

# Architectural Views

CSSE6400

Richard Thomas

March 7, 2022

*Interesting* Software is Complex

Many aspects to the design of its architecture.

Architectural Design

Managing technical complexity.

### Question

How do you describe a complex architecture, without making it too difficult to understand?

### Question

How do you describe a complex architecture, without making it too difficult to understand?

### Answer

#### *Architectural Views*

- Only consider one aspect at a time.

# Architectural Views

- 4+1 Views [1]
  - logical, process, development, physical, scenario

# Architectural Views

- 4+1 Views [1]
  - logical, process, development, physical, scenario
- Software Architecture in Practice [2]
  - module, component-and-connector, allocation

# Architectural Views

- 4+1 Views [1]
  - logical, process, development, physical, scenario
- Software Architecture in Practice [2]
  - module, component-and-connector, allocation
- Rozanski and Woods [3]
  - context, building block, runtime, deployment



# Architectural Views

- 4+1 Views [1]
  - logical, process, development, physical, scenario
- Software Architecture in Practice [2]
  - module, component-and-connector, allocation
- Rozanski and Woods [3]
  - context, building block, runtime, deployment
- NATO Architecture Framework [4]
  - concepts, service, logical, physical resource, architecture foundation

# Architectural Views

- 4+1 Views [1]
  - logical, process, development, physical, scenario
- Software Architecture in Practice [2]
  - module, component-and-connector, allocation
- Rozanski and Woods [3]
  - context, building block, runtime, deployment
- NATO Architecture Framework [4]
  - concepts, service, logical, physical resource, architecture foundation
- The Open Group Architecture Framework (TOGAF) [5]
- ISO/IEC/IEEE 42010:2011 [6]

## 4+1 Views

Logical – *Structure* of how the software is implemented.

- components/classes, relationships, interactions

## 4+1 Views

**Logical** – *Structure* of how the software is implemented.

- components/classes, relationships, interactions

**Process** – *Dynamic* behaviour.

- concurrency & distribution, fault tolerance, process control, ...

## 4+1 Views

**Logical** – *Structure* of how the software is implemented.

- components/classes, relationships, interactions

**Process** – *Dynamic* behaviour.

- concurrency & distribution, fault tolerance, process control, ...

**Development** – *Organisation* of the software in the development environment.

## 4+1 Views

**Logical** – *Structure* of how the software is implemented.

- components/classes, relationships, interactions

**Process** – *Dynamic* behaviour.

- concurrency & distribution, fault tolerance, process control, ...

**Development** – *Organisation* of the software in the development environment.

**Physical** – *Map* executable software containers to hardware.

- address non-functional requirements
  - availability, reliability, scalability, throughput, ...

## 4+1 Views

**Logical** – *Structure* of how the software is implemented.

- components/classes, relationships, interactions

**Process** – *Dynamic* behaviour.

- concurrency & distribution, fault tolerance, process control, ...

**Development** – *Organisation* of the software in the development environment.

**Physical** – *Map* executable software containers to hardware.

- address non-functional requirements
  - availability, reliability, scalability, throughput, ...

**Scenario** – *Demonstrate* functionality delivered by architecture.

- use case details
  - *drive* functional design of architecture
  - *validate* design of architecture
  - *illustrate* purpose of architecture

# Diagrams & Notation

- A *good* diagram is worth a thousand words.
  - A thousand diagrams is just confusing.



## Diagrams & Notation

- A *good* diagram is worth a thousand words.
  - A thousand diagrams is just confusing.
- UML – formal, well-defined language [7]
- C4 – informal, simple structure [8]
- You probably don't want to know about alternatives.

Reading...

# “Architectural Views” Notes<sup>1</sup> [9]

---

<sup>1</sup>Remember, I said you had to read the notes.

## References

[1] Philippe Kruchten.

*Architectural blueprints — the ‘4+1’ view model of software architecture.*

*IEEE software*, 12(6):42–50, 1995.

[https:](https://www.cs.ubc.ca/~gregor/teaching/papers/4+1view-architecture.pdf)

[//www.cs.ubc.ca/~gregor/teaching/papers/4+1view-architecture.pdf](https://www.cs.ubc.ca/~gregor/teaching/papers/4+1view-architecture.pdf).

[2] Len Bass, Paul Clements, and Rick Kazman.

*Software Architecture in Practice.*

Addison-Wesley, 4th edition, August 2021.

[3] Nick Rozanski and Eóin Woods.

*Software Systems Architecture: Working With Stakeholders Using Viewpoints and Perspectives.*

Addison-Wesley, 2nd edition, 2012.

- [4] Architecture Capability Team.  
*NATO Architecture Framework.*  
NATO, 4th edition, September 2020.
  
- [5] The Open Group Architecture Forum.  
*The Open Group Architecture Framework Standard.*  
The Open Group, 9.2 edition, 2018.  
<https://pubs.opengroup.org/architecture/togaf9-doc/arch/index.html>.
  
- [6] *ISO/IEC/IEEE 42010:2011.*  
*ISO, 2011.*
  
- [7] *Unified Modeling Language.*  
*OMG, 2.5.1 edition, December 2017.*  
<https://www.uml.org/>.

[8] Simon Brown.

*Software Architecture for Developers - Volume 2.*

Leanpub, January 2022.

<https://leanpub.com/visualising-software-architecture>.

[9] Richard Thomas and Brae Webb.

*Architectural views.*

March 2022.

<https://csse6400.uqcloud.net/handouts/views.pdf>.