Design Patterns & Software Architecture

Introduction to Software Architecture

dr. Joost Schalken-Pinkster

Windesheim University of Applied Science

The Netherlands

The contents of these course slides are based on:
Leo Pruijt, *Course on Software Architecture*. Hogeschool Utrecht, 2010-2013.

Session overview

- Definition of software architecture
- Software architecture products
- Software architecture & software projects
  - Role of Software Architect
  - Software architecture and quality attributes

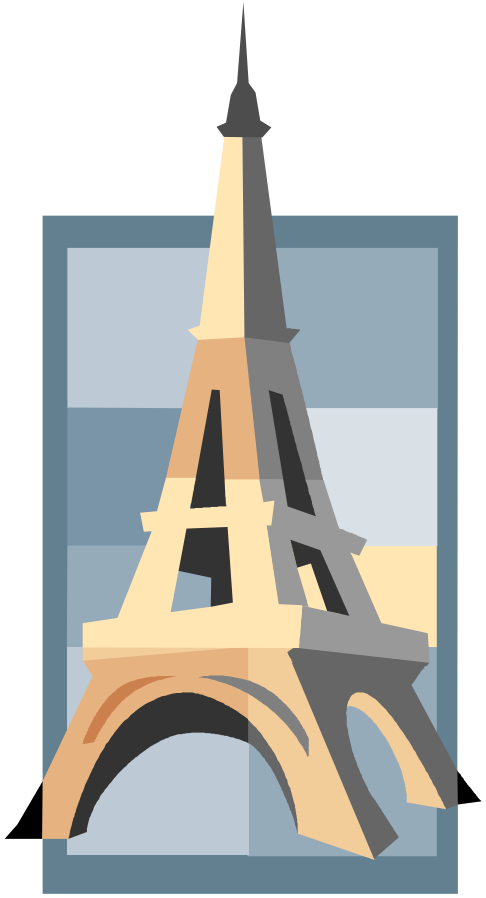
Definition of software architecture

What is Software Architecture…
Process of defining a structured solution

Software application architecture is the process of defining a structured solution

- that meets all of the technical and operational requirements,
- while optimizing common quality attributes such as performance, security, and manageability.

It involves a series of decisions based on a wide range of factors, and each of these decisions can have considerable impact on the quality, performance, maintainability, and overall success of the application.

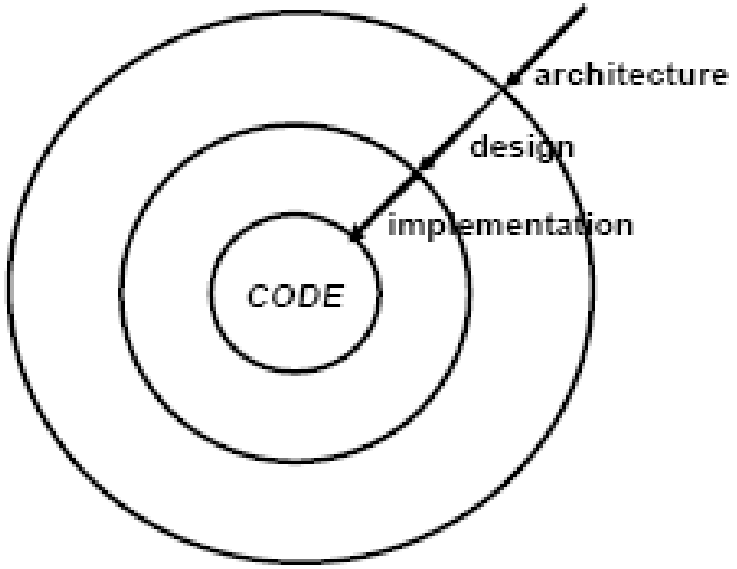


J.D. Meier, David Hill et. al (2009) Microsoft Application Architecture Guide, 2nd Edition. Microsoft Press, 2009. Chapter 1. http://msdn.microsoft.com/en-us/library/ff650706.aspx

What is Software Architecture….
A set of significant decisions…

Software architecture encompasses the set of significant decisions about the organization of a software system:

- Selection of the structural elements and their interfaces by which a system is composed
- Behavior as specified in collaborations among those elements
- Composition of these structural and behavioral elements into larger subsystems
- Architectural style that guides this organization



Architecture decisions are the most fundamental decisions. And changing them will have significant ripple effects!

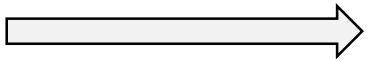
G. Booch, J. Rumbaugh, and I. Jacobson (1999). *The Unified Modeling Language User Guide*. Addison-Wesley, 1999 (as cited in C. Larman (2004), Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development, 3rd ed. Addison Wesley, 2004. Chapter 13)

Software architecture products

Products of software architecture

A software architect has to find solutions to implement all the:

- Functional requirements
  - Specification of the functionality required by the user organization
- Non-functional requirements (Quality requirements)
  - E.g. maintainability, performance, scalability, security, portability, …

Products (artifacts) 

- Architectural products should explain how the requirements can be realized.

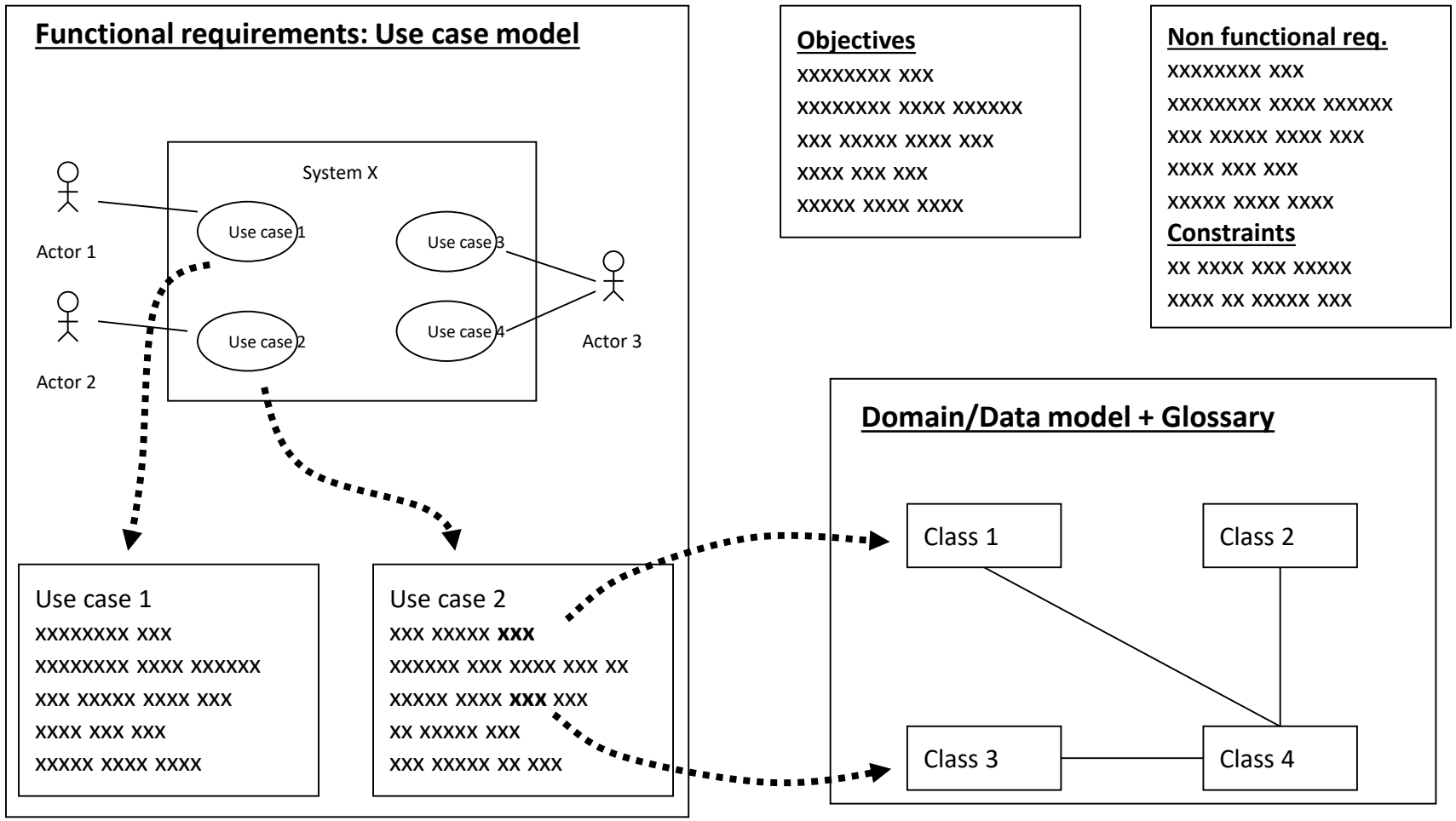
Architecture Notebook contains at least:

1. Architectural goals
2. Architecturally significant requirements
   - Non-functional requirements
   - Key functional requirements
     - Use case model
3. Domain class model
4. Decisions, principles, justifications
5. Software partitioning model
   - System/subsystem decomposition
   - Layer model (logical)
   - Logical clusters
   - Component model (logical & physical)
   - Key scenario's to validate the partitioning
6. Tier model (physical)
7. Deployment model

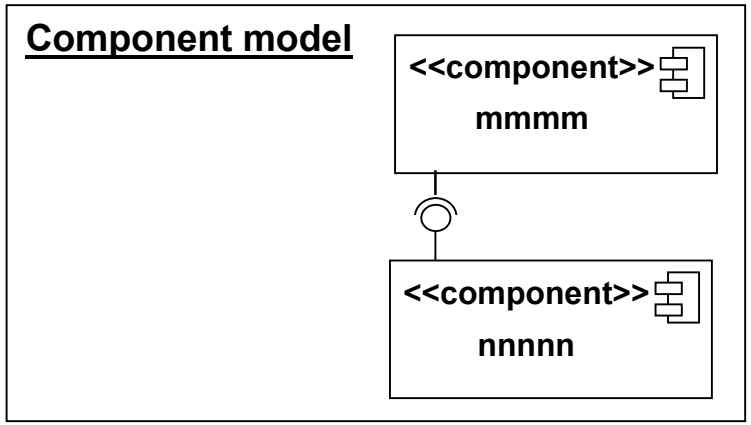
Requirements (what & why)

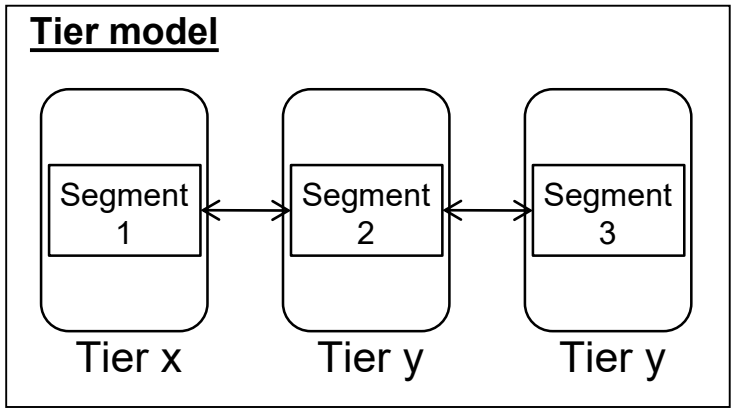
Design (how)

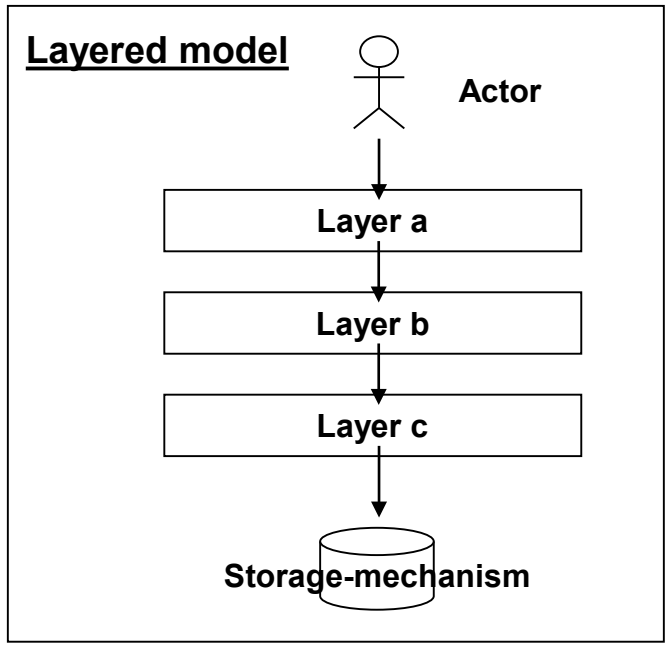
Characteristic Products of Requirements Analysis

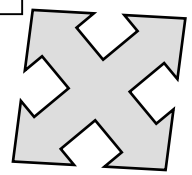


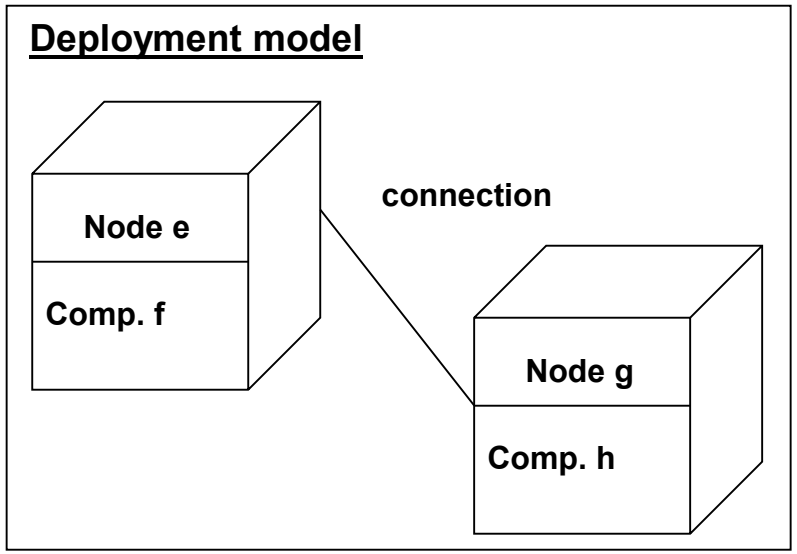
Characteristic products of Software Architecture







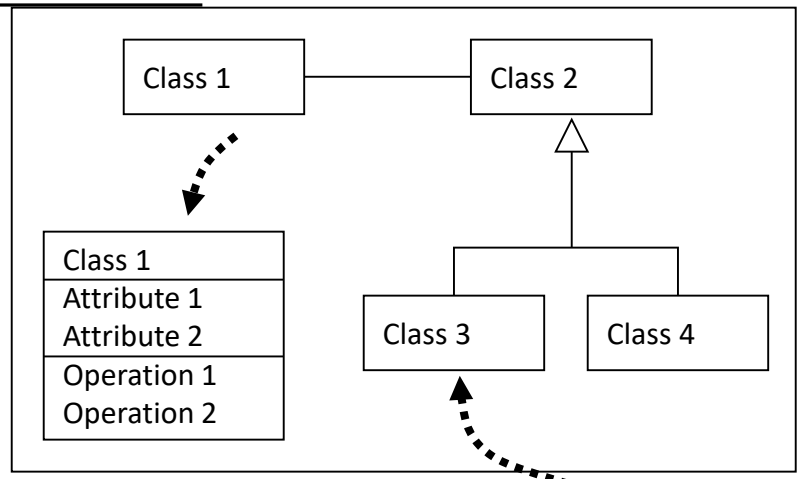




Characteristic Design Products

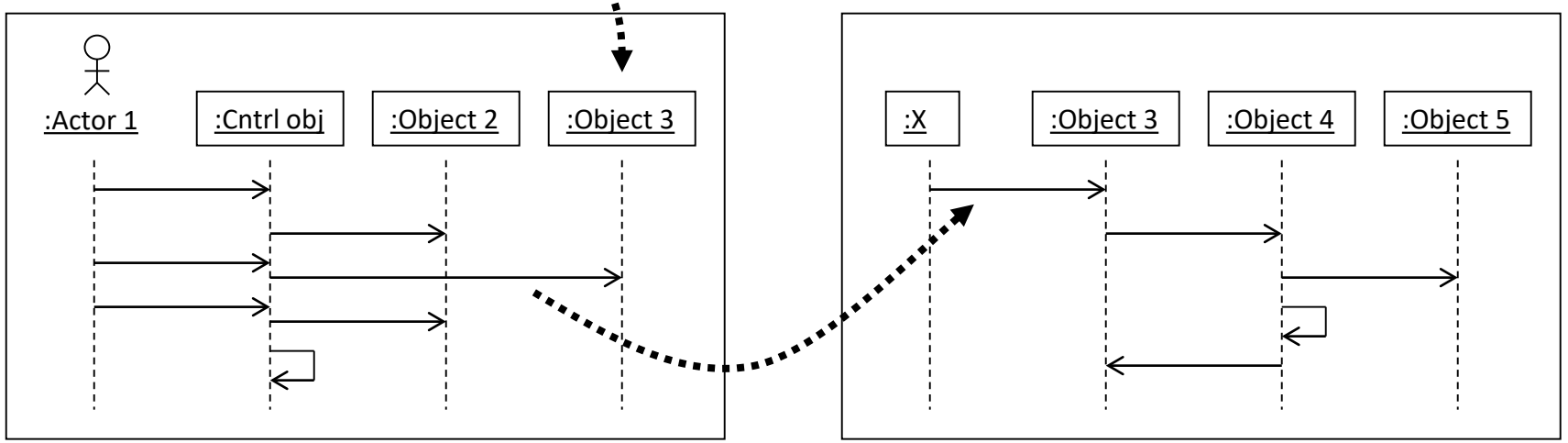
(Used to design and document Key Scenarios)

Class model



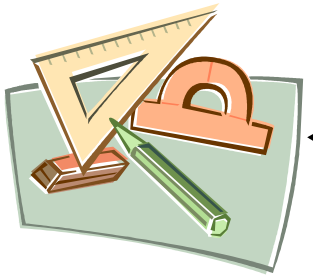
+ Textual Specifications

Sequence diagram for a scenario



Software architecture & software projects

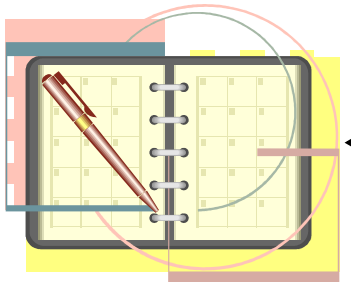
The role of the architect in projects



Analyse & design:
- Requirements
- Architecture Notebook
- Prototype

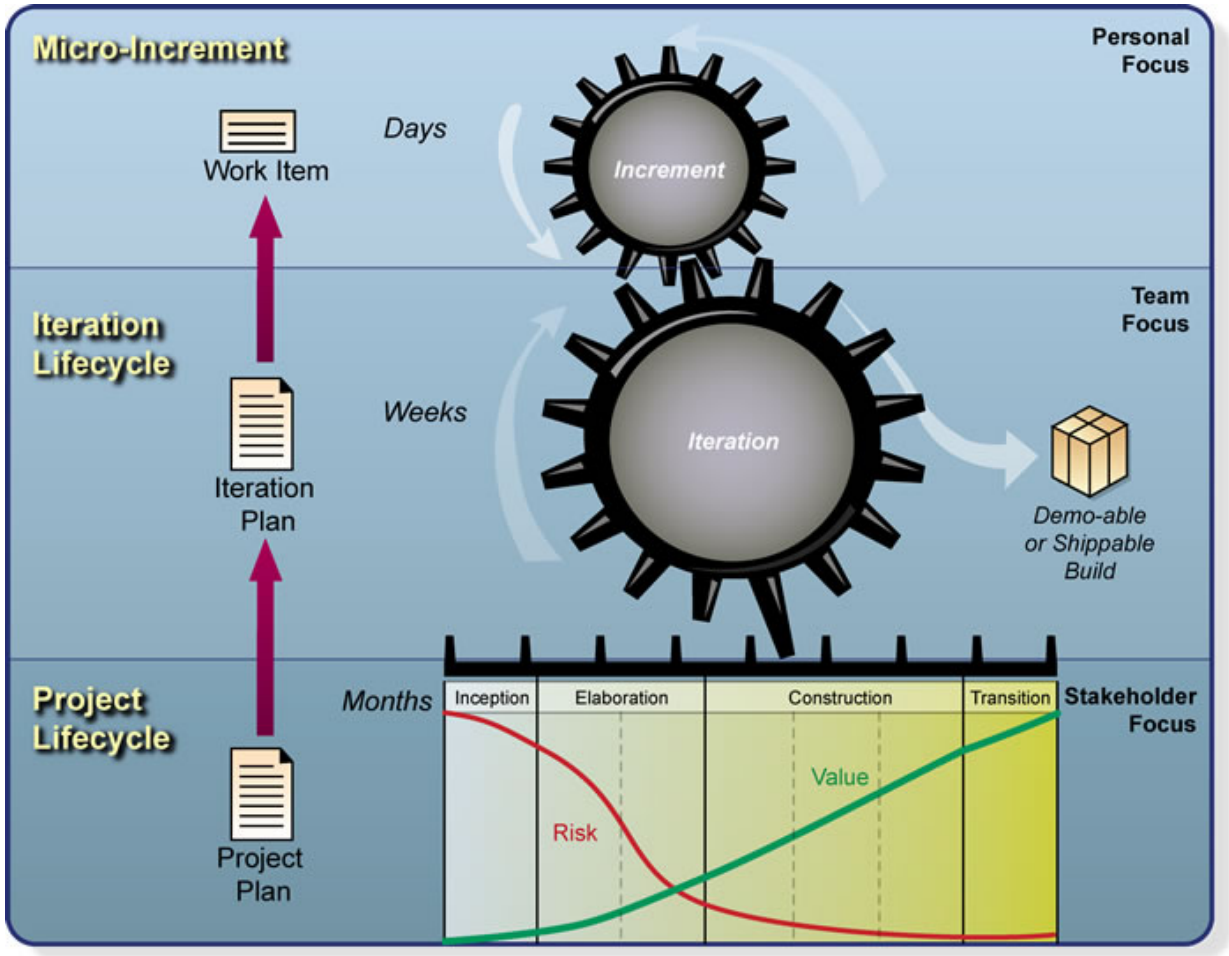
Communicate with Stakeholders:
- Shared Vision
- Requirements

Communicate with Project Manager:
- Risks
- Work division



Communicate with Developers:
- Guidelines
- Prototypes
- Implementation problems

Software Architecture happens in a process: Example: RUP/OpenUp



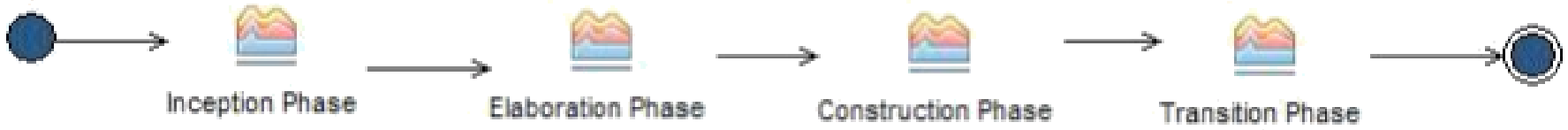
Michael Yang. *OpenUp*. Eclipse Process Framework (EPF), June 2012. http://epf.eclipse.org/wikis/openup/

Software Architecture and OpenUp

Architecture takes an important position within OpenUp!

- And it describes and explaines relevant products.
- Missing in most other agile methods.
- Reason why the method is suitable for education and young professionals.

Architecture and OpenUp Phases:



Inception

- Outline the Architecture

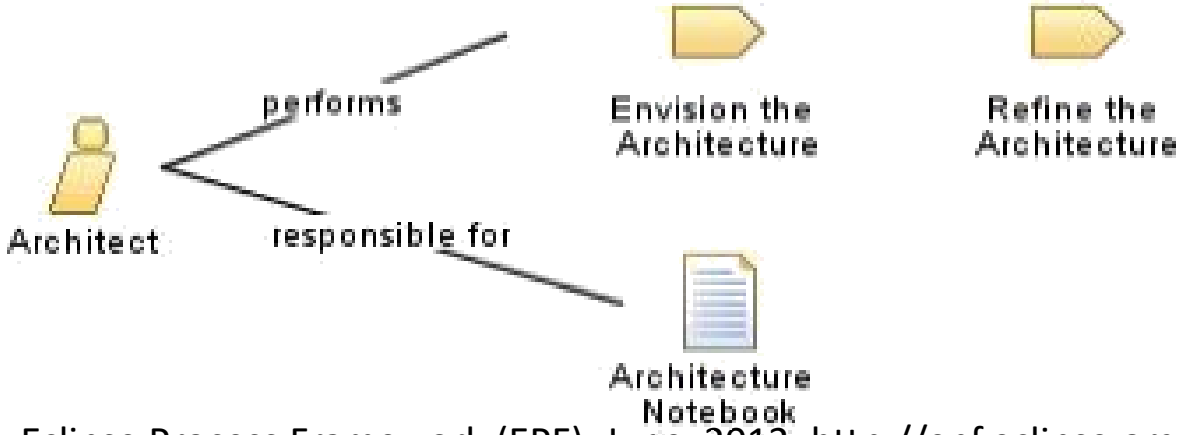
Elaboration

- Refine the Architecture
- At the end of this phase, the main milestone is to have a stable architecture that will be used as the basis for system development.

Michael Yang. *OpenUp*. Eclipse Process Framework (EPF), June 2012. http://epf.eclipse.org/wikis/openup/

Purpose of Architecture conform OpenUP

- To describe the essential structure of the system
- To describe the decisions guiding the structure of the system
- To identify and attack risks to the system
- To provide context and guidance for developers
  - The architecture serves as the blueprint for development.
- To define the project structure and team organization.
  - Architectural elements make excellent units of work

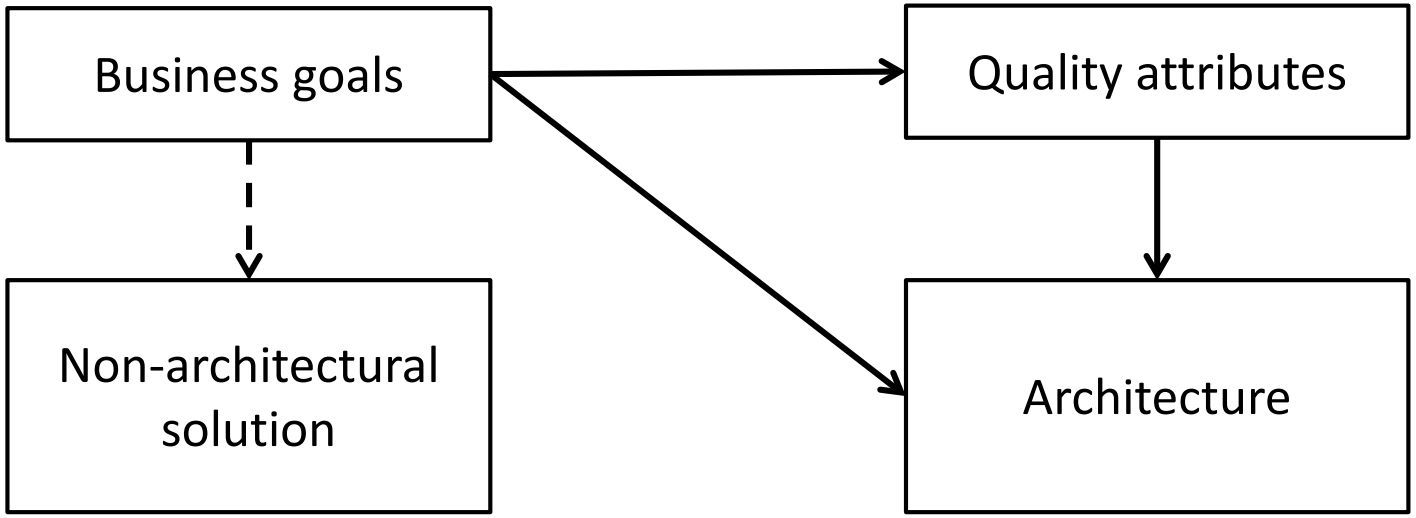


Michael Yang. *OpenUp*. Eclipse Process Framework (EPF), June 2012. http://epf.eclipse.org/wikis/openup/

Achieving quality attributes

Architecture design focuses on achieving quality attributes:

- Software is created to satisfy the business needs of organizations.
- Business needs are translated into quality attributes.



Bass, Clements & Kazman (2013), Software Architecture in Practice, 3rd. Addison-Wesley. Ch. 3.3

Quality attributes guide architectural design

Quality attributes focus thinking on the critical problems that the architecture should solve.

- Depending on the requirements, sometimes all quality attribute should be covered, sometimes a subset.
  - E.g. every design must consider security and performance, but always interoperability or scalability.
- Understand requirements and deployment scenarios to know the important quality attributes.
- Keep in mind that quality attributes may conflict; for example, security often requires a trade-off against performance or usability.

J.D. Meier, David Hill et. al (2009) Microsoft Application Architecture Guide, 2nd Edition. Microsoft Press, 2009. Chapter 2. http://msdn.microsoft.com/en-us/library/ff650706.aspx

Quality attributes according to ISO 25010:

Product quality

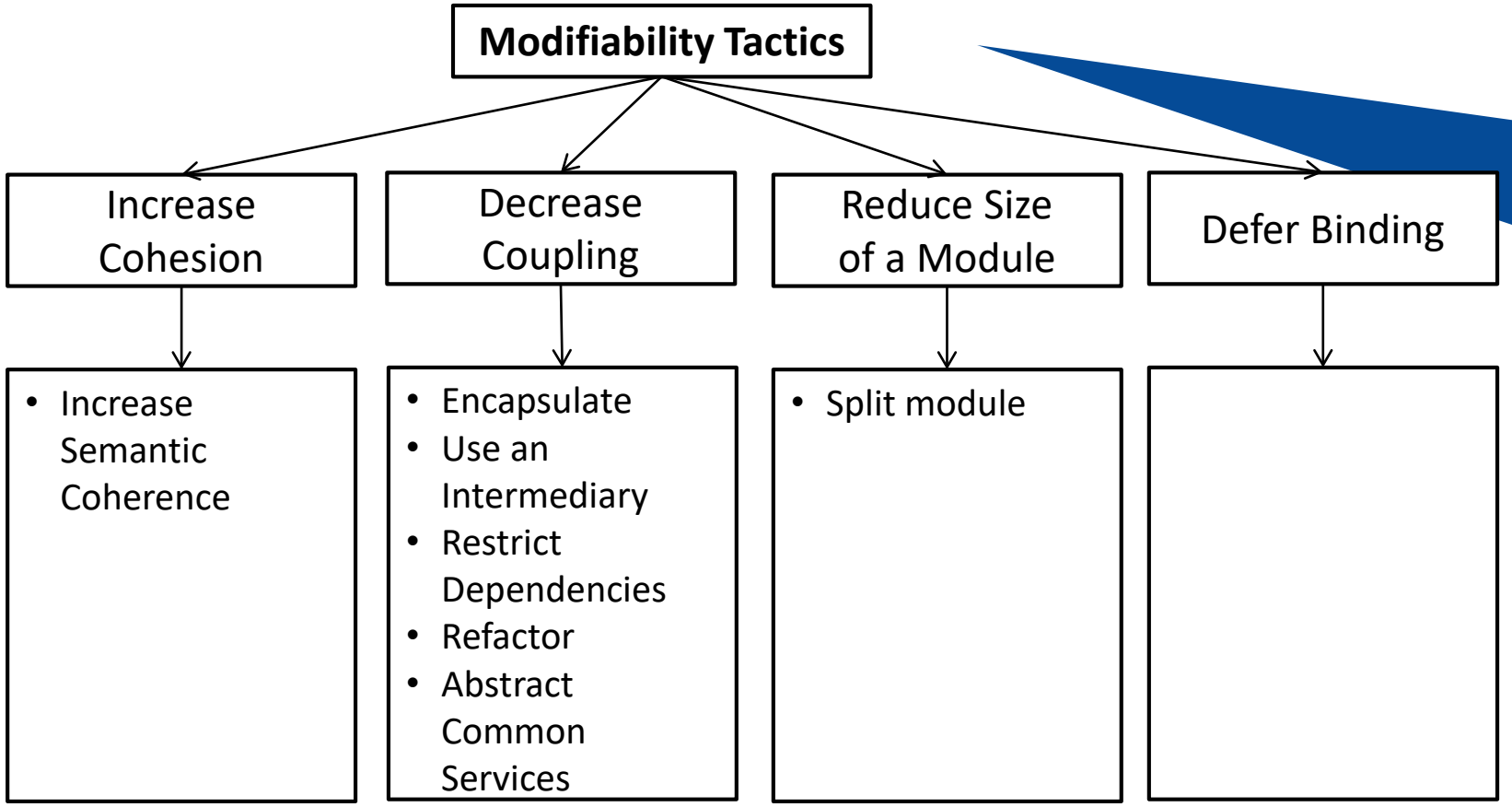
- Functional suitability
- Performance efficiency
- Compatibility
- Usability
- Reliability
- Security
- **Maintainability** ⟵ Focus of this course…
- Portability

Quality in use

- Effectiveness
- Efficiency
- Satisfaction
- Freedom from risk
- Context coverage

ISO/IEC (2010) Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models. International Standard 25010. ISO/IEC.

Tactics to achieve maintainability/modifiability



Design patterns help you achieve the implementation of these tactics on a design level…

Bass, Clements & Kazman (2013), *Software Architecture in Practice*, 3rd. Addison-Wesley. Ch. 7

Guidelines on quality attributes

When designing to accommodate quality attributes, consider the following guidelines:

- Quality attributes are system properties that are separate from the functionality of the system.
- From a technical perspective, implementing quality attributes can differentiate a good system from a bad one.
- Analyze the trade-offs between quality attributes.
  - What are the key quality attributes required for your application?
  - What are the key requirements for addressing these attributes?
  - What are the acceptance criteria that will indicate that you have met the requirements?

J.D. Meier, David Hill et. al (2009) Microsoft Application Architecture Guide, 2nd Edition. Microsoft Press, 2009. Chapter 2. http://msdn.microsoft.com/en-us/library/ff650706.aspx

Where do we document the quality attributes?

Architecture Notebook

1. **Architectural goals**
2. **Architecturally significant requirements**
   - **Non-functional requirements**
   - Key functional requirements (including use case model)
3. Domain class model
4. Decisions, principles, justifications
5. Software partitioning model
   - System/subsystem decomposition
   - Layer model (logical)
   - Logical clusters
   - Component model (logical & physical)
   - Key scenario's to validate the partitioning
6. Tier model (physical)
7. Deployment model

Architectural goals contains the goals that the organisation wants to reach in the future (with the system) and changes in the IT landscape that are expected…

Architectural significant requirements contains the goals that must be by this project…

Reading

For next lesson please read:

- Chapter 1 (What is Software Architecture?) of the Microsoft Application Architecture Guide. http://msdn.microsoft.com/en-us/library/ee658098.aspx
  (also available as pdf on: http://msdn.microsoft.com/en-us/library/ff650706.aspx)

Additional reading (not part of the examination, not required):

- Chapter 2 (Key Principles of Software Architecture) of the Microsoft Application Architecture Guide. http://msdn.microsoft.com/en-us/library/ee658124.aspx