Statements and Consistency

Design of Software Architectures

Dr. Vadim Zaytsev aka @grammarware, 22 September 2023

UNIVERSITY OF TWENTE.

Role of the software architecture

to provide

solution direction

for

most important properties

that are

the most difficult to realise

Why should you architect?

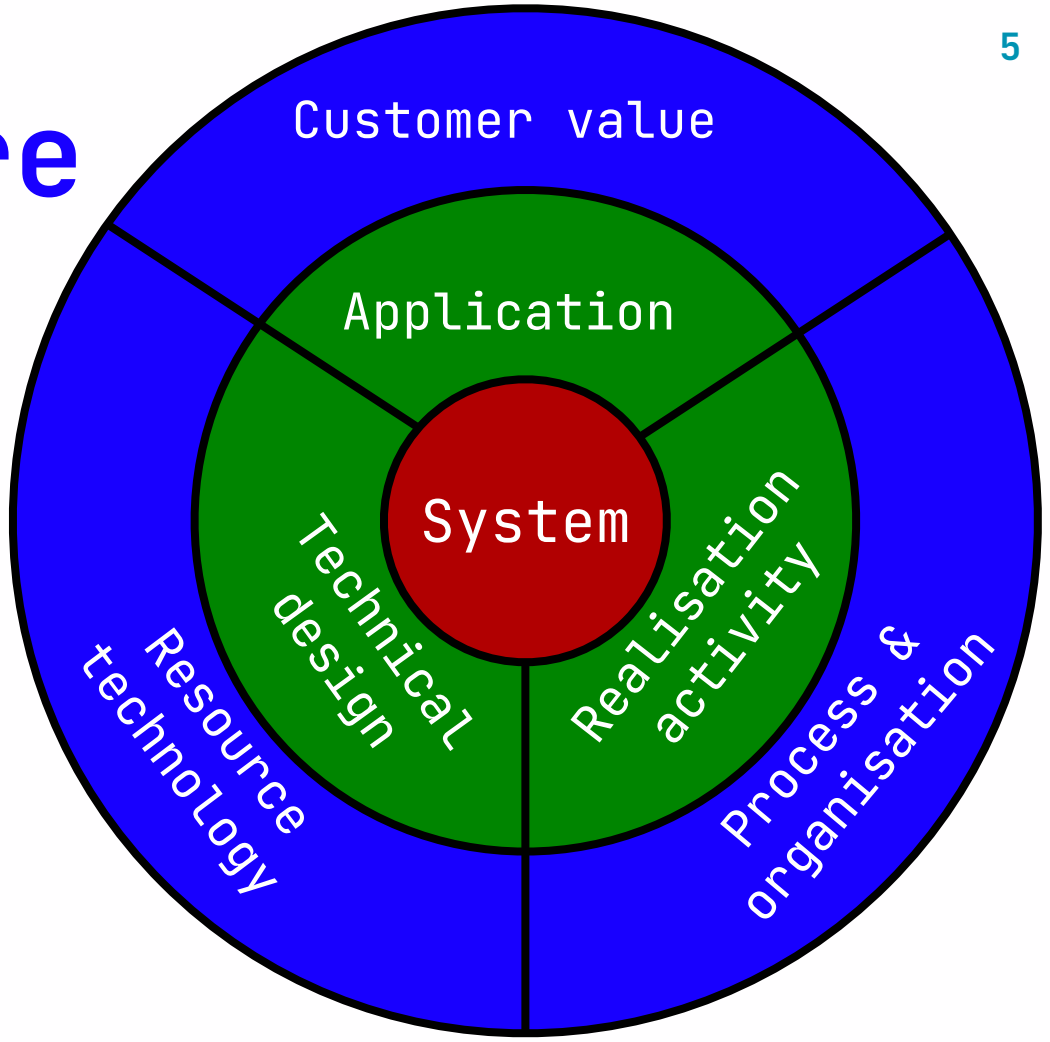
(why provide directions for complicated properties)

- make the system maintainable
- try to prevent problems before they occur
- make properties concrete
- keep *IT* consistent
- create structure in software
- keep the complexity as low as possible
- keep production cost low
- to be paid

What happens if you don’t?

- unmaintainable mess
- not meeting deadlines
- miss core concerns and requirements
- ???

Software architecture



- The software architecture of a system is
  - a collection of statements
  - that gives direction to
    - the design
    - the realisation and
    - the evolution
  - of the software in its environment
- Statements are a model of:
  - structure of the system elements and their relations, or
  - guidelines for creating structure, elements and their relationships

Statements are about aspects

- 1-1..*
- Aspects are coming from the domain
- The architect
  - thinks about aspects
  - communicates about aspects
  - reasons about aspects
  - is responsible for a set of aspects

What is a specification?

What is a specification?

- Formal specification:

```
VARIABLE clock
Init == clock \in {0, 1}
Tick == IF clock = 0 THEN clock' = 1 ELSE clock' = 0
Spec == Init /\ [][Tick]_<<clock>>
```

https://en.wikipedia.org/wiki/TLA+

```
System simple_cs = {
  Component client = {
     Port sendRequest;
     Properties { requestRate : float = 17.0;
                  sourceCode : externalFile = "CODE-LIB/client.c" }}
  Component server = {
     Port receiveRequest;
     Properties { idempotent : boolean = true;
                  maxConcurrentClients : integer = 1;
                  multithreaded : boolean = false;
                  sourceCode : externalFile = "CODE-LIB/server.c" }}
  Connector rpc = {
     Role caller;
     Role callee;
     Properties { synchronous : boolean = true;
                  maxRoles : integer = 2;
                  protocol : WrightSpec = "..." }}
  Attachments { client.send-request to rpc.caller ;
                server.receive-request to rpc.callee }
}
```

Garlan, *Formal Modeling and Analysis of Software Architecture: Components, Connectors, and Events*, SFM 2003

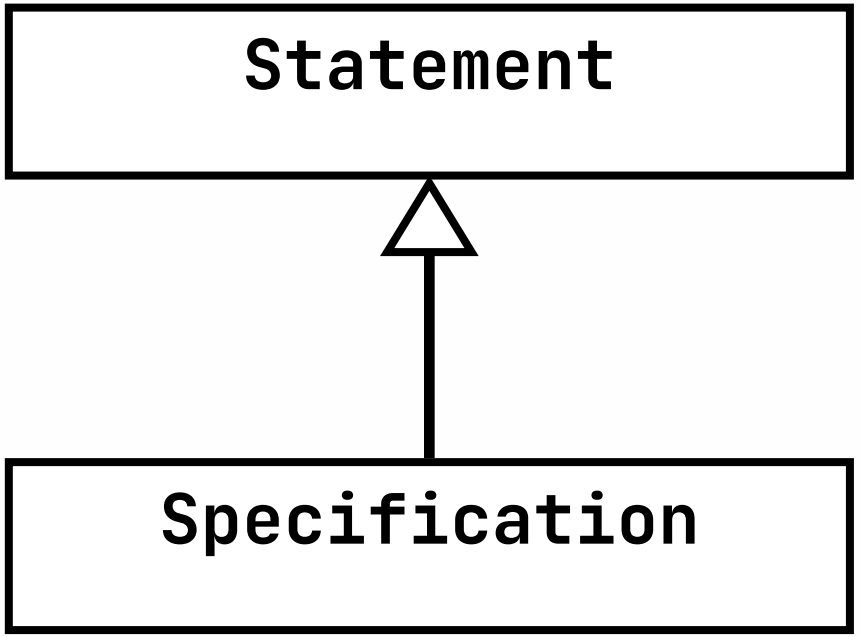
What is a specification?

- Formal specification
- “Specification” aka documentation/standard
  - requirement specification
  - functional specification
  - language/format/protocol specification
  - architecture specification

https://en.wikipedia.org/wiki/TLA+

What is a specification?

- Formal specification
- “Specification” aka documentation/standard
- Specification is a statement
  - about some aspects
  - has defined context
  - is assignable



Recap

- **Domain** is an area of coherent activity
  - a problem space that the software system addresses
- **Aspect** is a different perspective within a domain
  - collectively forming a complete picture
- **Concept** is a element of a domain or an aspect
  - can be shared
  - contribute to the overall design

Identify aspects in statements

Boldly can predict traffic patterns, peak hours and potential bottlenecks.

WiseWorld aids in dispatching resources effectively and communicates critical updates to residents by emergency.

WiseWorld is an evolving solution that needs regular updates, maintenance and hardware upgrades.

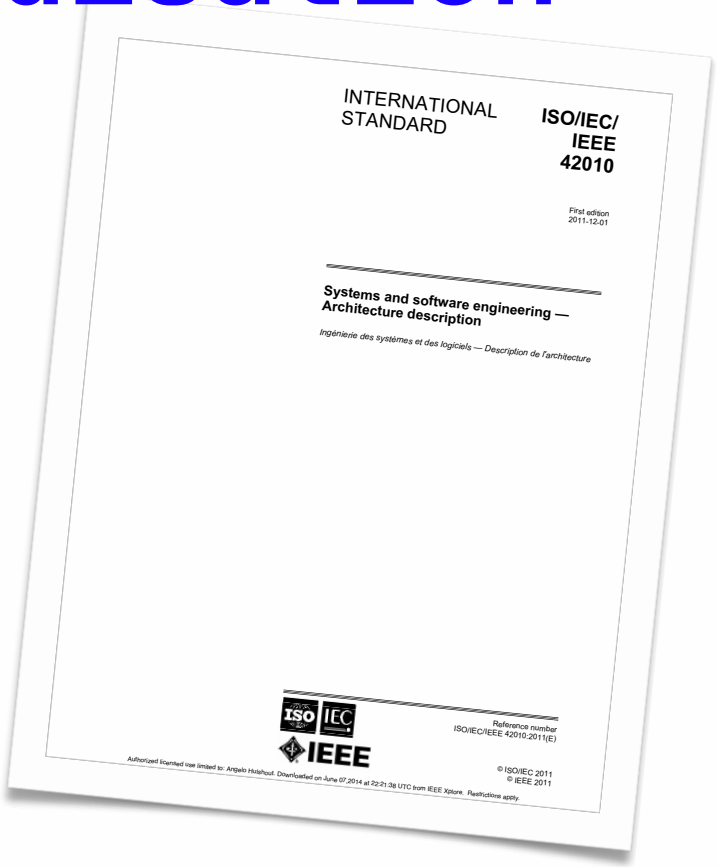
Boldly implements sophisticated dynamic and adaptive traffic signal control algorithms.

Keep track of specifications

Keep track of all the specifications that you use and introduce

- Identification (name, version, source)
- Relation to the architecture:
  - For what architecture element is it used?
  - How does it use the architecture?
- Dependencies on other specifications
  - (graph) specifications are nodes, relations are arrows
  - (table) a row for each specification

Specification ~ Standardisation


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

Reference number ISO/IEC/IEEE 42010:2011(E)

- Consistently specific everywhere
  - ⇝ standardised
- ISO 22737:2021: LSAD
- ERTMS
- GDPR
- All standards you use are (external) specifications

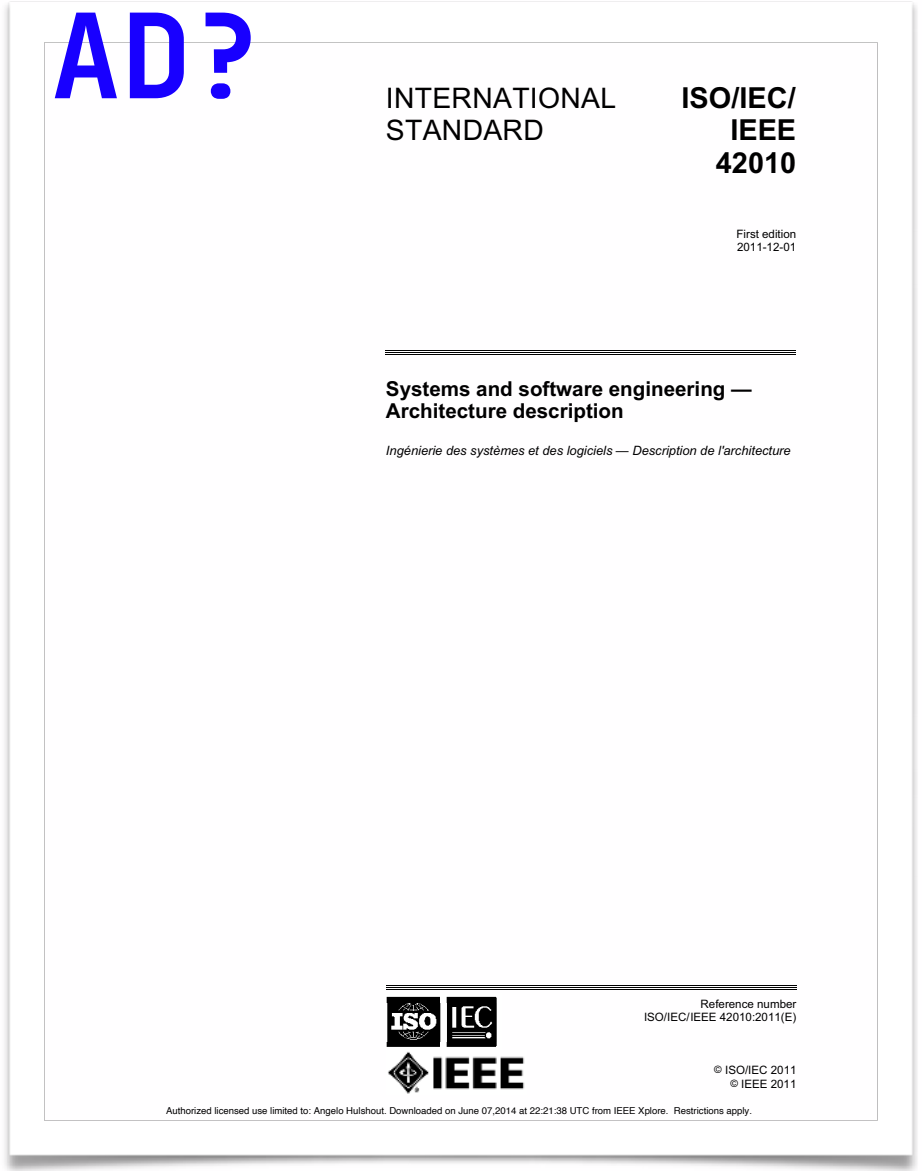
Specification properties

- Specifications have a validity
  - wish – intention – requirement – assumption – proven
- Specifications have a scope
  - system
  - system element
  - environment element
  - range of the above
- Direct relations
  - composed of other specifications
  - derived from / based on other specifications
  - made according to other specifications
- Indirect relations via shared aspects or shared context

What should be in the AD?

What should be in the AD?

- Introduction
- Identification & overview
- Stakeholders & concerns
- Architecture viewpoints
- Views & models & relations
- Architecture rationale
  - including decisions



INTERNATIONAL STANDARD

ISO/IEC/ IEEE 42010

First edition
2011-12-01

Systems and software engineering — Architecture description

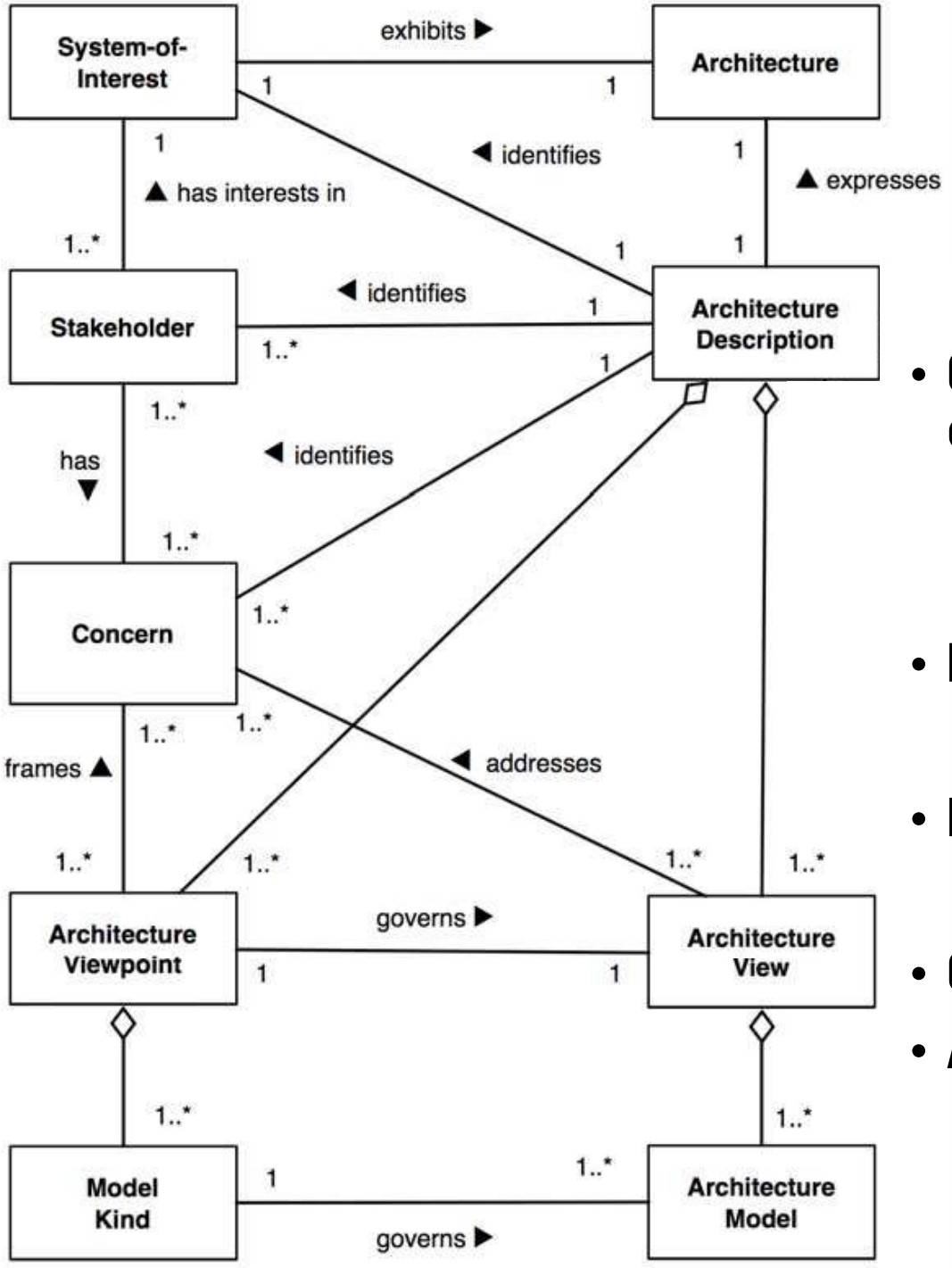
Ingénierie des systèmes et des logiciels — Description de l'architecture

Reference number
ISO/IEC/IEEE 42010:2011(E)

Intended audience?

- Those who use, own or acquire the system
  - clients
- Those who develop, describe and document architectures
  - architects
- Those who develop, deliver and maintain the system
  - devs, designers, coders, QA, testers
- Those who oversee & evaluate systems
  - auditors

AD should contain…



How to use 42010?

- Conceptual model provides architecting guidance
  - Completeness analysis
  - Consistency analysis
- Basis for other architecture concepts
  - Extend it!
- Basis for documentation methods
  - (e.g. templates)
- Content listings are check points
- A guideline, not a law

Filing 42010 template ≠ architecting

- No architecture creation/reasoning → decision
- No architecting as activity → process
- No architecting by reference → pattern
- No general specification: → statement
  → specification
  → aspect

- Use an extended subset!

Revisit the document

- Use an extended subset of ISO/IEC/IEEE 42010
- Is the scope defined? Do you communicate the context?
- Do you have enough stakeholders? Concerns?
- Are stakeholders linked to concerns?
- Do you have all the viewpoints?
- One view per viewpoint