Architectural views, styles, and patterns

Prof. Dr. Everton Cavalcante

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

Things to remember

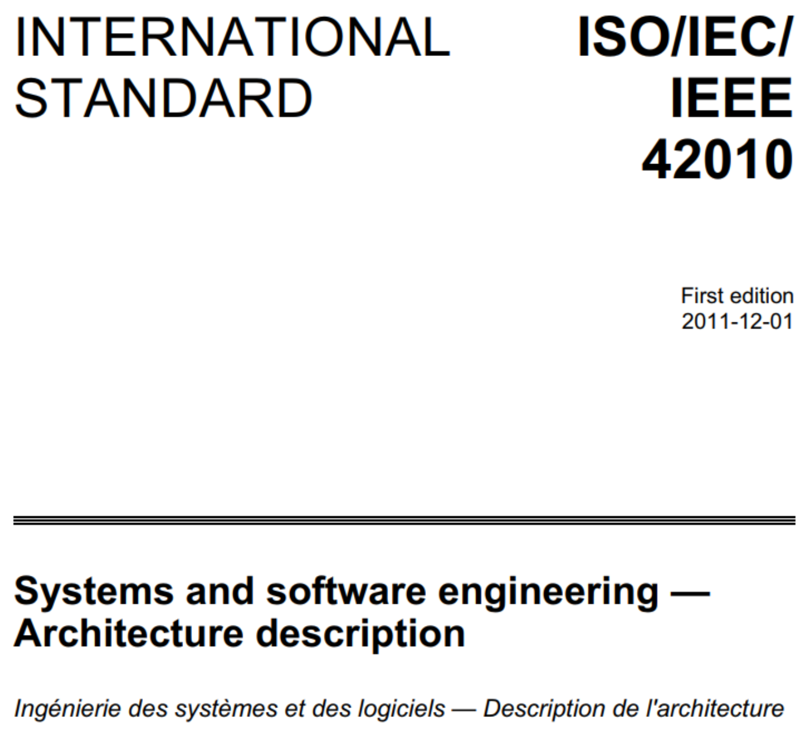
- Software Architecture is an important discipline to understand structure, behavior, and properties of complex software systems
- The issue is on how to organize a system to simultaneously
  - make the suitable decisions
  - provide the required functionalities (functional requirements)
  - guarantee the required quality of service (non-functional requirements)

Things to remember

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

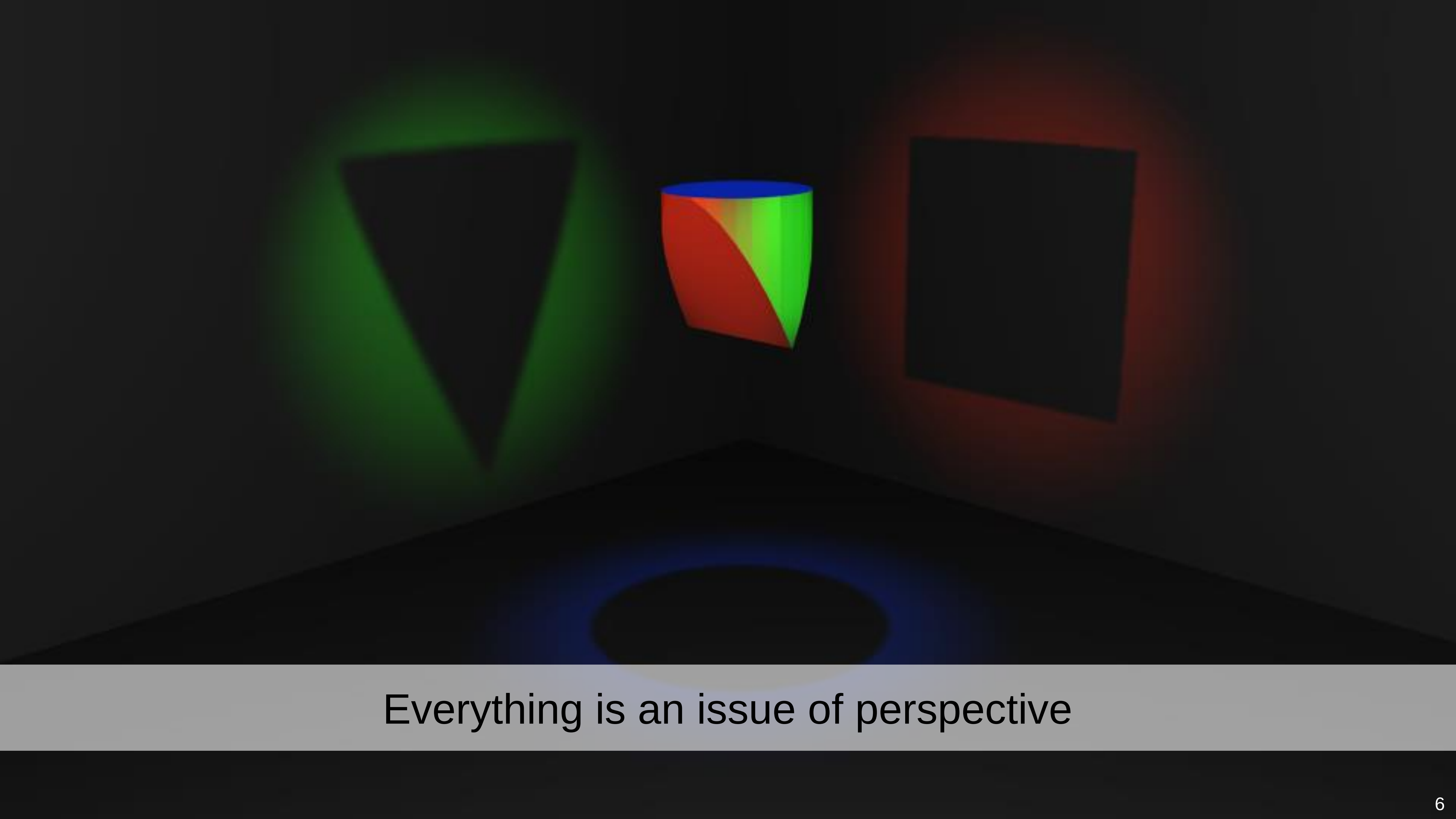
Systems and software engineering — Architecture description

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

Things to remember

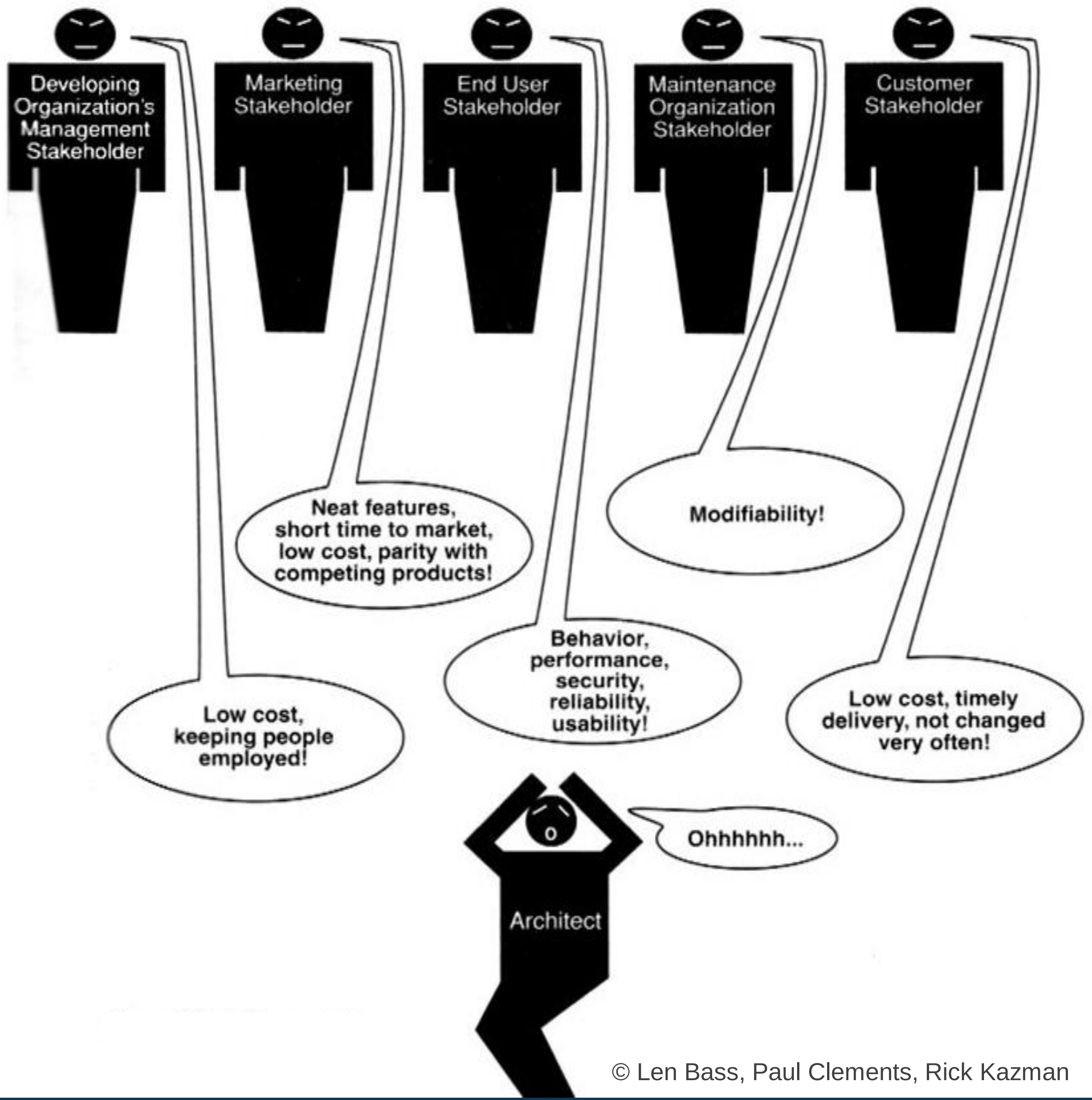
Software architectures materialize important concerns

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	characteristics and/or constraints (typically related to non-functional requirements) 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



Everything is an issue of perspective

Ponto de contato


Developing Organization's Management Stakeholder
Marketing Stakeholder
End User Stakeholder
Maintenance Organization Stakeholder
Customer Stakeholder
Neat features, short time to market, low cost, parity with competing products!
Modifiability!
Behavior, performance, security, reliability, usability!
Low cost, keeping people employed!
Low cost, timely delivery, not changed very often!
Ohhhhhh...
Architect

© Len Bass, Paul Clements, Rick Kazman

For the case of software architectures, concerns may vary depending on the stakeholder

Decision-making in architecting

- What are the main functional elements of the architecture?
- How will these elements interact with one another and with the outside world?
- What information will be managed, stored, and presented?
- What physical hardware and software elements will be required to support functional and information elements?
- What operational features and capabilities will be provided?
- What development, test, support, and training environments will be provided?

Goals

- To introduce the concepts of
  - architectural view
  - architectural style
  - architectural pattern
- To briefly present the main existing architectural views, styles, and patterns

One size does not fit all

One size does not fit all

- A software architecture is a complex entity that cannot be considered in a one-dimensional fashion
- A single, all-encompassing model covering different concerns will become heavily overloaded and hard to understand
- Not all perspectives are of value to all stakeholders

Architectural views

The problem of architecting must be addressed from **different directions**

– Separate, but interrelated **views**, each one concerning a different concern of the architecture

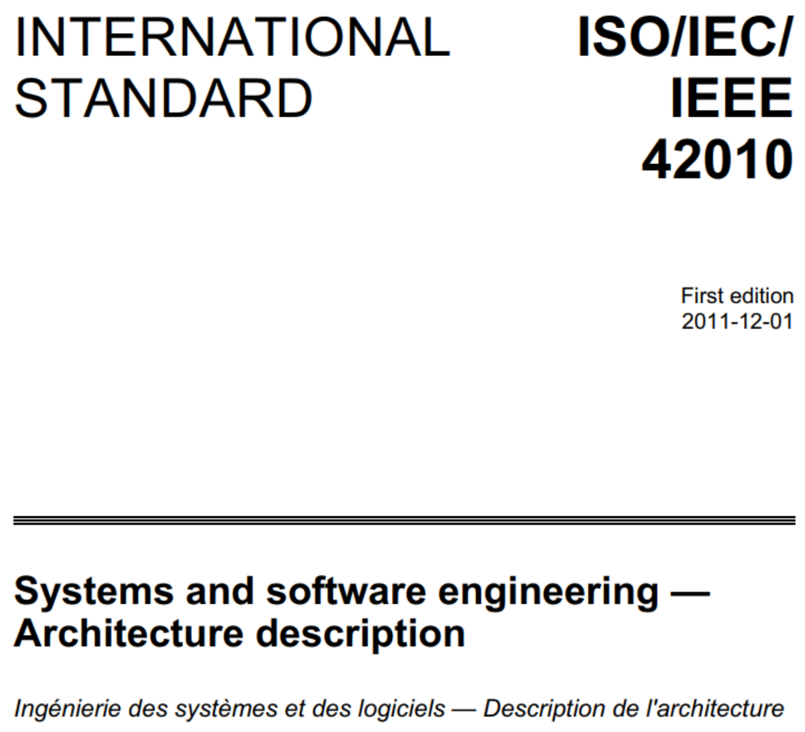
– The **ensemble of different views** will provide the understanding of the system as a whole

Architectural views

Architectural view

Work product expressing the architecture of a system from the **perspective of specific system concerns**

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

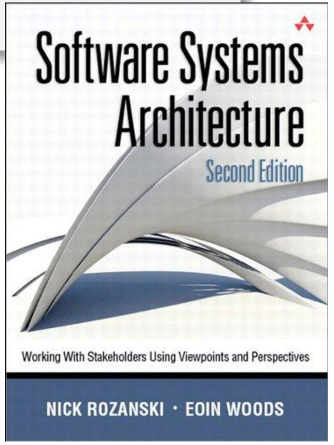
Systems and software engineering — Architecture description

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

Architectural views

Architectural view

is a representation of one or more structural aspects of an architecture that illustrates **how the architecture addresses one or more concerns held by one or more of its stakeholders**

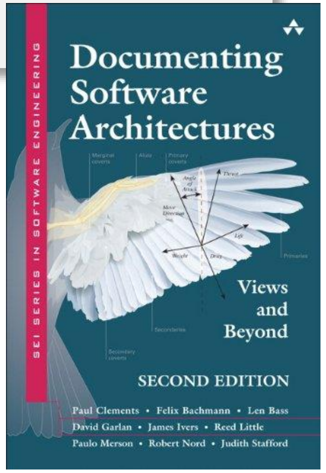

Software Systems Architecture
Second Edition
Working With Stakeholders Using Viewpoints and Perspectives
NICK ROZANSKI · EOIN WOODS

Nick Rozanski, Eóin Woods.
Software systems architecture:
Working with stakeholders using viewpoints and perspectives
– 2nd ed. USA: Addison-Wesley/Pearson Education, Inc., 2012

Architectural views

Architectural view

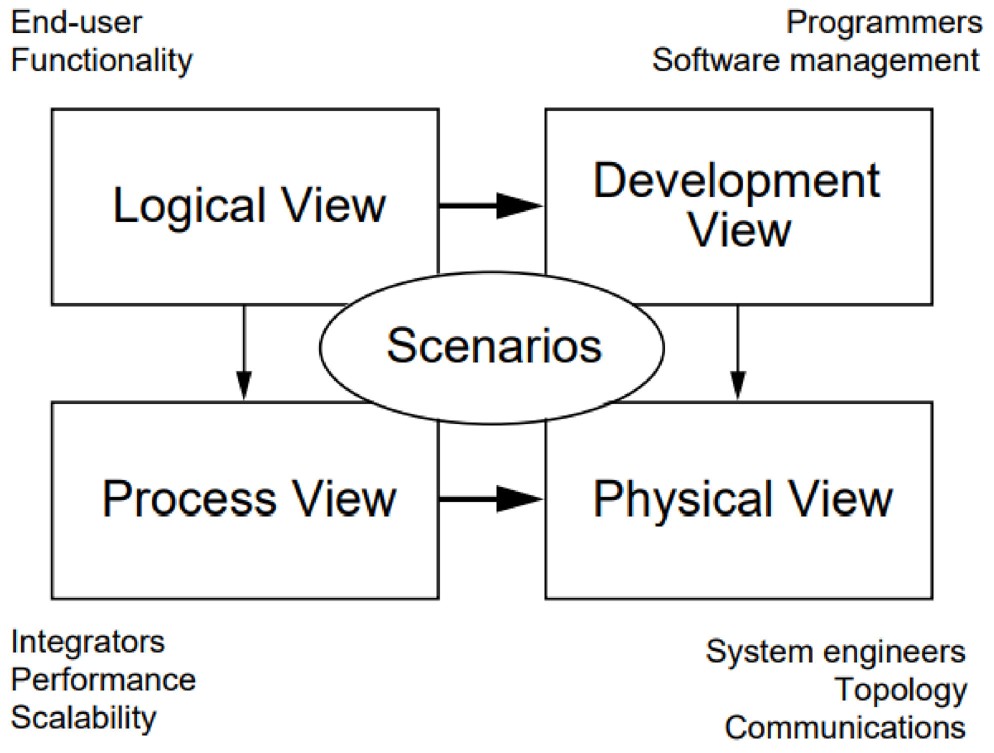
is a **representation** of a set of **system elements and the relationships** associated with them

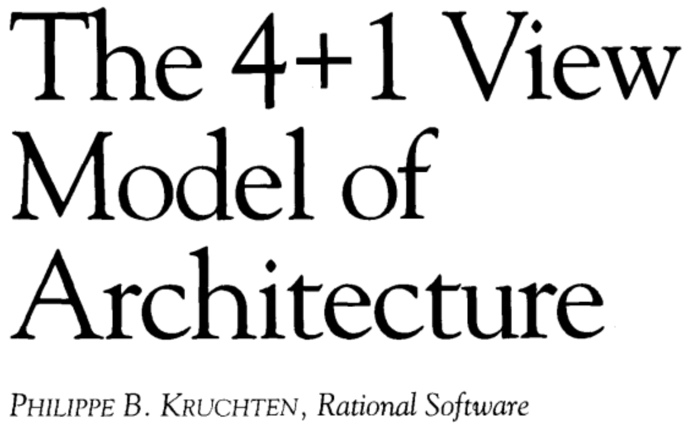


Paul Clements et al.
Documenting software architectures: Views and beyond – 2nd ed. USA: Addison-Wesley/Pearson Education, Inc., 2011

Architectural views

The Kruchten's "4+1" Architectural View Model (1995)




The 4+1 View Model of Architecture

PHILIPPE B. KRUCHTEN, *Rational Software*

Phillipe Kruchten. **Architectural blueprints – The "4+1" View Model for software architecture.** IEEE Software, vol. 12, no. 6, November 1995, pp. 42-50

Architectural views

– A view is typically represented by one or more models

– Multiple views are essential to cover all the stakeholders’ concerns and to detail the architecture from different perspectives

- Each view emphasizes certain aspects of the system while deemphasizing or ignoring other aspects, all in the interest of making the problem at hand tractable
- A reality in industry

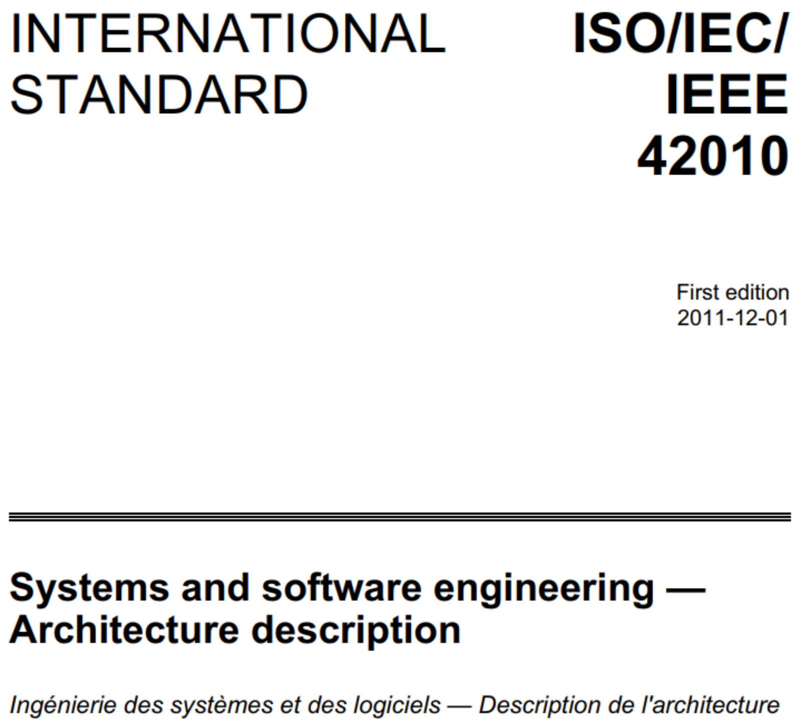
Architectural views vs. Architectural viewpoints

Architectural views are associated with architectural viewpoints

Architectural viewpoint

work product establishing the **conventions for the construction, interpretation and use of architecture views** to frame **specific system concerns**

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

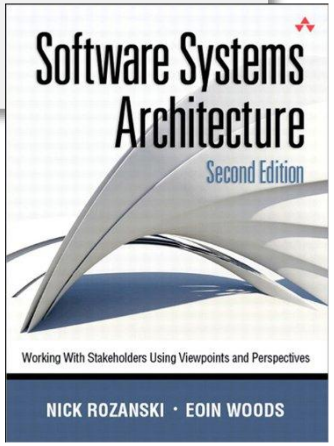
Ingénierie des systèmes et des logiciels — Description de l'architecture

Architectural views vs. Architectural viewpoints

Architectural views are associated with architectural viewpoints

Architectural viewpoint

is a collection of **patterns, templates, and conventions for constructing one type of view**. It defines the **stakeholders** whose concerns are reflected in the viewpoint and the **guidelines, principles, and template models** for constructing its views


Software Systems Architecture
Second Edition
Working With Stakeholders Using Viewpoints and Perspectives
NICK ROZANSKI · EOIN WOODS

Nick Rozanski, Eóin Woods.
Software systems architecture:
Working with stakeholders using viewpoints and perspectives
– 2nd ed. USA: Addison-Wesley/Pearson Education, Inc., 2012

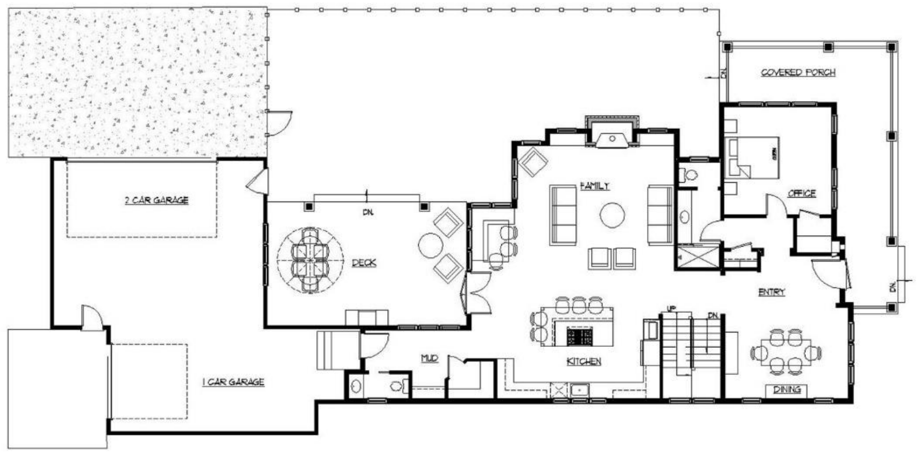
Architectural views vs. Architectural viewpoints

Architectural viewpoints realize stakeholders concerns by means of architecture views

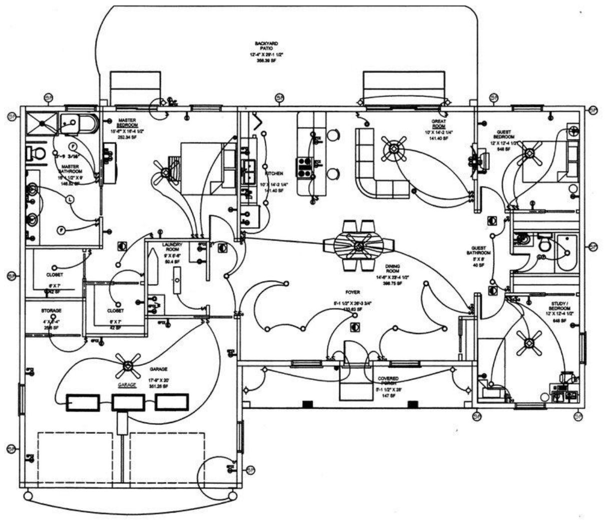
– A view is what a stakeholder sees from a given viewpoint (i.e., how he/she sees or what he/she is interested into)

– The different views communicate the architecture to the different stakeholders

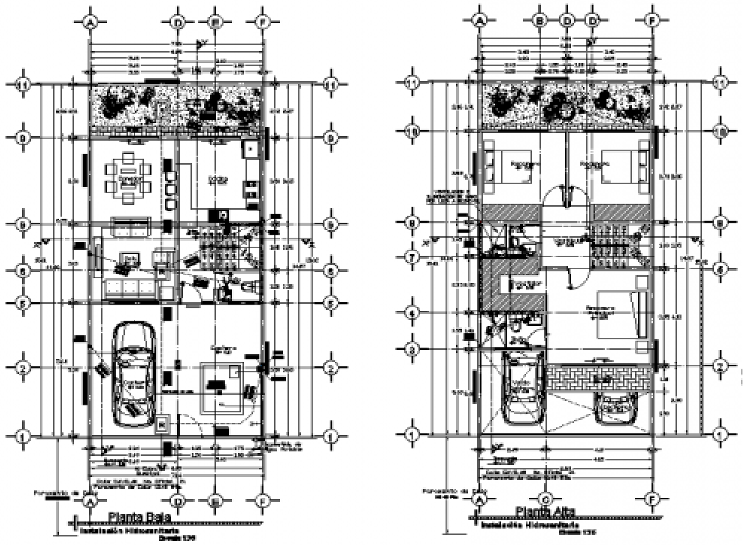
Architectural views vs. Architectural viewpoints



Building structural plan



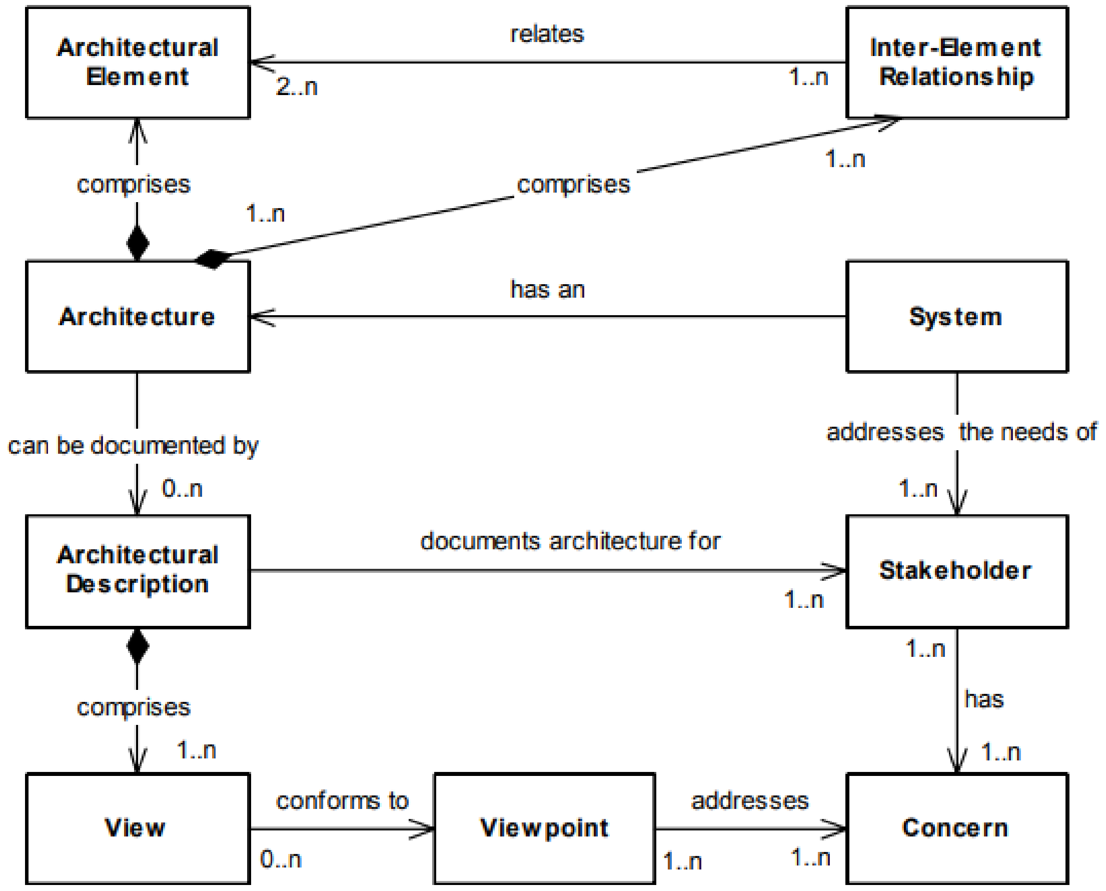
Building electrical plan



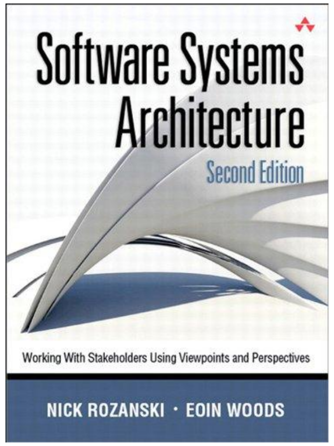
Building hydraulic plan

A building architecture can be viewed from different, specific perspectives of interest for different stakeholders

Architectural views vs. Architectural viewpoints



Nick Rozanski, Eóin Woods.
Software systems architecture: Working with stakeholders using viewpoints and perspectives – 2nd ed.
USA: Addison-Wesley/Pearson Education, Inc., 2012



Architectural views vs. Architectural viewpoints

Typical architectural views and viewpoints

Structural viewpoint

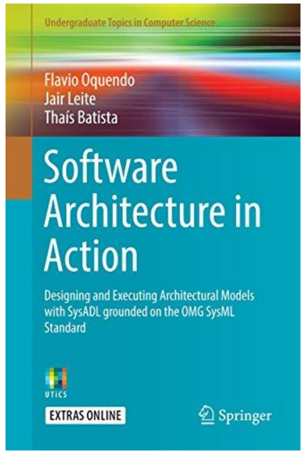
- **Conceptual elements** composing the architecture and **how they are interconnected** to achieve the system functionality
- Main view elements: **components**, **connectors**, **ports**, **configurations**

Behavioral viewpoint

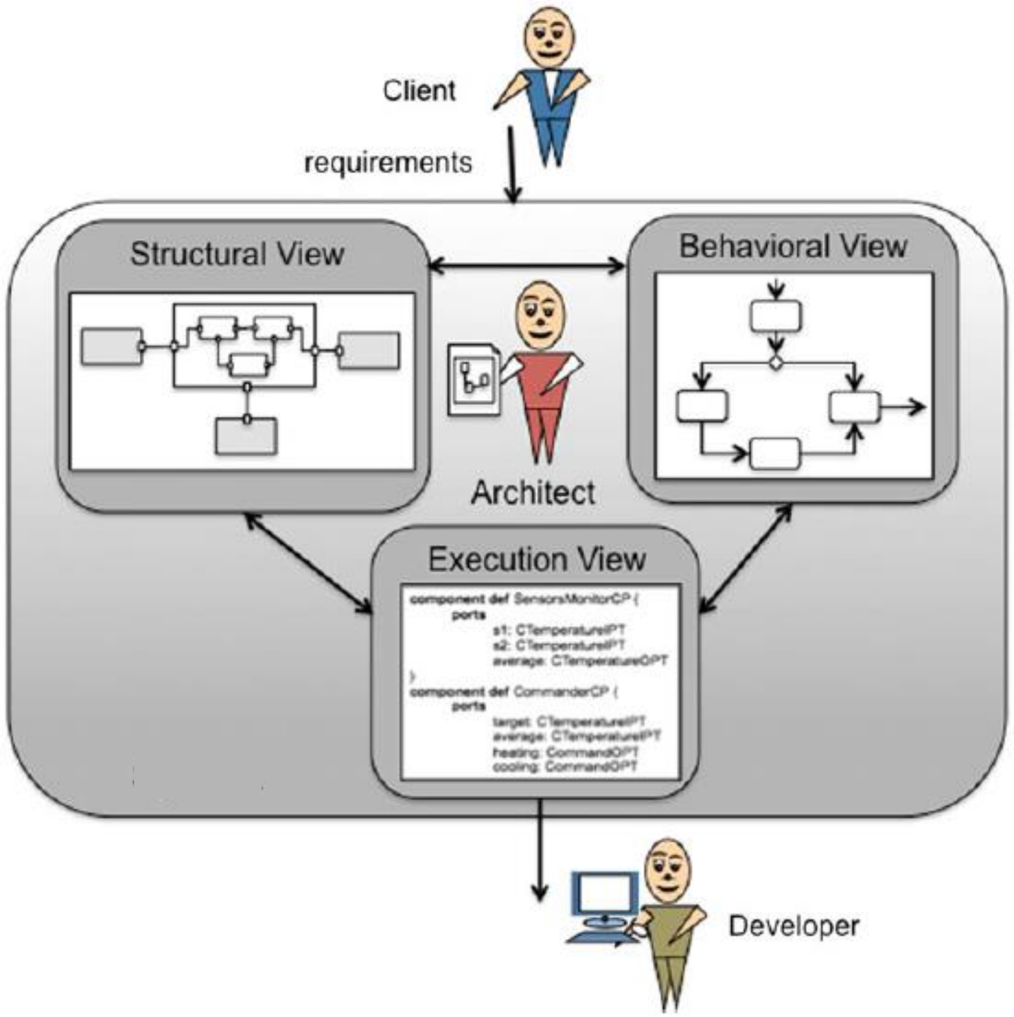
- **The way** in that the elements perform activities and **interact** with each other to achieve the required system functionality
- Main view element: **behavior** of architectural elements

Architectural views vs. Architectural viewpoints

Other proposed viewpoints


Undergraduate Topics in Computer Science
Flavio Oquendo
Jair Leite
Thaís Batista
Software Architecture in Action
Designing and Executing Architectural Models with SysADL grounded on the OMG SysML Standard
UTICS
EXTRAS ONLINE
Springer

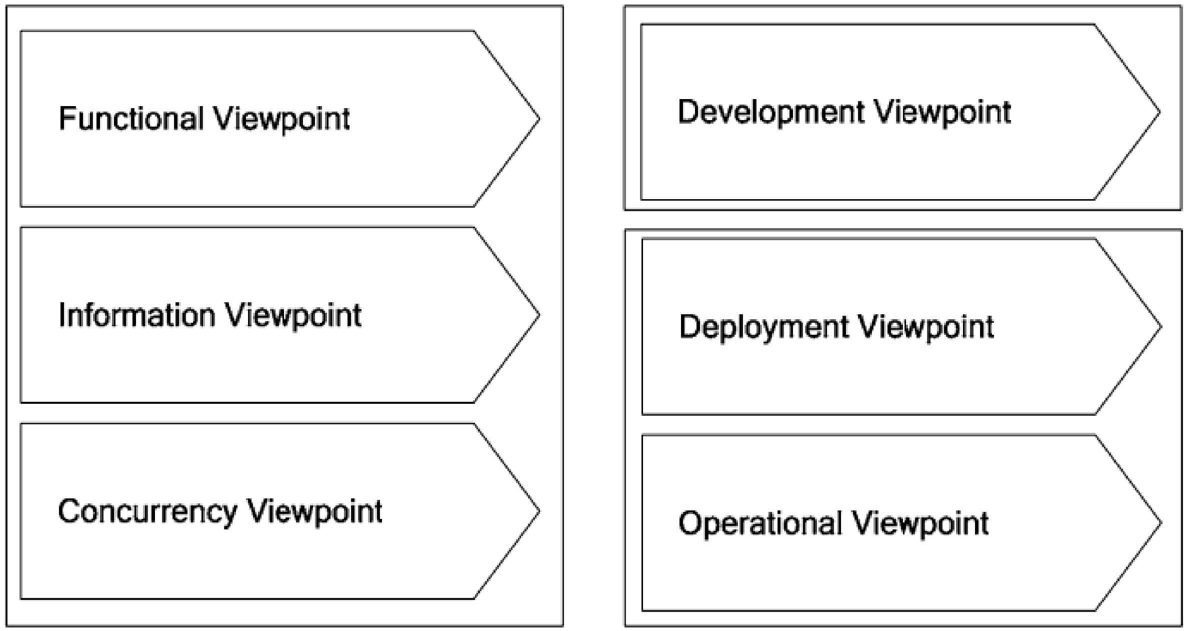
Flavio Oquendo, Jair Leite,
Thais Batista. **Software Architecture in action: Designing and executing architectural models with SysADL grounded on the OMG SysML Standard.** Switzerland:
Springer International Publishing, 2016



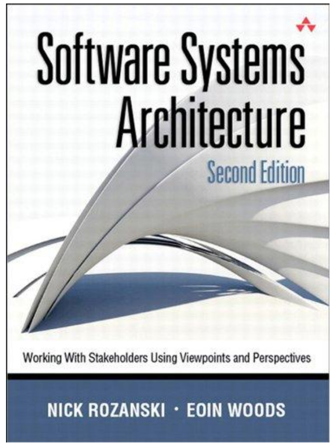
```
component def SensorsMonitorCP {
        ports
                s1: CTemperatureIPT
                s2: CTemperatureIPT
                average: CTemperatureOPT
}
component def CommanderCP {
        ports
                target: CTemperatureIPT
                average: CTemperatureIPT
                heating: CommandOPT
                cooling: CommandOPT
```

Architectural views vs. Architectural viewpoints

Other proposed viewpoints



Nick Rozanski, Eóin Woods. **Software systems architecture: Working with stakeholders using viewpoints and perspectives – 2nd ed.** USA: Addison-Wesley/Pearson Education, Inc., 2012


Software Systems Architecture
Second Edition
Working With Stakeholders Using Viewpoints and Perspectives
NICK ROZANSKI · EOIN WOODS

Architectural views vs. Architectural viewpoints

The benefits and pitfalls of using architectural views and viewpoints

+

- Clearer separation of concerns
- Suitable communication with different stakeholders
- Better management of complexity
- Improved developer focus

−

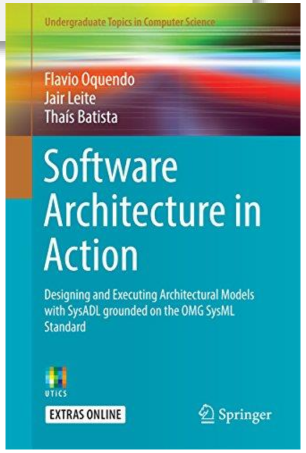
- Inconsistency among views, which need to be manually checked
- Selection of an unsuitable set of views
- Fragmentation
- Significant creation and maintenance effort

Architectural styles

Architectural style

can be seen as a collection of **principles shaping the design of a software architecture** to achieve a set of related quality attributes

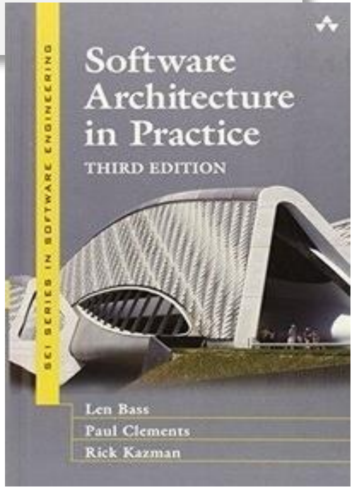
Flavio Oquendo, Jair Leite, Thais Batista. **Software Architecture in action: Designing and executing architectural models with SysADL grounded on the OMG SysML Standard.** Switzerland: Springer International Publishing, 2016


Undergraduate Topics in Computer Science
Flavio Oquendo
Jair Leite
Thaís Batista
Software Architecture in Action
Designing and Executing Architectural Models with SysADL grounded on the OMG SysML Standard
EXTRAS ONLINE
Springer

Architectural styles

Architectural style

is a specialization of **elements and relation types**, together with a set of **constraints on how they can be used**

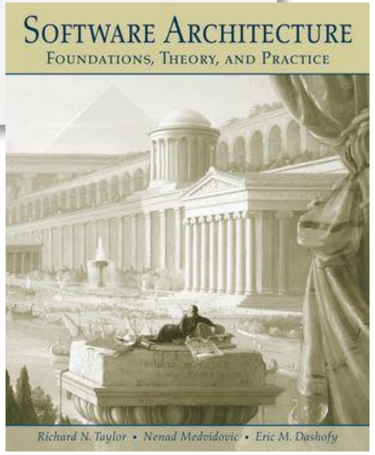

Software Architecture in Practice
THIRD EDITION
SEI SERIES IN SOFTWARE ENGINEERING
Len Bass
Paul Clements
Rick Kazman

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

Architectural styles

Architectural style

is a named collection of **architectural design decisions** that (1) are applicable in a given development context, (2) constrain architectural design decisions that are specific to a particular system within that context, and (3) elicit beneficial qualities in each resulting system



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

Architectural styles

An architectural style

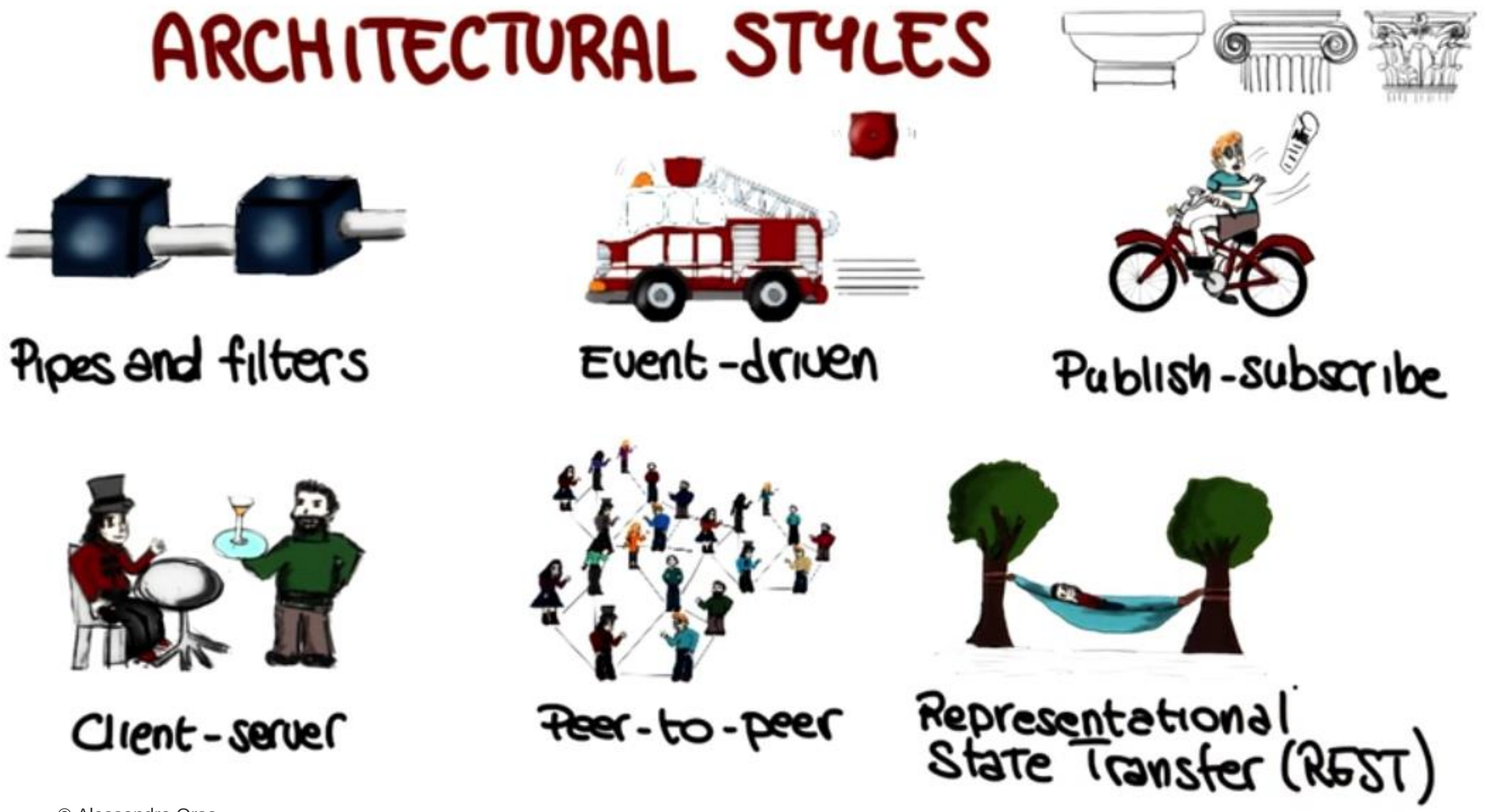
– is expressed by a vocabulary of element types determining the allowed arrangements in terms of topology, behavior, communication, etc.

– guides how to organize the elements of an architected system so that one can design the architecture

– is used to derive instances of the architecture, which will be characteristics in common as they follow the style

– is used in conformance to requirements

– can be implemented in several ways when concretizing the architecture



© Alessandro Orso

Architectural styles

Name	Suitable for	Elements	Constraints
Pipe-Filter	Sequential data processing	• *Filters*: components that read streams of data on their inputs and produces streams of data on their outputs • *Pipes*: connectors that transmit output streams of one filter to inputs of another	There must be at least one pipe connecting an output of a filter to an input of another filter
Layered (*n*-tiers)	Decomposition and assignment of functionalities to different, hierarchical parts	*Layers* have well-defined functionalities	A given layer provides services to the layer above and uses services provided by the layer below

Architectural styles

Name	Suitable for	Elements	Constraints
Client-Server	Provision and consumption of services	• *Clients*: components that send service requests and may receive service responses from the servers • *Servers*: components that receive service requests, process them, and may send service responses to clients	A client cannot directly connect to another client (except if it also acts as a server), but only with a server
Service-Oriented Architecture (SOA)	Service-oriented networked computing	Interoperable, loosely coupled *services* provide functionalities and may be discovered and composed to offer other value-added functionalities	Services must publish well-defined interfaces

Architectural styles

Name	Suitable for	Elements	Constraints
Blackboard	Data structure sharing	• *Blackboard*: central component that aggregates shared information • *Knowledge sources*: components that provide and/or consume information available at the Blackboard	Interactions among knowledge sources are coordinated by the *Blackboard* component
Representational State Transfer (REST)	Design of loosely coupled distributed applications over the HTTP protocol	• *Clients*: components that send requests to access resources deployed in servers • *Servers*: components that receive requests, process them, and return representations of deployed resources	• Uniform interface for uniquely identifying resources • Statelessness • Cacheability

Architectural patterns

Designed architecture

Implemented architecture

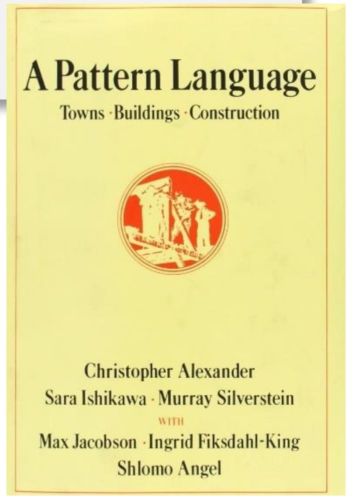
Expertise is required

Architectural patterns

Pattern

describes a **problem** which occurs **over and over again** in our **environment**, and then describes the **core of the solution** to that problem, in such a way that you can **use this solution a million times over**, **without ever doing it the same way twice**

Christopher Alexander, Sara Ishikawa, Murray Silverstein, Max Jacobson, Ingrid Fiksdahl-King, Shlomo Angel. **A Pattern Language.** USA: Oxford University Press, 1977

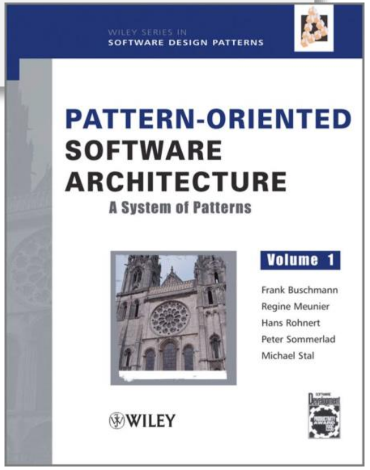

A Pattern Language
Towns · Buildings · Construction
Christopher Alexander
Sara Ishikawa · Murray Silverstein
WITH
Max Jacobson · Ingrid Fiksdahl-King
Shlomo Angel

Architectural patterns

Pattern

addresses a **recurring design problem** that arises in specific design situations and presents a **solution** to it. […] Patterns document **existing, well-proven design experience**. […] Patterns identify and specify **abstractions** that are above the level of single classes and instances, or of components. […] Patterns help you to **manage software complexity**.

Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, Michael Stal. **Pattern-Oriented Software Architecture – Volume 1: A system of patterns.** United Kingdom: John Wiley & Sons, Inc., 1996.

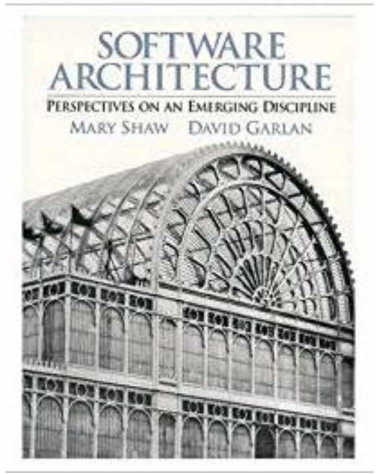


Patterns help to
not reinvent the wheel

Don't reinvent the wheel!

Architectural patterns vs. Architectural styles

The Shaw and Garlan’s book on Software Architecture has attempted to catalogue some architectural styles aiming at defining some taxonomies to organize them



Mary Shaw, David Garlan. **Software Architecture: Perspectives on an emerging discipline.** USA: Prentice-Hall, 1996.

Architectural patterns vs. Architectural styles

The identified architectural styles have evolved over time and now are often confused with some existing architectural patterns, but they are different concepts in terms of scope

– Patterns are more detailed than styles, with a well-defined body

– Patterns capture context, constraints, and effects rather than only architectural elements and their organization

– Patterns explain their rationale, *raison d'être*

– Patterns allow for reusing experience for solving specific problems

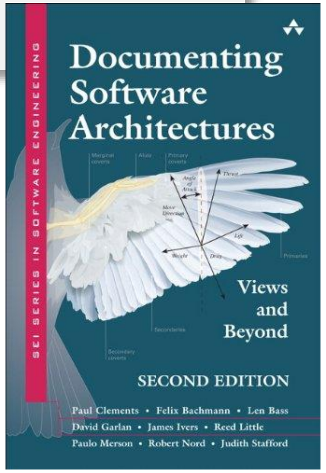
– Patterns concern solutions to problems whereas styles do not

Architectural patterns vs. Architectural styles

Architectural pattern vs. Architectural style

An essential part of an architecture pattern is its **focus on the problem and context** as well as **how to solve the problem** in that context. An architecture style **focuses on the architecture approach**, with more **lightweight guidance** on when a particular style may or may not be useful

Paul Clements et al.
Documenting software architectures: Views and beyond – 2nd ed. USA: Addison-Wesley/Pearson Education, Inc., 2011


Documenting Software Architectures
Views and Beyond
SECOND EDITION
SEI SERIES IN SOFTWARE ENGINEERING
Paul Clements • Felix Bachmann • Len Bass
David Garlan • James Ivers • Reed Little
Paulo Merson • Robert Nord • Judith Stafford

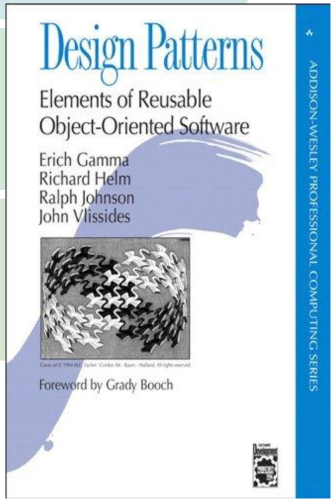
Architectural patterns vs. Design patterns

- Architectural patterns are similar to software design patterns
  - Both aggregate well-proven experience
  - Architectural patterns have a broader scope as they concern the software architecture and its elements
- The existing types of design patterns reveal their difference in comparison to architectural patterns
  - Design patterns concern lower level object-oriented elements (classes, objects, etc.)
  - Architectural patterns concern coarse-grained elements (such as components)

Architectural patterns vs. Design patterns

Categories of design patterns

Creational patterns	describe how to instantiate objects that are part of the problem's context
Structural patterns	describe how to compose entities (classes, objects) to form larger structures
Behavioral patterns	describe interactions among objects focusing on how they communicate with each other
Concurrency patterns	describe how to design programs with multiple concurrent processes/threads

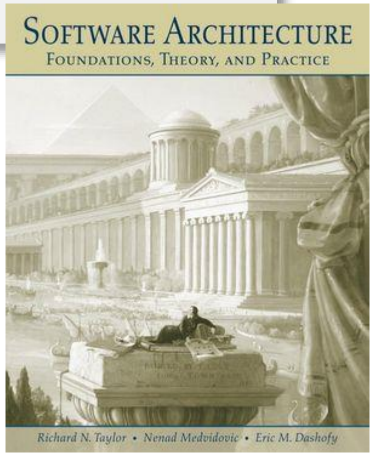

Design Patterns
Elements of Reusable Object-Oriented Software
Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides
Foreword by Grady Booch
ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES

Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides. **Design patterns: Elements of reusable object-oriented software.** USA: Addison-Wesley, 1995

Architectural patterns

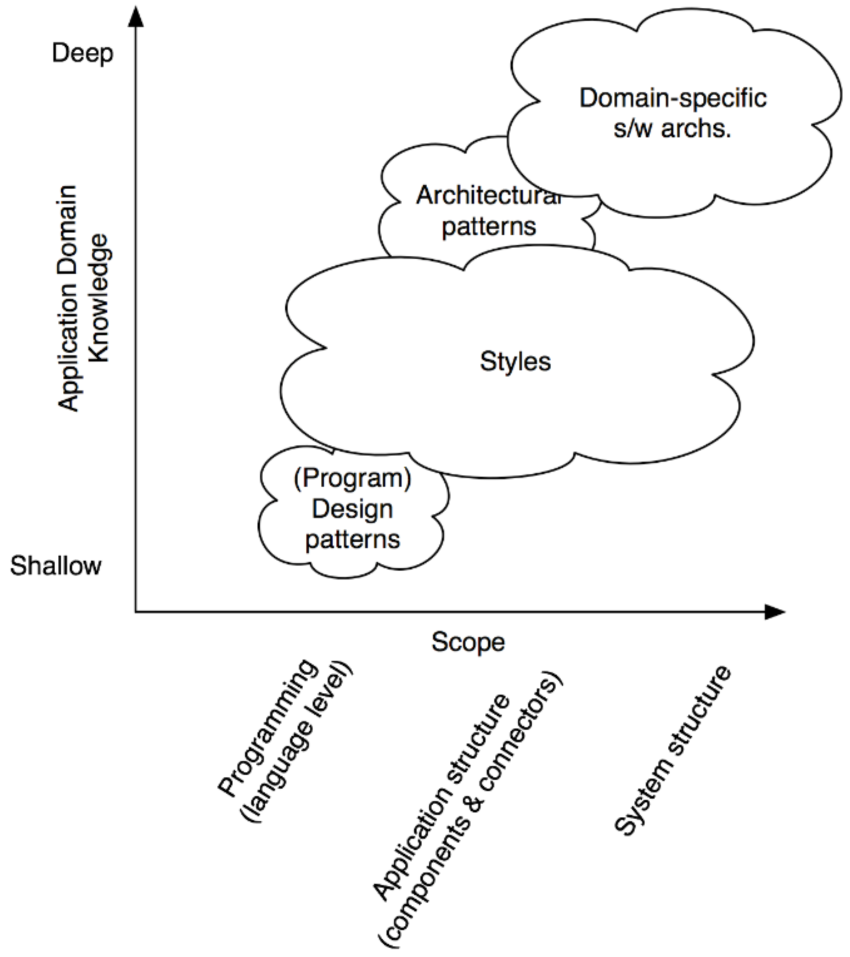
Architectural pattern

is a named collection of **architectural design decisions** that are applicable to a **recurring design problem**, parameterized to account for different software development contexts in which that problem appears

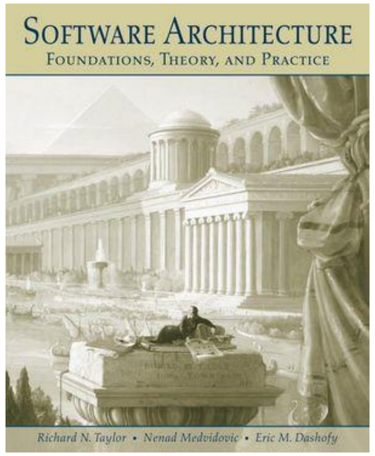

SOFTWARE ARCHITECTURE
FOUNDATIONS, THEORY, AND PRACTICE
Richard N. Taylor • Nenad Medvidovic • Eric M. Dashofy

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

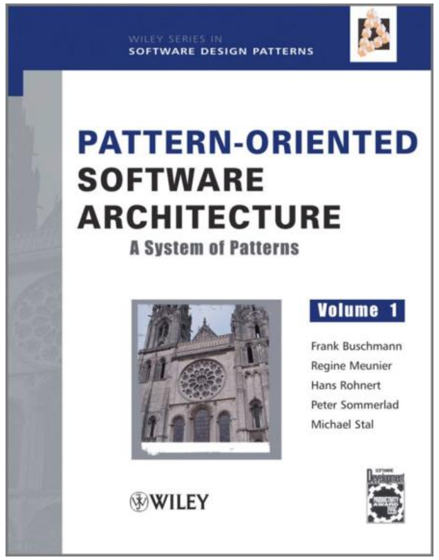
Architectural patterns


Deep
Application Domain Knowledge
Shallow
Domain-specific s/w archs.
Architectural patterns
Styles
(Program) Design patterns
Scope
Programming (language level)
Application structure (components & connectors)
System structure

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



Architectural patterns


WILEY SERIES IN SOFTWARE DESIGN PATTERNS

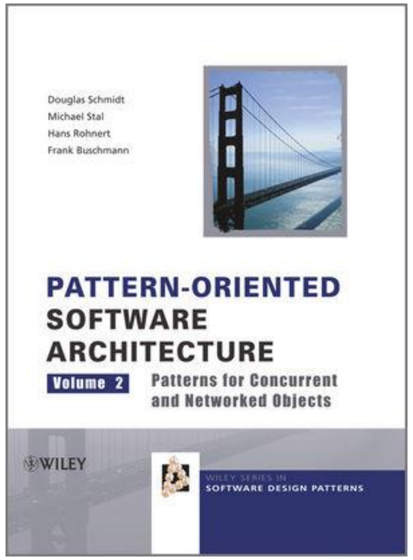
PATTERN-ORIENTED SOFTWARE ARCHITECTURE

A System of Patterns

Volume 1

Frank Buschmann
Regine Meunier
Hans Rohnert
Peter Sommerlad
Michael Stal

WILEY

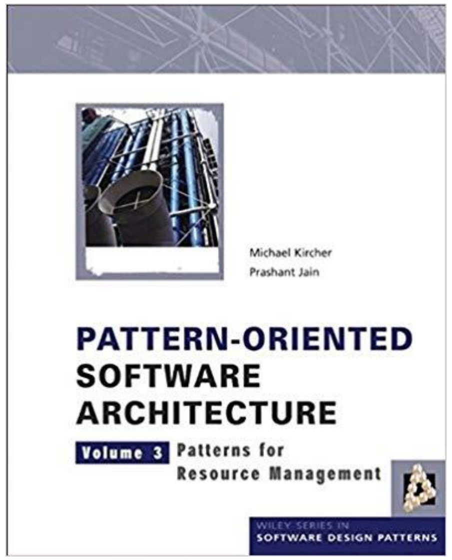

Douglas Schmidt
Michael Stal
Hans Rohnert
Frank Buschmann

PATTERN-ORIENTED SOFTWARE ARCHITECTURE

Volume 2 Patterns for Concurrent and Networked Objects

WILEY

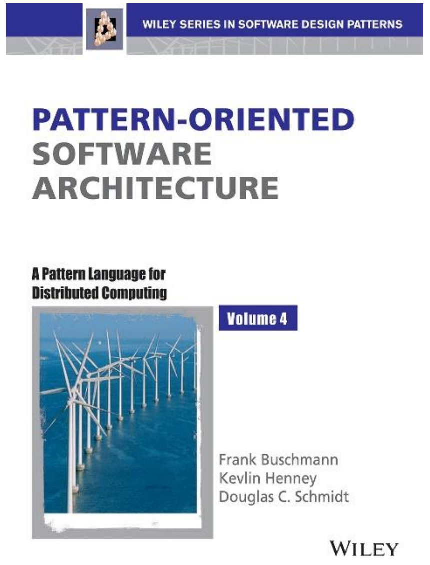
WILEY SERIES IN SOFTWARE DESIGN PATTERNS


Michael Kircher
Prashant Jain

PATTERN-ORIENTED SOFTWARE ARCHITECTURE

Volume 3 Patterns for Resource Management

WILEY SERIES IN SOFTWARE DESIGN PATTERNS


WILEY SERIES IN SOFTWARE DESIGN PATTERNS

PATTERN-ORIENTED SOFTWARE ARCHITECTURE

A Pattern Language for Distributed Computing

Volume 4

Frank Buschmann
Kevlin Henney
Douglas C. Schmidt

WILEY

WILEY SERIES IN SOFTWARE DESIGN PATTERNS

PATTERN-ORIENTED SOFTWARE ARCHITECTURE

On Patterns and Pattern Languages

Volume 5

Frank Buschmann
Kevlin Henney
Douglas C. Schmidt

Architectural patterns

Name	Suitable for	Organization
Broker	Distributed systems with decoupled components	A *broker* component is responsible for coordinating remote service invocations among components, which interact only with it
Proxy	Protection of direct access to a resource/service	Clients communicate with a *proxy* component as if they were communicating with the actual service/resource provider
Pipe-Filter	Sequential data processing	*Filter* components process data streams transmitted through *pipe* connectors
Client-Server	Request-based provision and consumption of services	A *server* component provides services consumed by *client* components

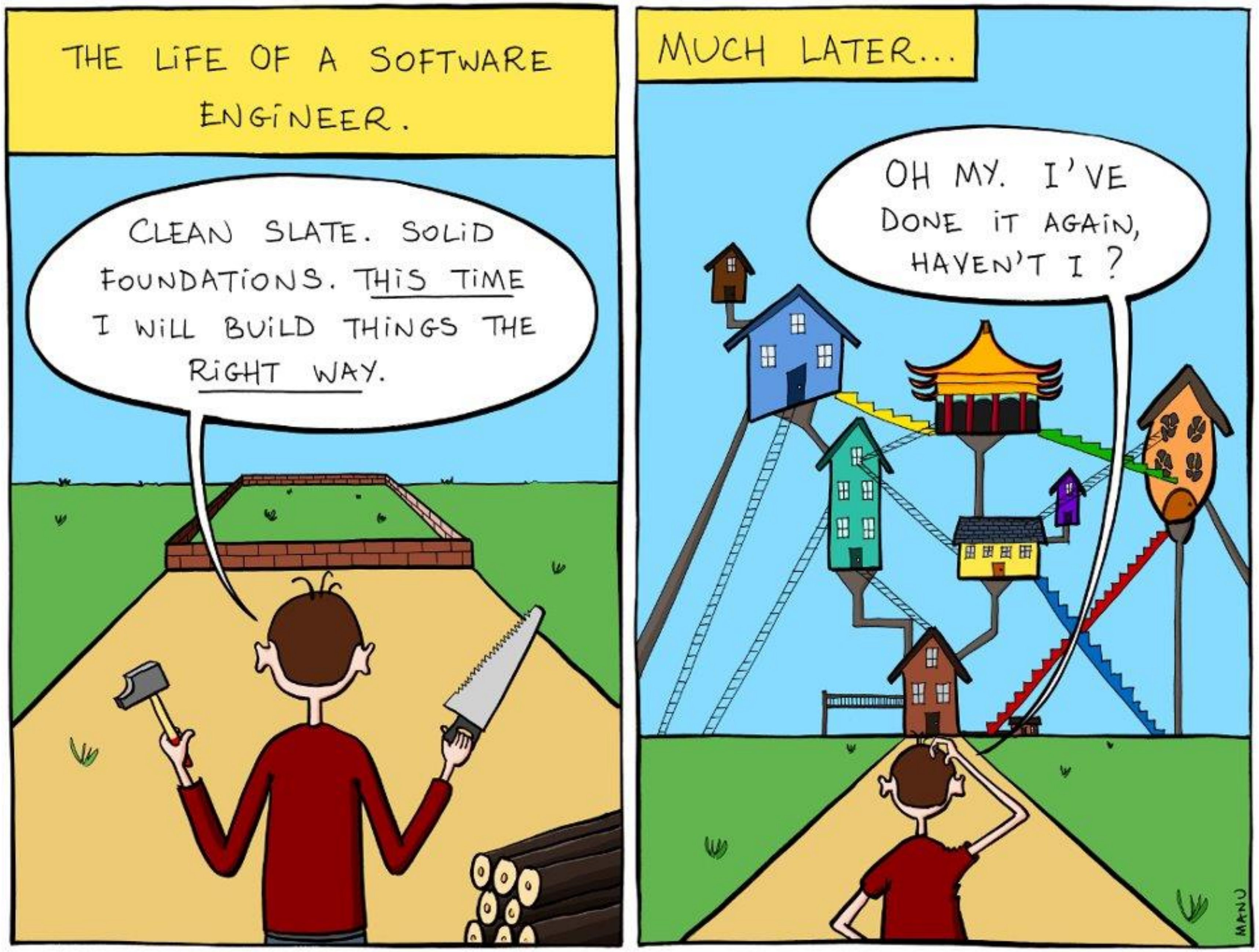
Architectural patterns

Name	Suitable for	Organization
Layered (*n*-tier)	Systems in which functionalities are decomposed and modularly assigned to different parts	Each *layer* has a well-defined functionality, providing services to the layer above and using services provided by the layer below
Master-Slave	Delegation of functionalities	The *master* component distributes the work among identical *slave* components and computes the final result
Model-View-Controller (MVC)	Functionality division of interactive applications	The *view* component displays information to the user, whose interaction is managed by the *controller* component while core functionality and data are within the *model* component

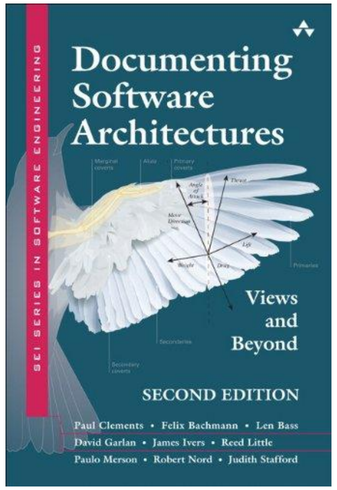
Architectural patterns

Name	Suitable for	Organization
Publish-Subscribe	Communication among distributed decoupled components	*Publisher* components publish messages to an *event bus*, to which *subscribers* subscribe aiming at receiving notification messages complying with their interest

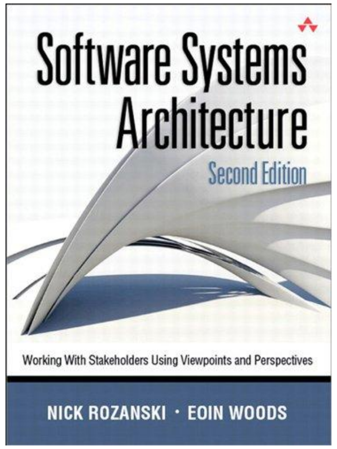
The take away message


THE LIFE OF A SOFTWARE ENGINEER.
CLEAN SLATE. SOLID FOUNDATIONS. THIS TIME I WILL BUILD THINGS THE RIGHT WAY.
MUCH LATER...
OH MY. I'VE DONE IT AGAIN, HAVEN'T I ?
MANU

Further reading

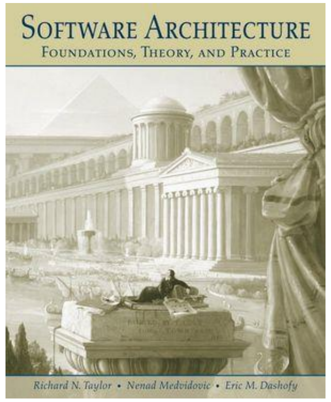


Paul Clements et al. Documenting software architectures: Views and beyond – 2nd ed. USA: Addison-Wesley/ Pearson Education, Inc., 2011

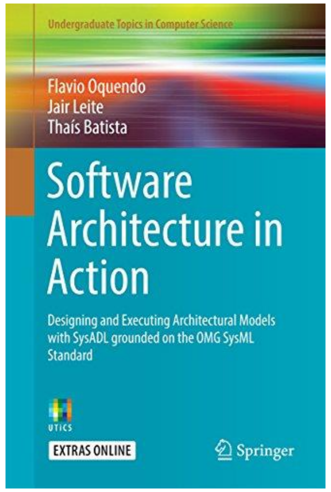


Nick Rozanski, Eóin Woods. Software systems architecture: Working with stakeholders using viewpoints and perspectives – 2nd ed. USA: Addison-Wesley/Pearson Education, Inc., 2012

Further reading

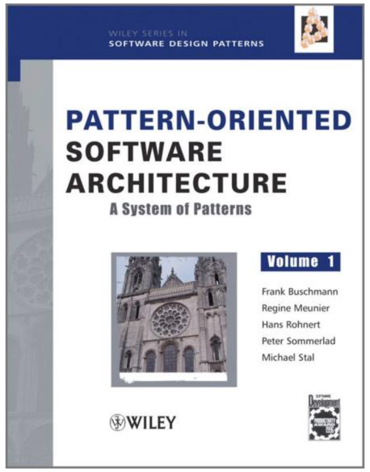


Richard N. Taylor, Nenad Medvidovic, Eric M. Dashofy. Software Architecture: Foundations, theory, and practice. USA: John Wiley & Sons, Inc., 2010



Flavio Oquendo, Jair Leite, Thais Batista. Software Architecture in action: Designing and executing architectural models with SysADL grounded on the OMG SysML Standard. Switzerland: Springer International Publishing, 2016

Further reading



Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, Michael Stal. Pattern-Oriented Software Architecture – Volume 1: A system of patterns. United Kingdom: John Wiley & Sons, Inc., 1996

Architectural views, styles, and patterns

Prof. Dr. Everton Cavalcante

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