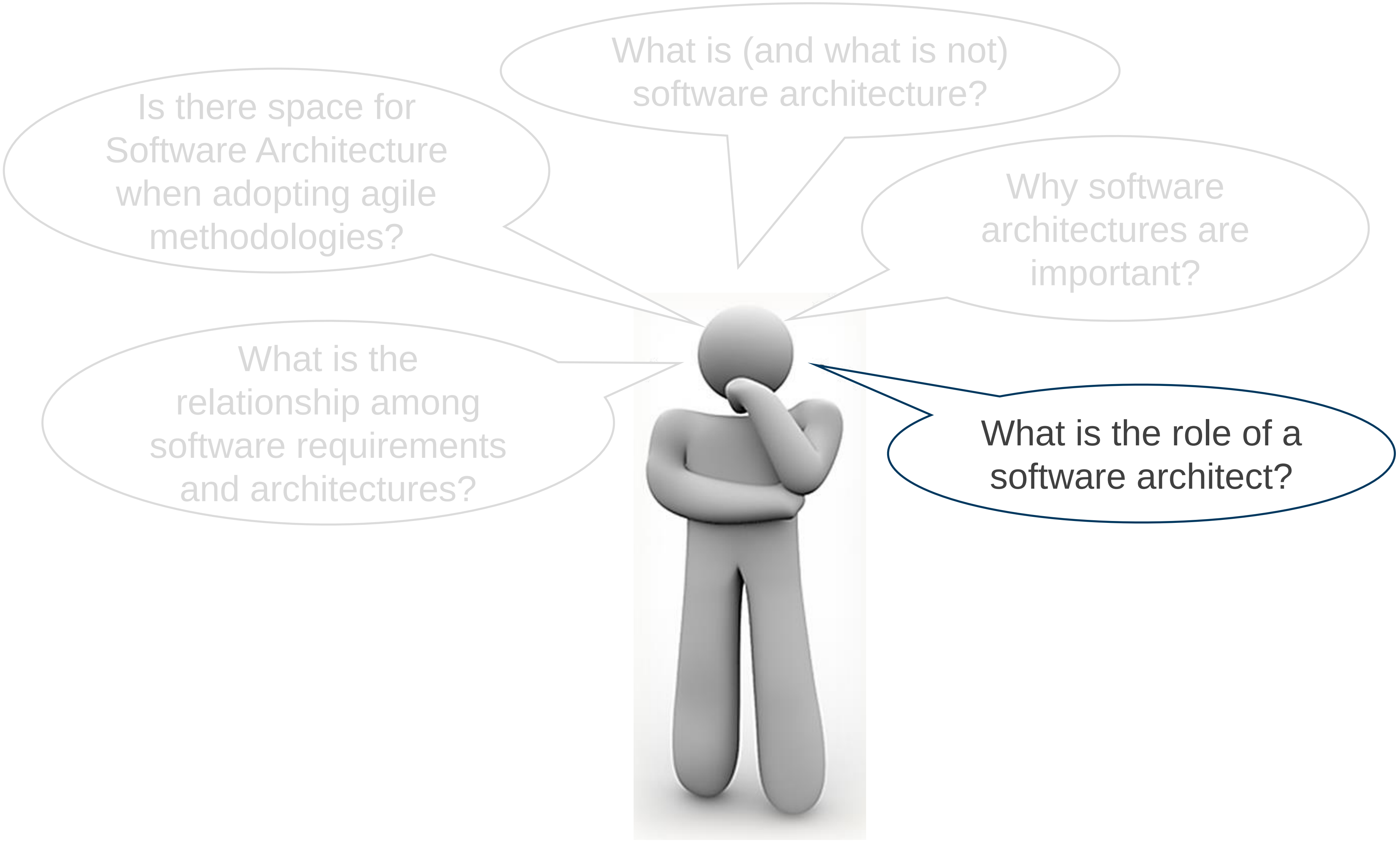
– successive refinements towards implementation (to also constrain it)

– reuse of elements and decisions as a transferrable model

– maintenance and evolution

– runtime adaptation

## Qual o papel de um arquiteto de *software*?


Is there space for Software Architecture when adopting agile methodologies?
What is (and what is not) software architecture?
Why software architectures are important?
What is the relationship among software requirements and architectures?
What is the role of a software architect?

## The role of a software architect

Four main core activities to be performed by a software architect

| | |
|---|---|
| Analysis | Activity of understanding the environment in which a proposed system(s) will operate and determining the requirements for the system |
| Synthesis | Activity of creating the architecture |
| Evaluation | Activity of determining how well the current design satisfies the derived requirements |
| Evolution | Activity of maintaining and adapting an existing software architecture to meet changes while preserving design |

## The role of a software architect

### Other important supporting activities

| Knowledge management and communication | Design reasoning and decision | Documentation (modeling) | Tutoring |
|---|---|---|---|
| Exploring and managing knowledge that is essential to design a software architecture | Evaluating design decisions<br>• gathering and associating decision contexts<br>• formulating design decision problems<br>• finding solution options<br>• evaluating trade-offs | Recording the design generated during the software architecture process | Guiding and assisting both development and testing teams |

## The role of a software architect

Software architects may have to deal with different stakeholders in and out of the project

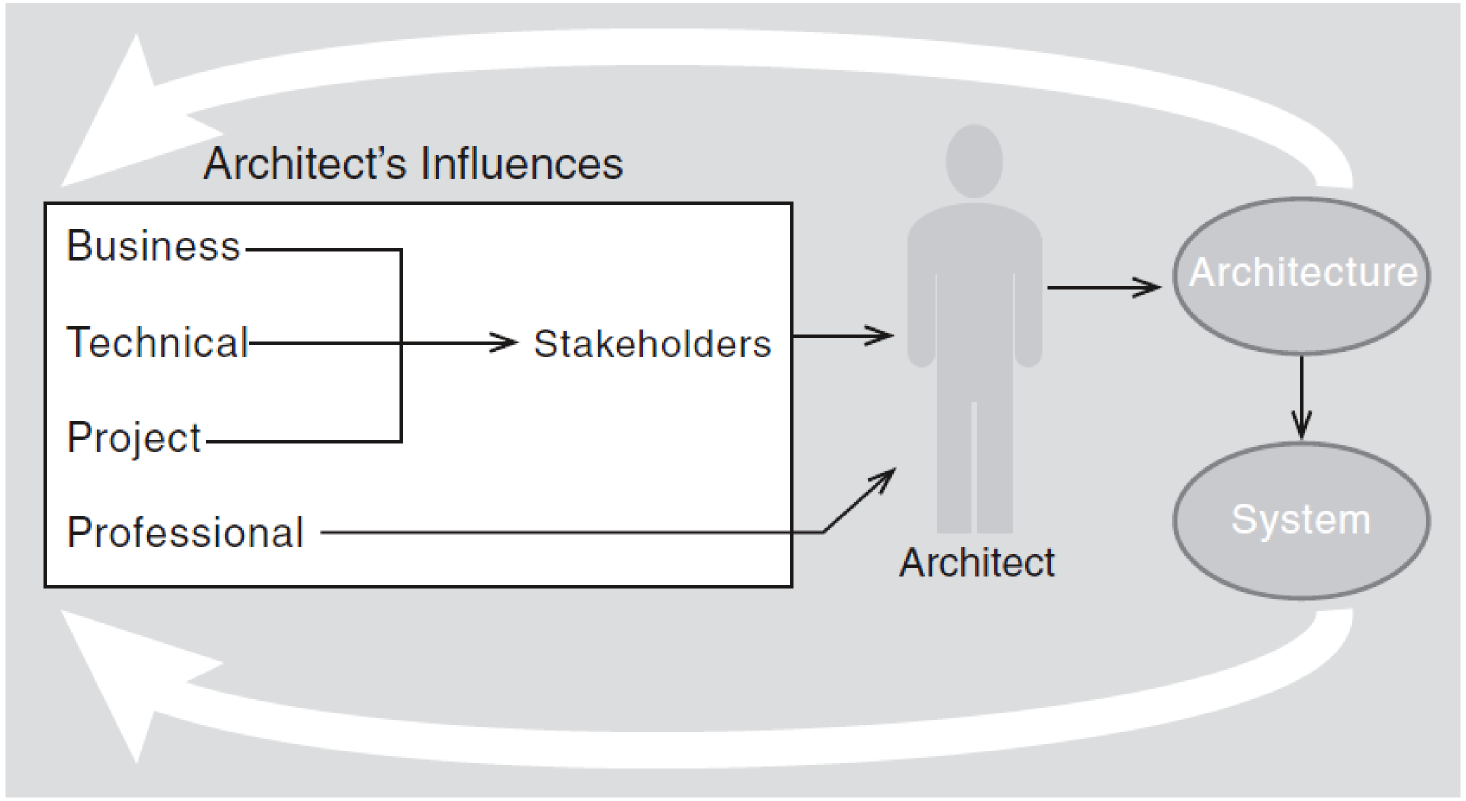
– each one with different (and potentially conflicting) concerns and goals for the software system and its architecture

## The role of a software architect

The architect’s role may be influenced by different contexts, but it indeed influences those contexts

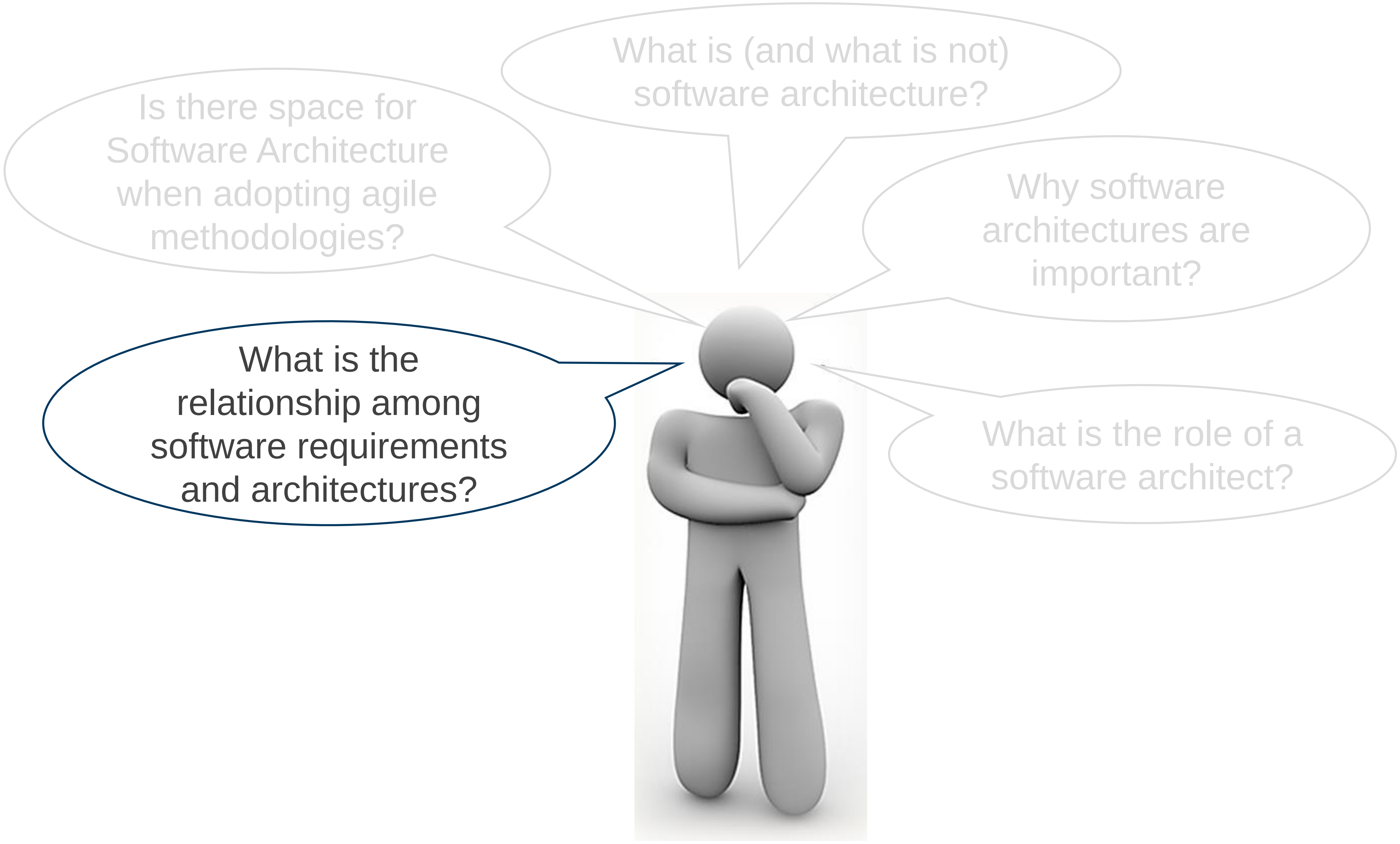
- Technical context: achievement of quality attribute requirements
- Project context: the architecture should be the core of the development activities and implementation must conform to the architecture
- Business context: the system created from the architecture must satisfy the business goals of a wide variety of stakeholders
- Professional context: some skills and knowledge are required to be an architect and there are certain duties to be perform as an architect

## The role of a software architect



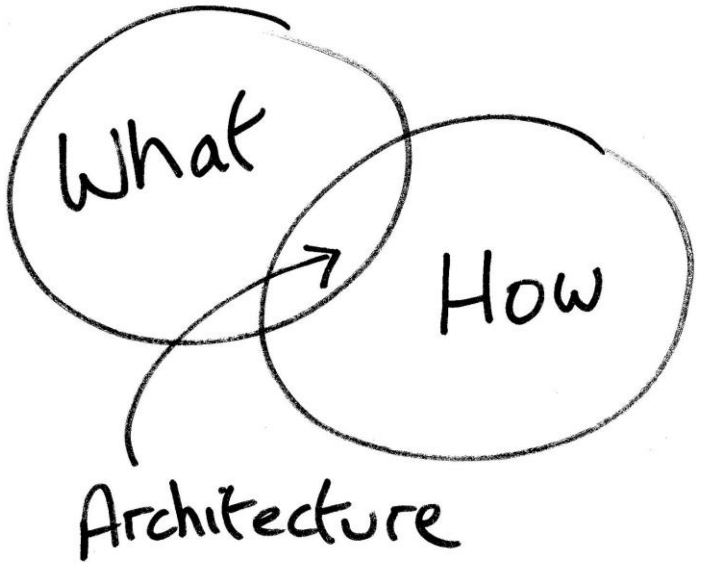
Len Bass, Paul Clements, Rick Kazman. **Software architecture in practice – 3rd ed**. USA: Addison-Wesley/Pearson Education, Inc., 2013

## Qual o relacionamento entre requisitos e arquiteturas de *software*?


What is (and what is not) software architecture?
Is there space for Software Architecture when adopting agile methodologies?
Why software architectures are important?
What is the relationship among software requirements and architectures?
What is the role of a software architect?

## The requirements–architecture relationship

- The architecture is a bridge between (often abstract) business goals and the final (concrete) resulting system
- Requirements Engineering and Software Architecture are complementary approaches
  - Requirements Engineering addresses the "problem space" or the "what"
  - Software Architecture targets the "solution space" or the "how"



## The requirements–architecture relationship

There is a synergy between Requirements Engineering and Software Architecture

– Inputs (goals, constrains, etc.) are usually ill-defined and only get discovered or better understood as the architecture starts to emerge

– The choice of required behavior for a problem impacts the architecture of the solution that addresses that problem while the architectural design may impact the problem and introduce new requirements

– There must be some sort of traceability between software elements and requirements

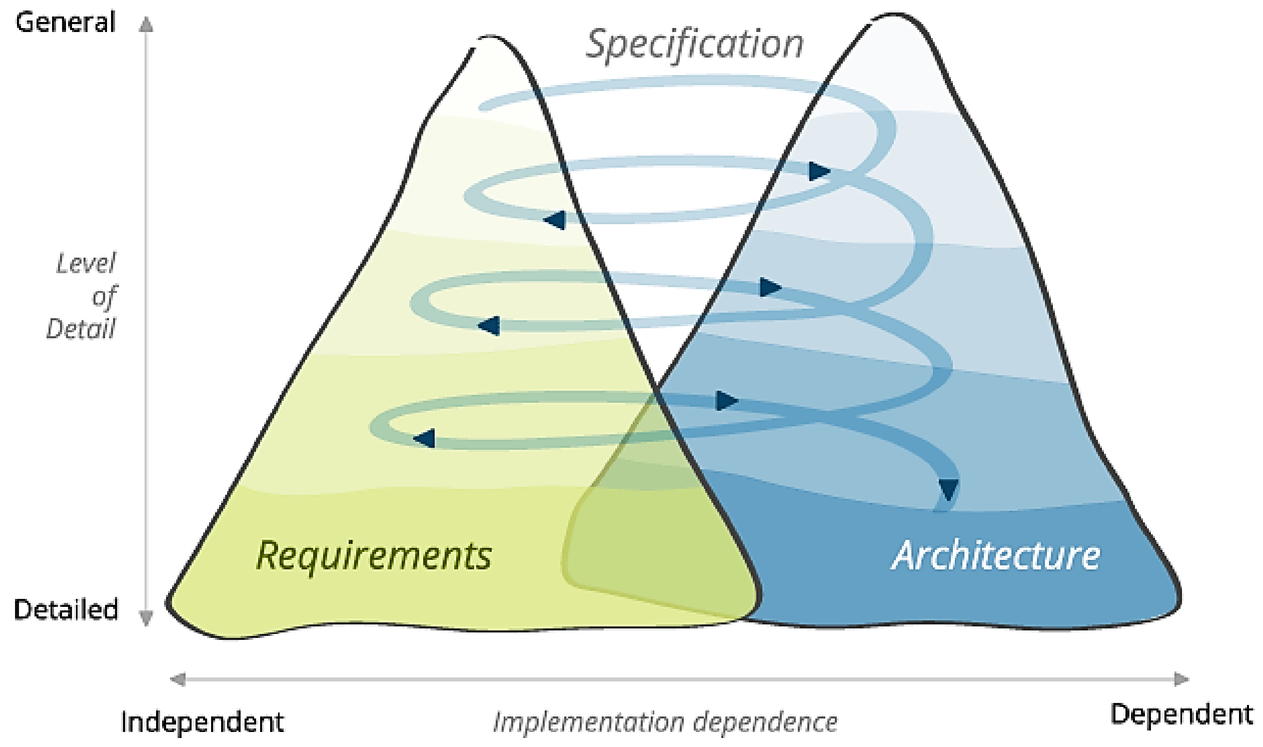
## The requirements–architecture relationship

Software architectures properly fill the gap between requirements for a system and its implementation

- Developers can use software architectures as a common ground towards an implementation ensured to fulfill requirements
- Software architectures guide design and evolution of the system as they materialize decisions, structure, behavior, and requirements conformance
- Software architectures can undergo successive refinements (ideally automated in part or in full) from an abstract model (closer to requirements) to a more concrete model (closer to code)

## The requirements–architecture relationship

### The Twin Peaks Model



Bashar Nuseibeh. **Weaving together requirements and architectures.** Computer, vol. 34, no. 3, March 2001, pp. 115-119

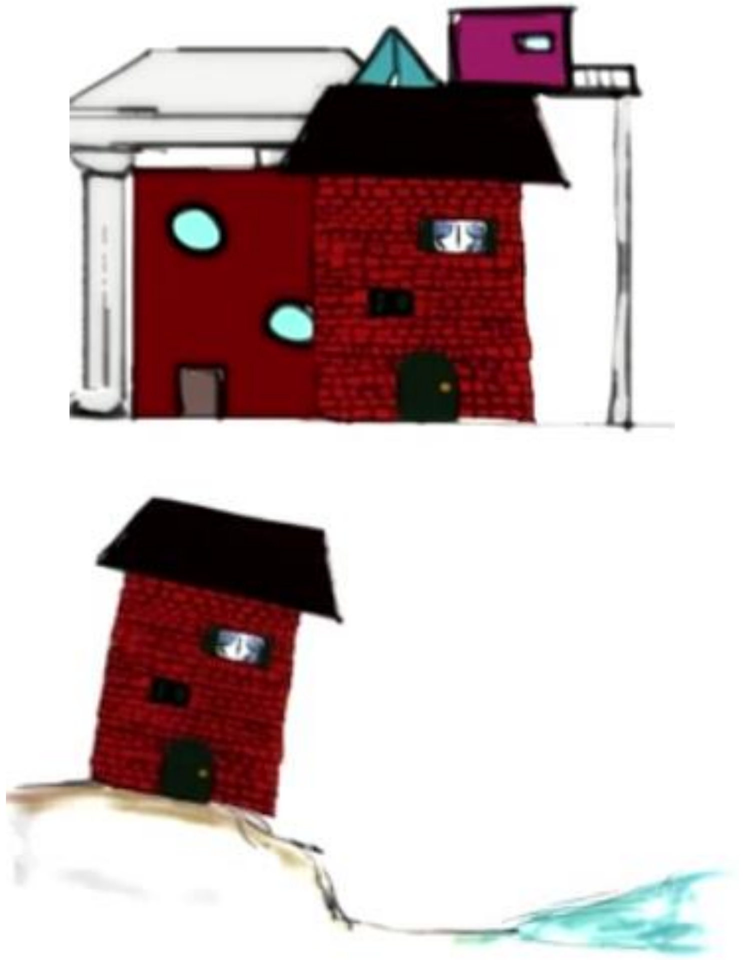
Jane Cleland-Huang, Robert S. Hanmer, Sam Supakkul, Mehdi Mirakhorli. **The Twin Peaks of requirements and architecture.** IEEE Software, vol. 30, no. 2, March-April 2013, pp. 24-29

Matthias Galster,Mehidi Mirakhorli, Jane Cleland-Huang, Janet E. Burge, Xavier Franch, Roshanak Roshandel, Paris Avgeriou. **Views on software engineering from the Twin Peaks of requirements and architecture**. ACM SIGSOFT Software Engineering Notes, vol. 38, no. 5, September 2013, pp.40-42

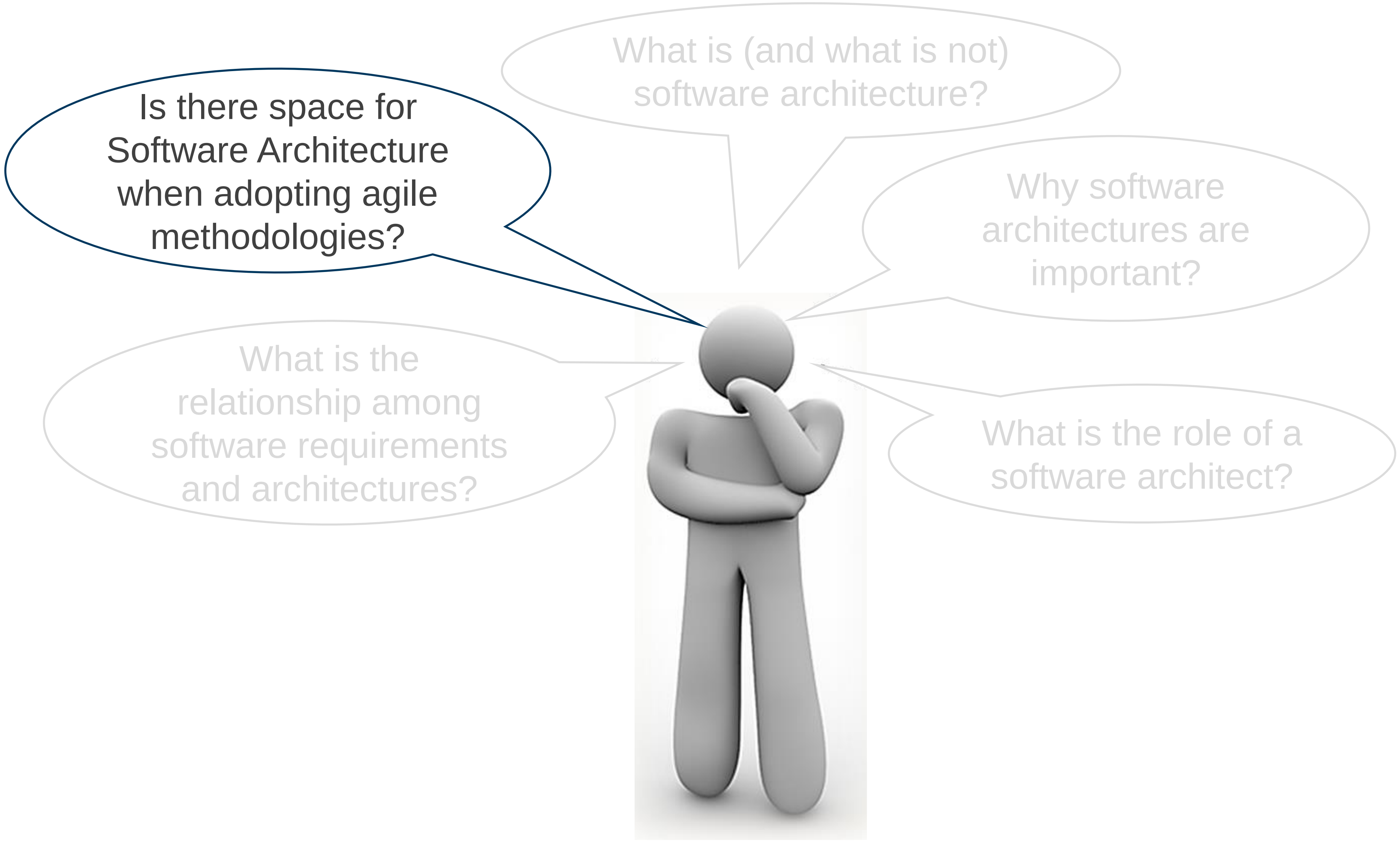
## The requirements–architecture relationship

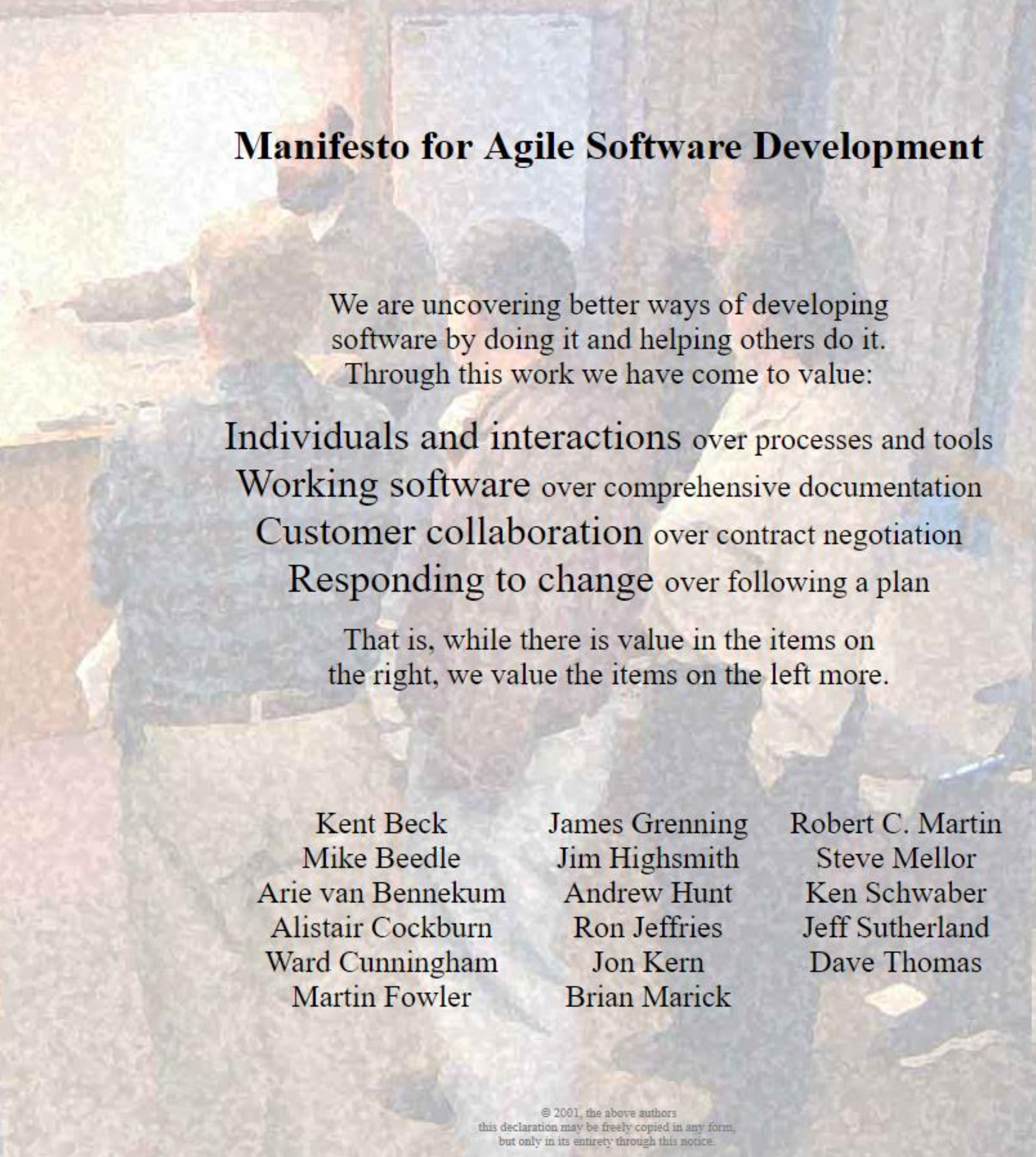
Software architectures properly fill the gap between requirements for a system and its implementation

– Implementation must be faithful to architecture and vice-versa, otherwise there is a risk for architectural degradation, i.e., divergences between the conceived and the implemented architecture



## Existe espaço para Arquitetura de Software na adoção de metodologias ágeis?


What is (and what is not) software architecture?
Is there space for Software Architecture when adopting agile methodologies?
Why software architectures are important?
What is the relationship among software requirements and architectures?
What is the role of a software architect?



## Manifesto for Agile Software Development

We are uncovering better ways of developing
software by doing it and helping others do it.
Through this work we have come to value:

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

That is, while there is value in the items on
the right, we value the items on the left more.

| | | |
|---|---|---|
| Kent Beck | James Grenning | Robert C. Martin |
| Mike Beedle | Jim Highsmith | Steve Mellor |
| Arie van Bennekum | Andrew Hunt | Ken Schwaber |
| Alistair Cockburn | Ron Jeffries | Jeff Sutherland |
| Ward Cunningham | Jon Kern | Dave Thomas |
| Martin Fowler | Brian Marick | |

http://agilemanifesto.org/

## Software architectures in agile software development

### A dichotomy (?)

– Advocates of architecture's vital role in achieving quality goals for large software-intensive systems doubt the scalability of any development approach that does not pay sufficient attention to architecture

– Some adopters of agile approaches usually see little value in up-front architectural design, believing that the architecture should emerge gradually sprint after sprint to avoid massive (and perhaps useless) documentation

## Software architectures in agile software development

Agile software development is well-known for adaptive planning, evolutionary development, early delivery, continual improvement, and rapid and flexible response to changes

Is it possible to follow well-structured architecting practices within agile software development?

Does “agile architecting” exist?

YES!!!

# Software architectures in agile software development

**Agility and Architecture:**
**Can They Coexist?**

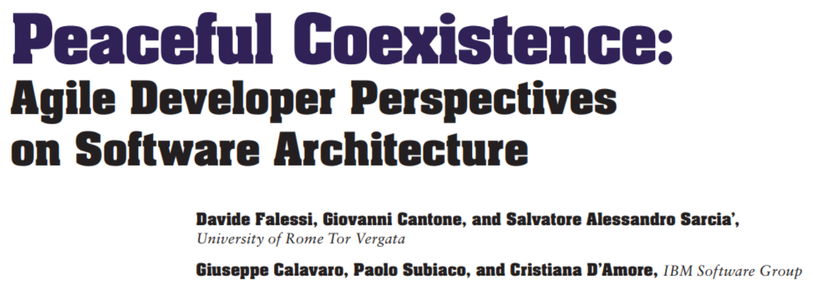
**Pekka Abrahamsson,** *University of Helsinki*
**Muhammad Ali Babar,** *IT University of Copenhagen*
**Philippe Kruchten,** *University of British Columbia*

Pekka Abrahamsson, Muhammad Ali Babar, Philippe Kruchten.
**Agility and architecture: Can they coexist?**
IEEE Software, vol. 27, no. 2, March-April 2010, pp. 16-22

**Agile–Architecture Interactions**

**James Madison**

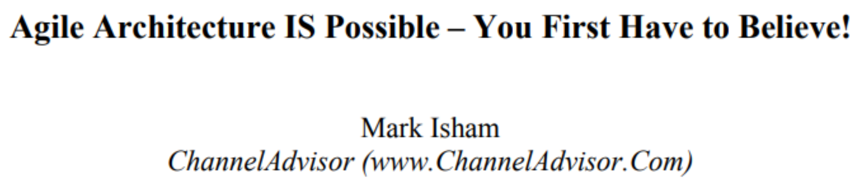
James Madison. **Agile-Architecture interactions.**
IEEE Software, vol. 27, no. 2, March-April 2010, pp. 41-48



**Peaceful Coexistence:**
**Agile Developer Perspectives on Software Architecture**

**Davide Falessi, Giovanni Cantone, and Salvatore Alessandro Sarcia',**
*University of Rome Tor Vergata*
**Giuseppe Calavaro, Paolo Subiaco, and Cristiana D'Amore,** *IBM Software Group*

Davide Falessi, Giovanni Cantone, Salvatore Alessandro Sarcia',
Giuseppe Calavaro, Paolo Subiaco, Cristiana D'Amore.
**Peaceful coexistence: Agile developer perspectives on Software Architecture.**
IEEE Software, vol. 27, no. 2, March-April 2010, pp. 23-25



**Agile Architecture IS Possible – You First Have to Believe!**

Mark Isham
*ChannelAdvisor (www.ChannelAdvisor.Com)*

Mark Isham. **Agile architecture IS possible – You first have to believe!**
Proceedings of the Agile 2008 Conference, Toronto, ON, Canada.
USA: IEEE, 2008, pp. 484-489.

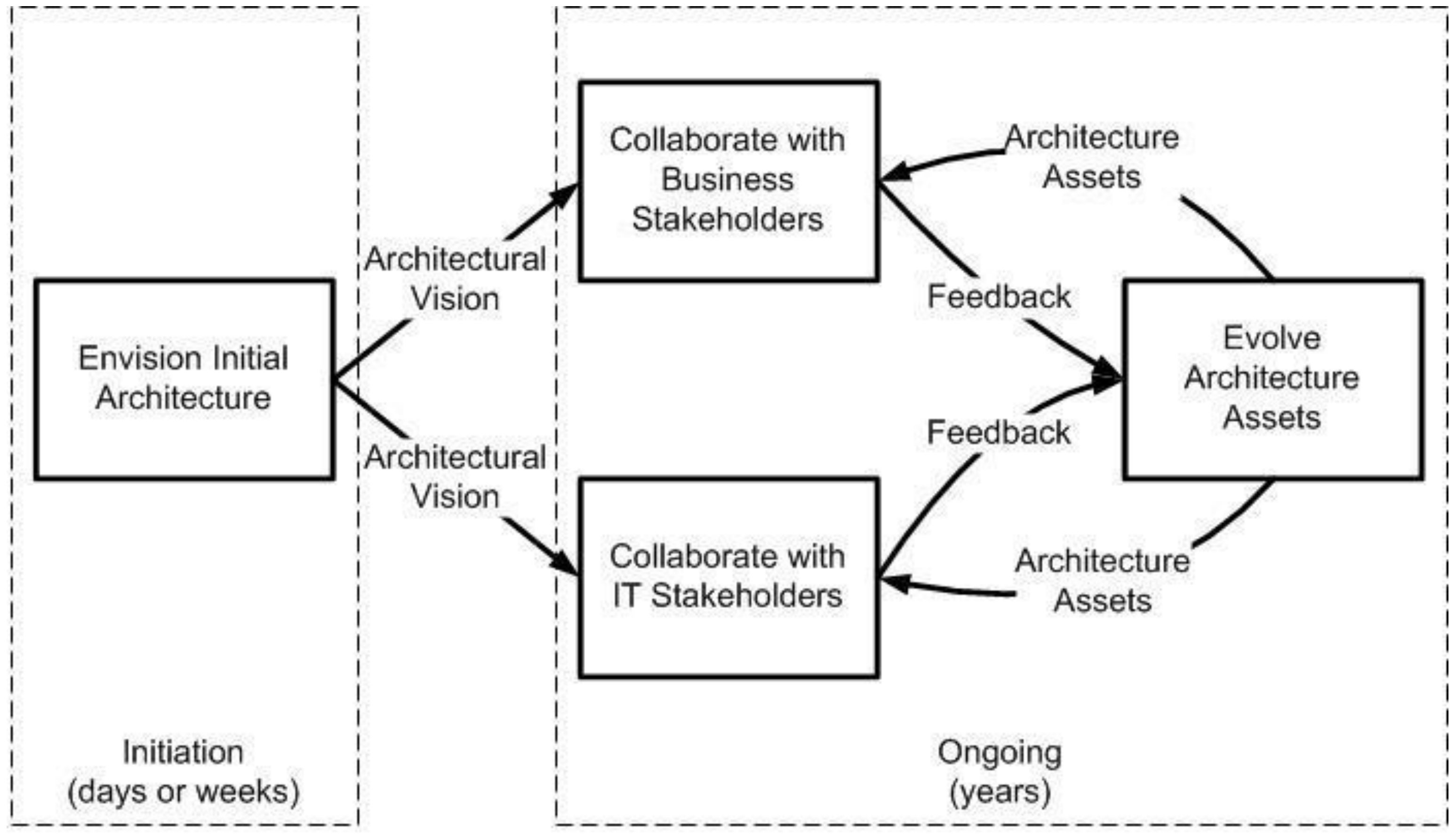
## Software architectures in agile software development

– Adoption of architecture-aware methods and techniques for designing, analyzing, and evolving systems in iterative and incremental ways, with reasonably relevant documentation

– Architecture should be seen not as immutable, but as an asset to reevaluate at each iteration in close collaboration between architects and developers

– There must be a balance on how much architectural activity is needed for the project depending on its size and complexity

## Software architectures in agile software development


Envision Initial Architecture
Architectural Vision
Architectural Vision
Collaborate with Business Stakeholders
Collaborate with IT Stakeholders
Architecture Assets
Feedback
Feedback
Architecture Assets
Evolve Architecture Assets
Initiation (days or weeks)
Ongoing (years)

## Software architectures in agile software development

– Software architecture is relevant not only for complex software systems, but also for
  - communication among team members
  - inputs to subsequent design decisions
  - documenting design assumptions
  - evaluating design alternatives

– Integrating architectural approaches in agile processes can enable software development teams to pay attention not only to functional requirements, but also to the non-functional ones

## Software architectures in agile software development

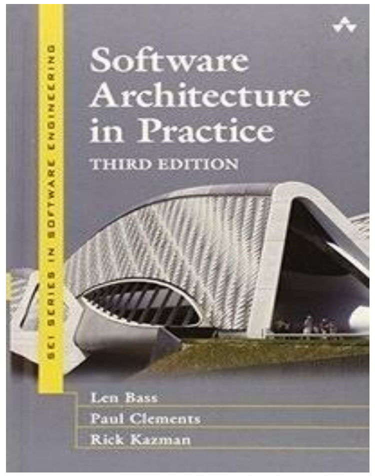
- Waiting too long to take care of architecturally significant decisions can put the whole project in chaos
- Even when using agile methodologies, it is possible to obtain a good architecture by appropriately applying suitable combinations of architectural functions in the development life cycle

## The take away message

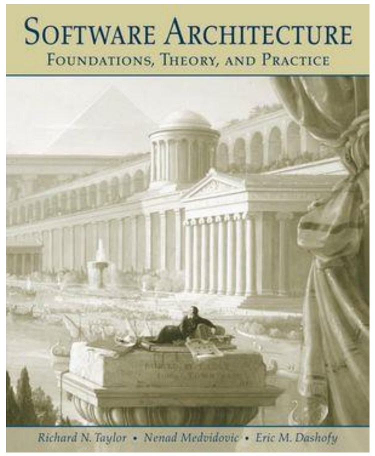
If you think good architecture is expensive, try bad architecture.

— Brian Foote —

# Further reading


Software Architecture in Practice
THIRD EDITION
SEI SERIES IN SOFTWARE ENGINEERING
Len Bass
Paul Clements
Rick Kazman

Len Bass, Paul Clements, Rick Kazman.
Software architecture in practice – 3rd ed.
USA: Addison-Wesley/Pearson Education, Inc., 2013


SOFTWARE ARCHITECTURE
FOUNDATIONS, THEORY, AND PRACTICE
Richard N. Taylor • Nenad Medvidovic • Eric M. Dashofy

Richard N. Taylor, Nenad Medvidovic, Eric M. Dashofy.
Software Architecture: Foundations, theory, and practice.
USA: John Wiley & Sons, Inc., 2010

# Software Architecture
Concepts

Prof. Dr. Everton Cavalcante

http://www.dimap.ufrn.br/~everton/