Architecting from Domains to Aspects, from Models to Views

Design of Software Architectures

Dr. Vadim Zaytsev aka @grammarware, 20 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

Identify Aspects, Functionalities, Components

5 minutes

Aspects everywhere

Definitions

- **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

Typical application aspects

- Legal compliance, actuality, traceability
- Social responsibility, impartiality
- Credibility, transparency, money traceability
- Safety, customer support, energy consumption
- Duration, eco footprint, yield, continuity
- Energy, availability, reliability, maintainability

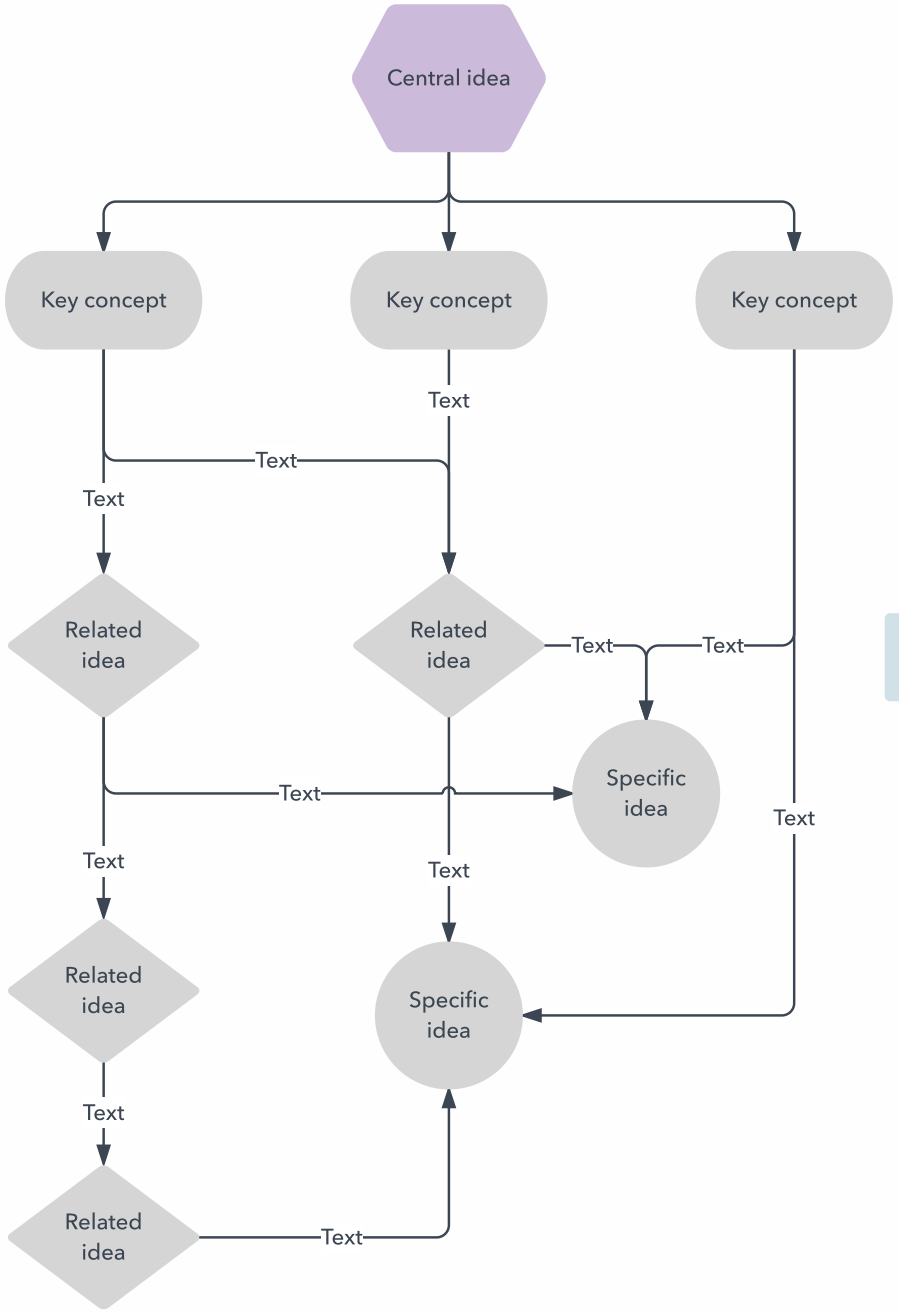
Guess the domain!

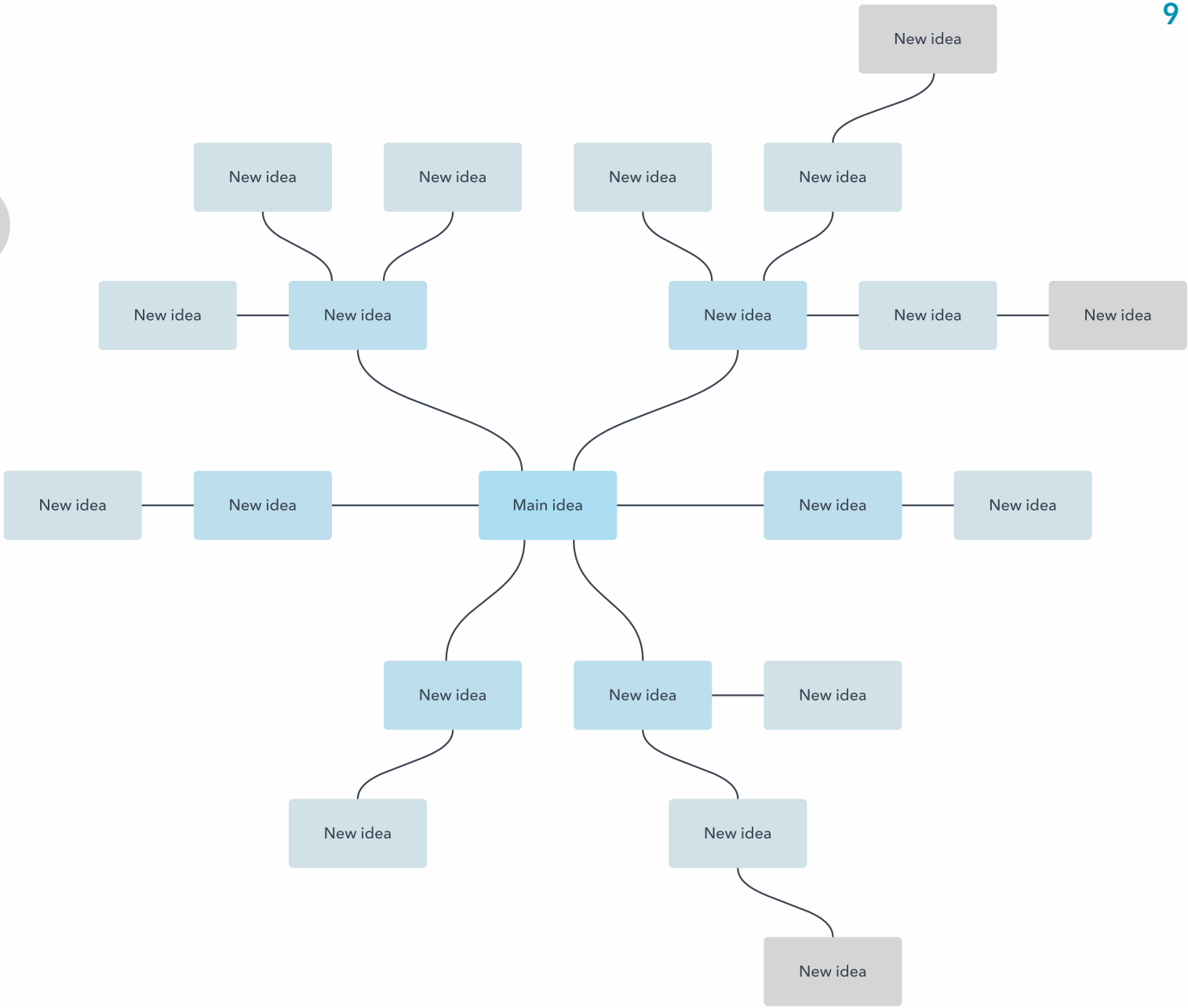
Typical design aspects

- Decomposition
  - components, patterns
  - data/control flow, interfaces
- Software quality
  - deployment, life cycle, upgrades, lock in
  - resource usage, energy efficiency, time behaviour
  - error handling, data integrity, recovery

Typical realisation aspects

- Waterfall ⟺ Iterative
- Manual ⟺ Automated
- Buy ⟺ Reuse
- Adapt ⟺ Remake
- Feature first ⟺ Market first
- Develop ⟺ Outsource
- Assign ⟺ Hire
- Test for release ⟺ Test for integration





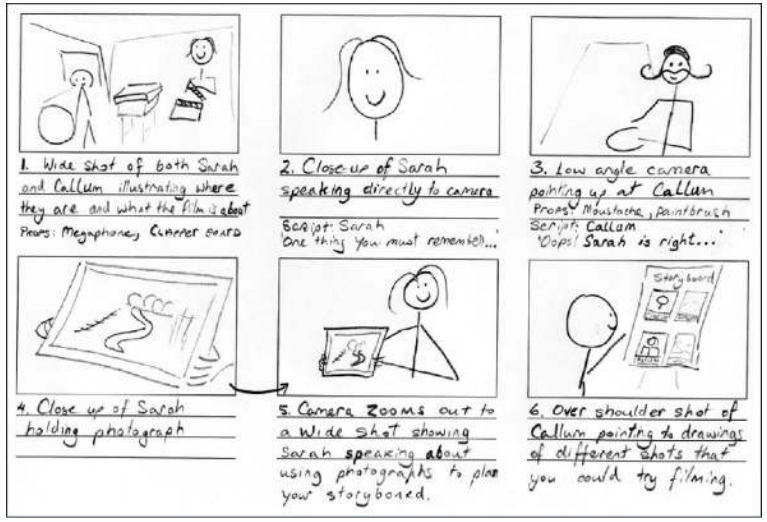
https://www.lucidchart.com/pages/concept-map

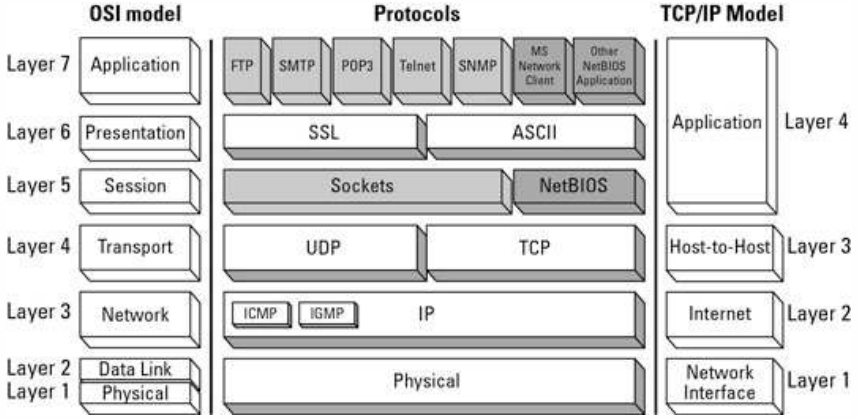
All models are wrong

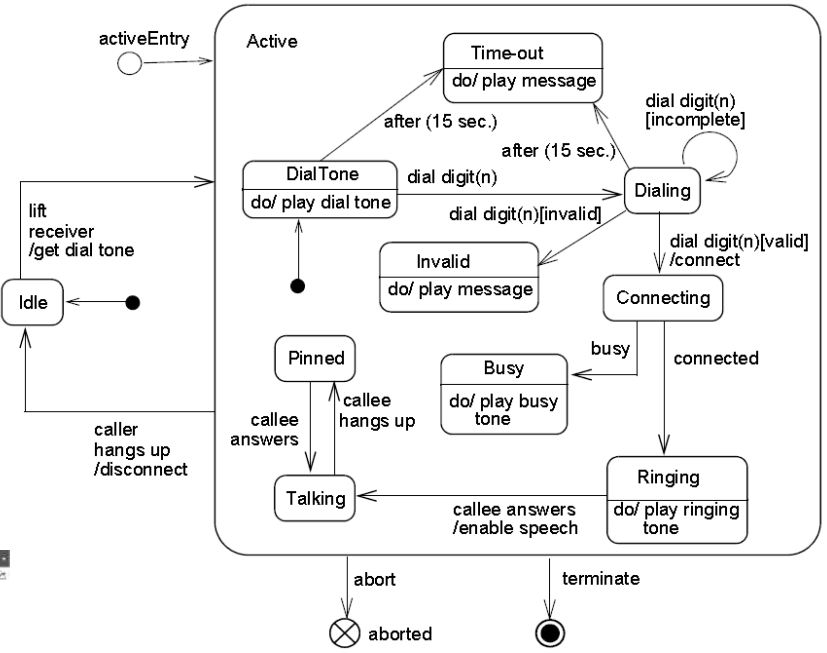
View

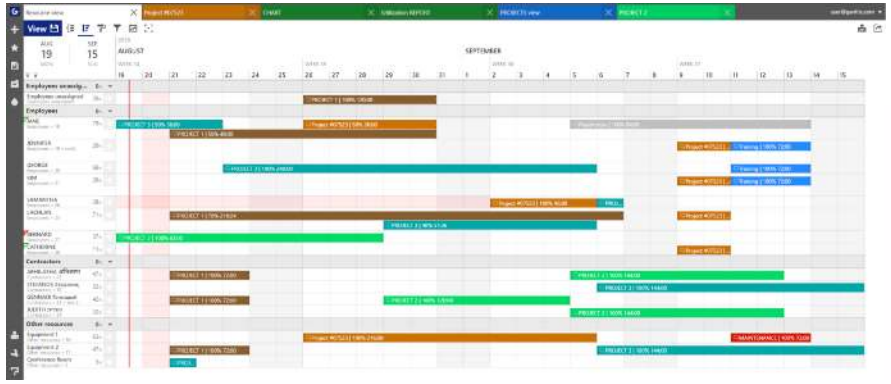
- Representation of
  - a system
  - its environment
  - their relation
- from the perspective of
  - a set of related concerns
- used by an architect
  - to define/explain/reason

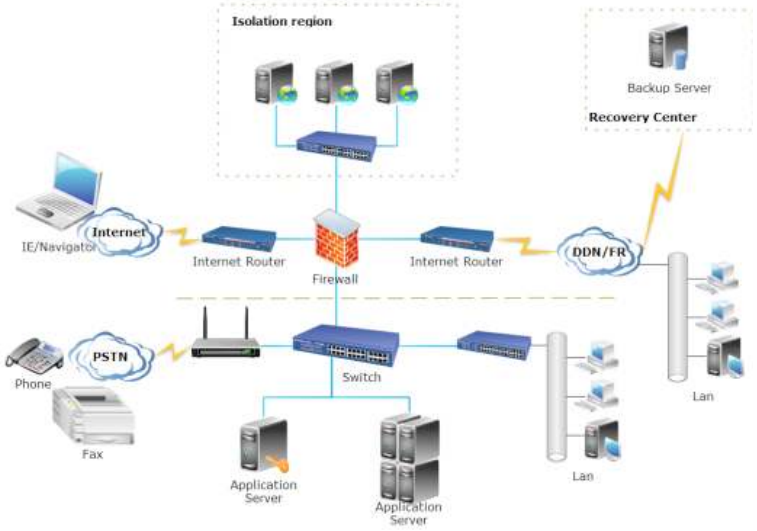
Views

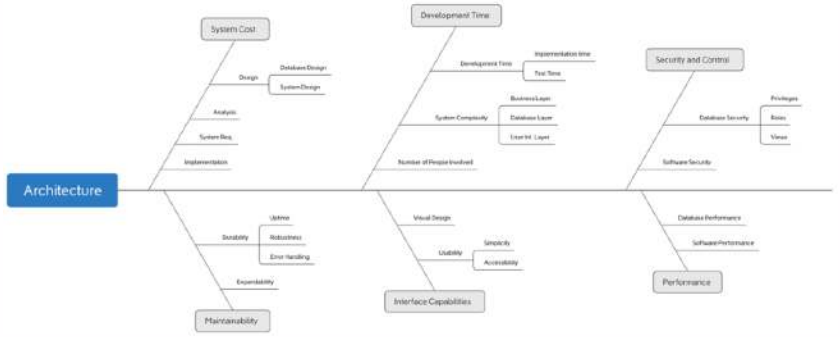


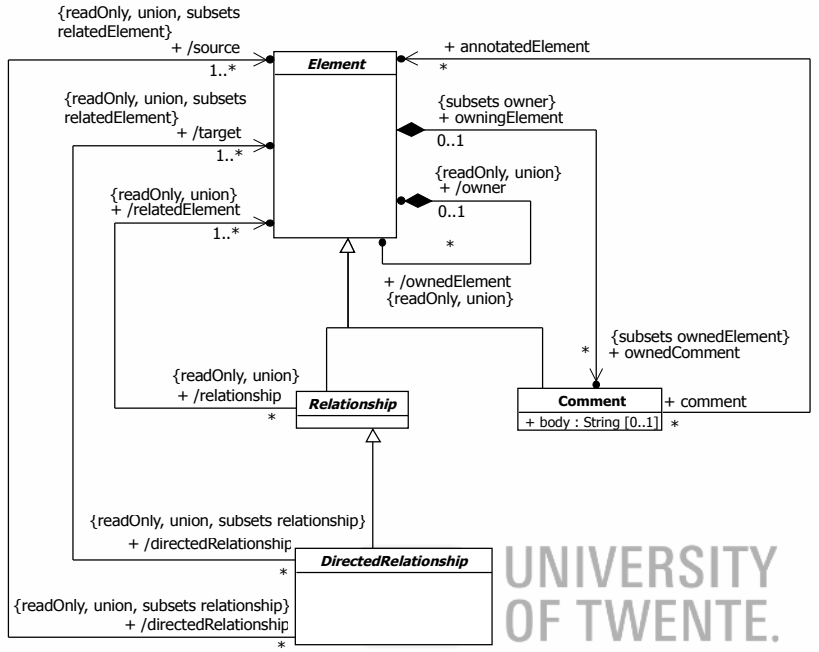












UNIVERSITY OF TWENTE.

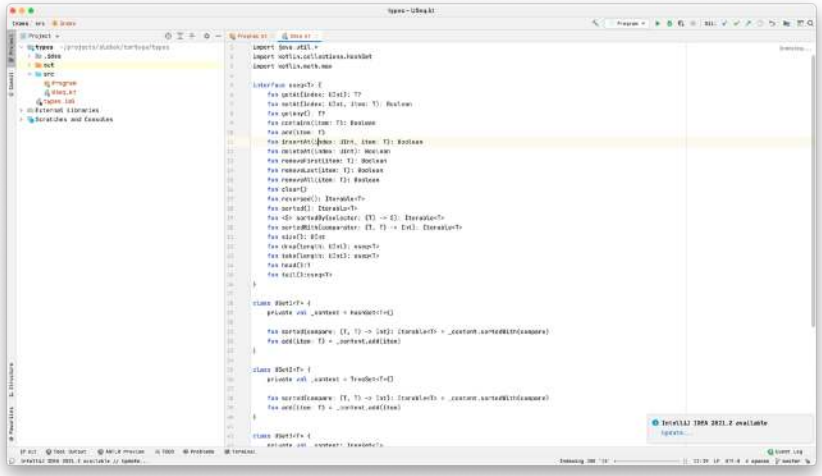
View == model? View ≠ model?

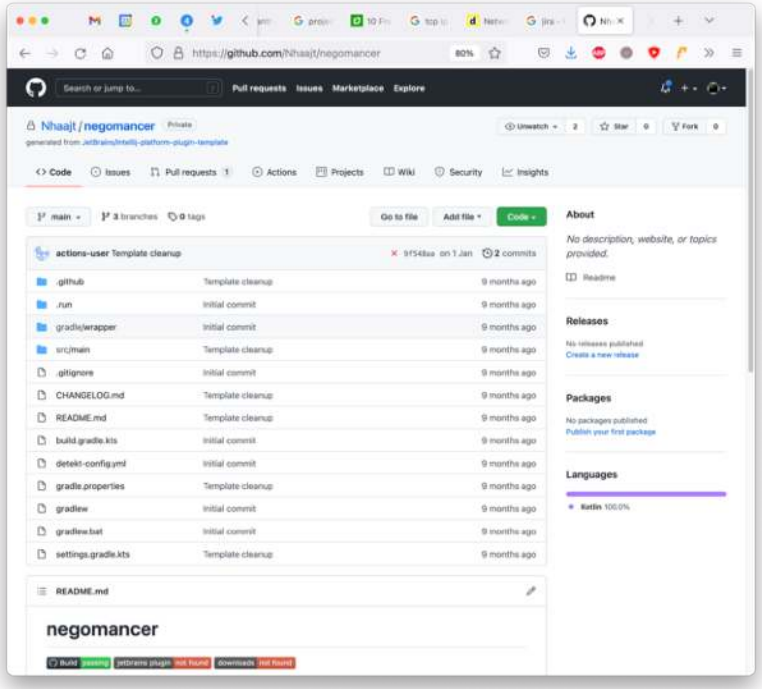
Model

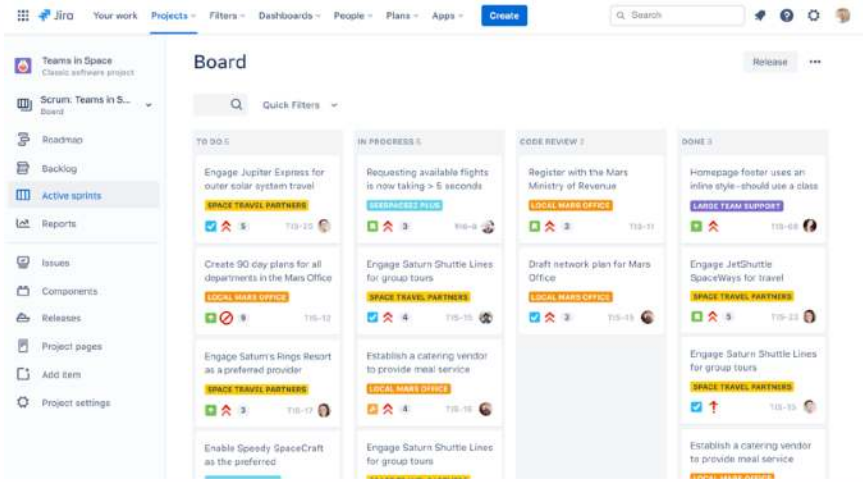
- *A model is*
  - *a simplified representation*
  - *of a part of the world*
  - *from a particular view*
- Architecture model
  - source specification
  - for one or more views

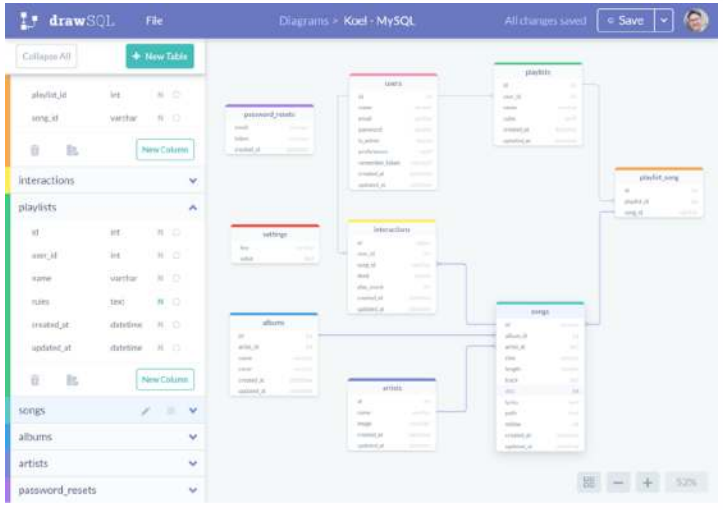
H. Stachowiak, *Allgemeine Modelltheorie*, 1973

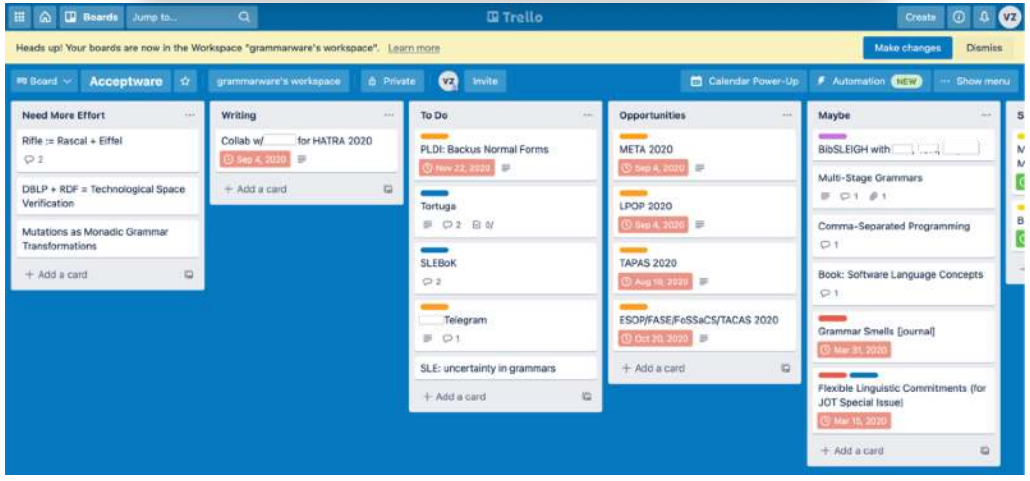
Models

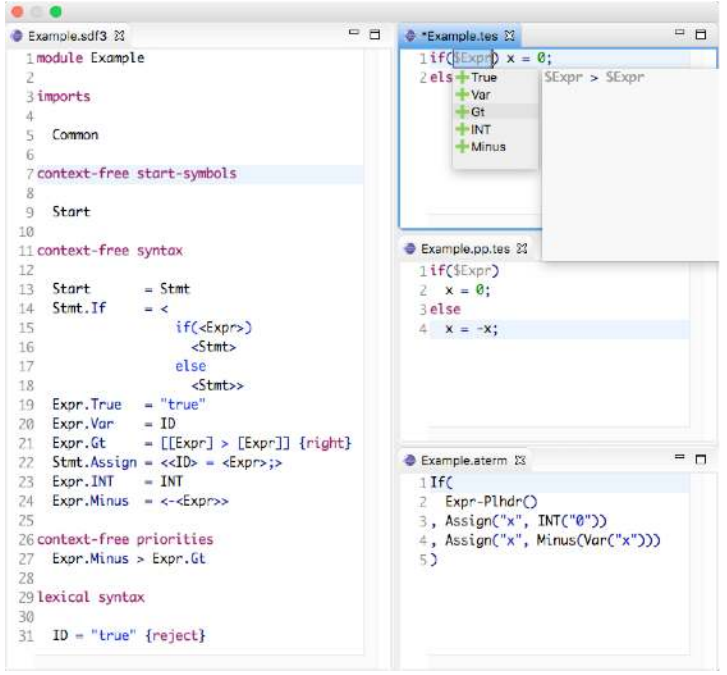



Nhaajt / negomancer
negomancer
README.md
About
Releases
Packages
Languages


Board
Teams in Space
Scrum: Teams in S...
Roadmap
Backlog
Active sprints
Reports
Issues
Components
Releases
Project pages
Add item
Project settings


drawSQL
Diagrams > Koel - MySQL
All changes saved
Save
New Table
interactions
playlists
songs
albums
artists
password_resets


Trello
Acceptware
Need More Effort
Writing
To Do
Opportunities
Maybe

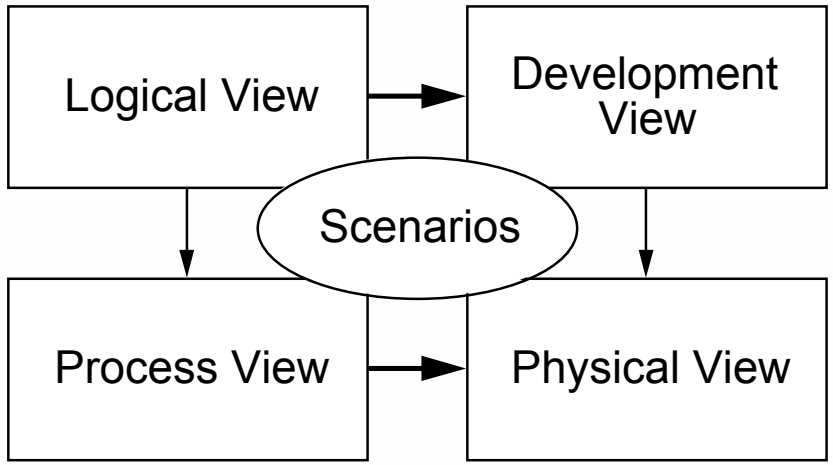

Example.sdf3
module Example
imports
Common
context-free start-symbols
Start
context-free syntax
Start = Stmt
Stmt.If = <
if(<Expr>)
<Stmt>
else
<Stmt>>
Expr.True = "true"
Expr.Var = ID
Expr.Gt = [[Expr] > [Expr]] {right}
Stmt.Assign = <<ID> = <Expr>;>
Expr.INT = INT
Expr.Minus = <-<Expr>>
context-free priorities
Expr.Minus > Expr.Gt
lexical syntax
ID = "true" {reject}
*Example.tes
Example.pp.tes
Example.aterm

Viewpoint

- Way of looking at
  - system
  - environment
- from the perspective of
  - stakeholder
  - concern
  - aspect

End-user
Functionality

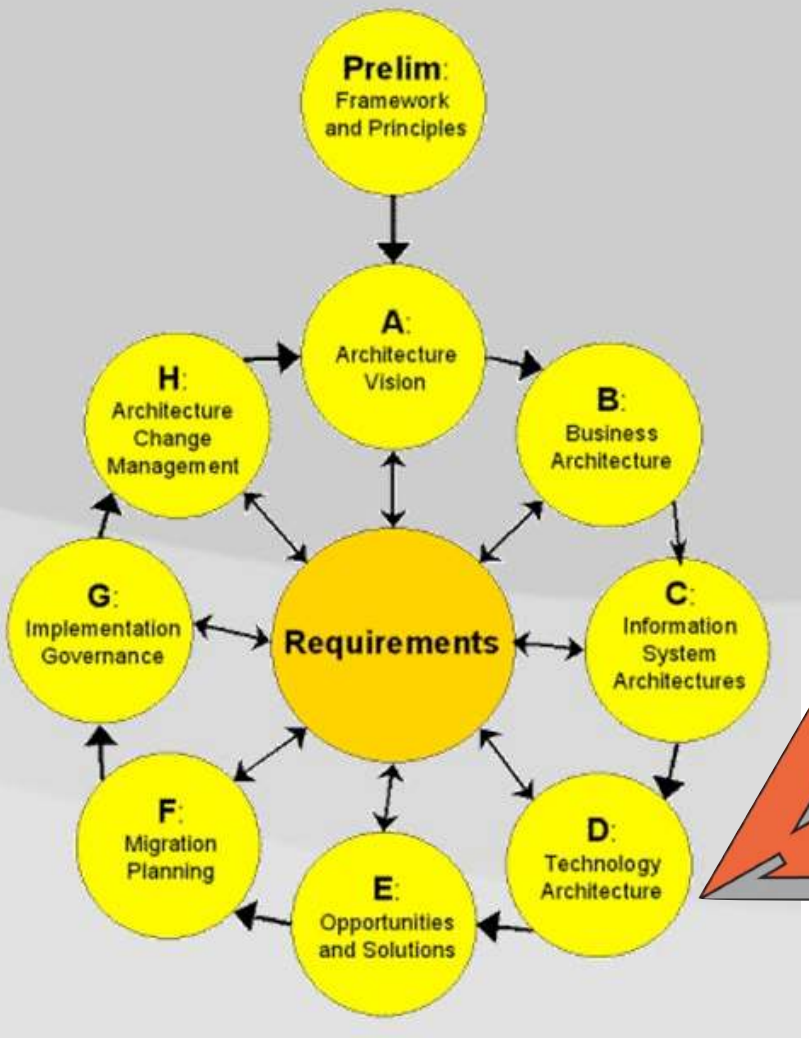
Programmers
Software management

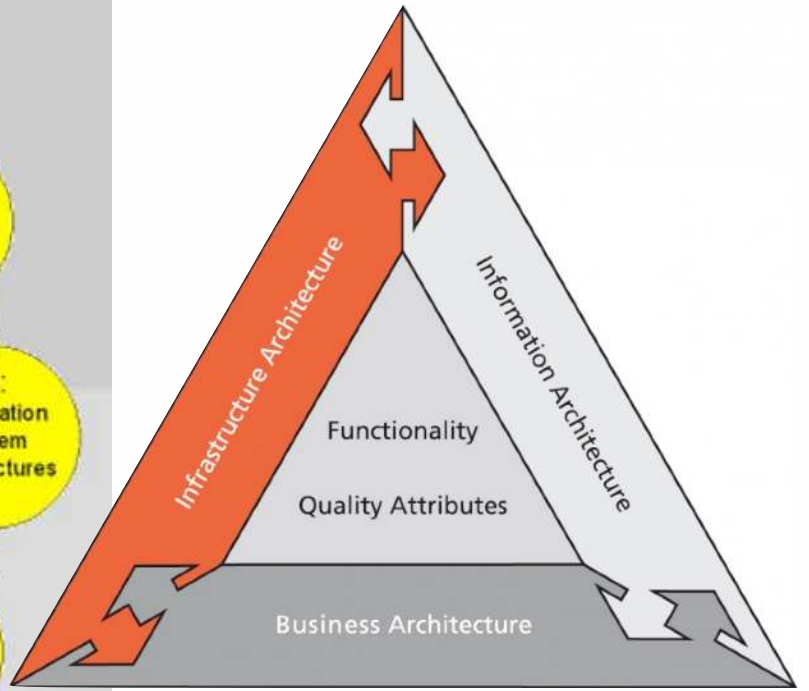


Integrators
Performance
Scalability

System engineers
Topology
Communications

	Why	How	What	Who	Where	When
Contextual	Goal List	Process List	Material List	Organisational Unit & Role List	Geographical Locations List	Event List
Conceptual	Goal Relationship	Process Model	Entity Relationship Model	Organisational Unit & Role Relationship Model	Locations Model	Event Model
Logical	Rules Diagram	Process Diagram	Data Model Diagram	Role Relationship Diagram	Locations Diagram	Event Diagram
Physical	Rules Specification	Process Function Specification	Data Entity Specification	Role Specification	Location Specification	Event Specification
Detailed	Rules Details	Process Details	Data Details	Role Details	Location Details	Event Details

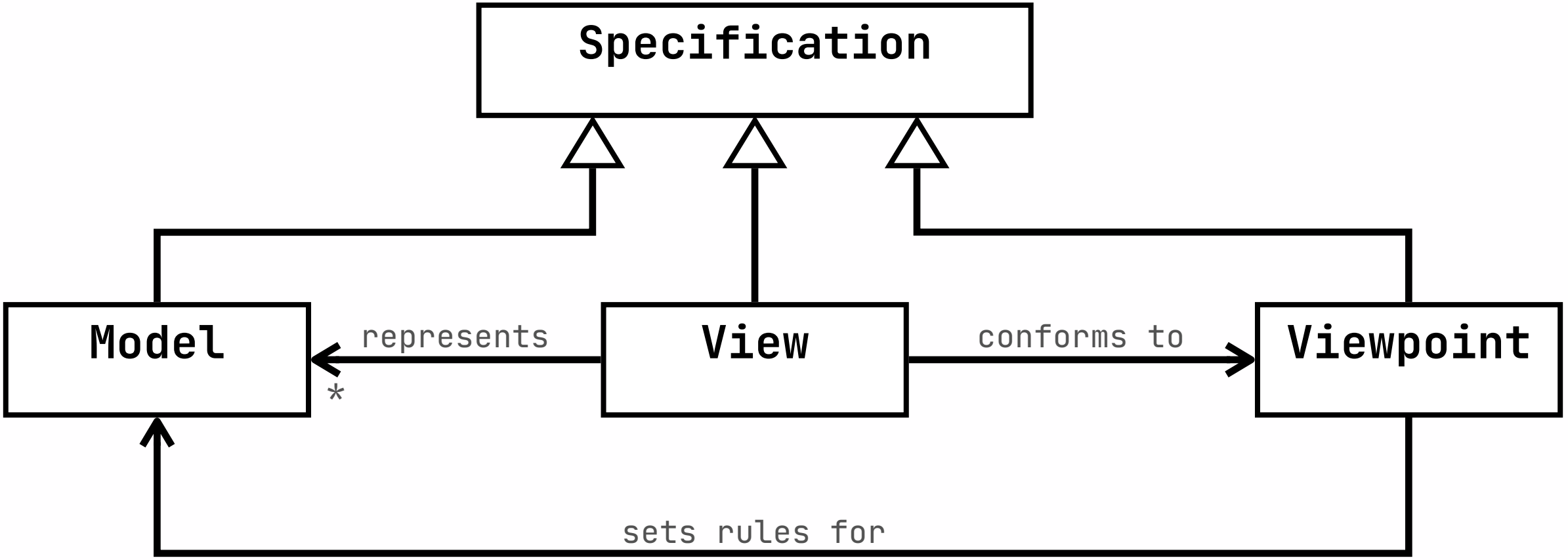




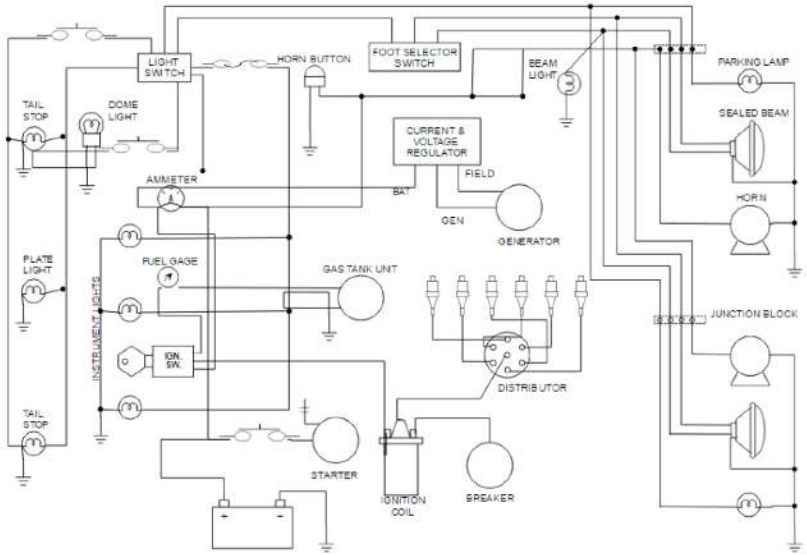
Viewpoints from a framework

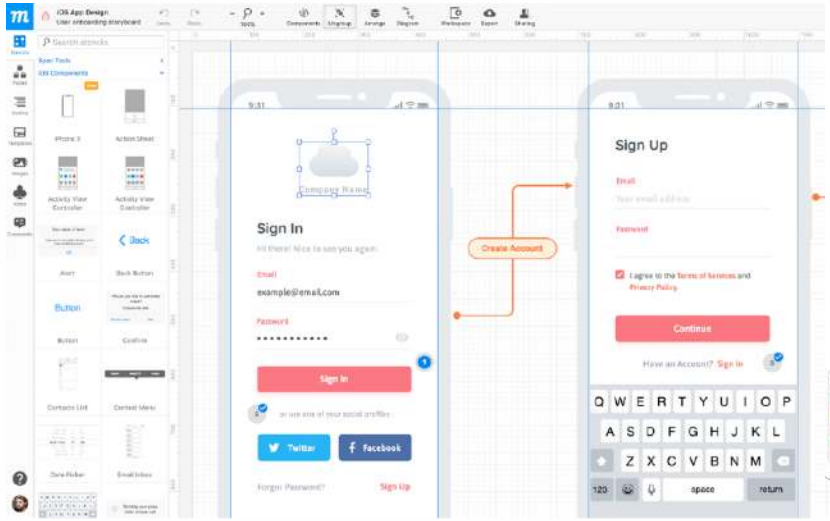
- 4+1 model…
- TOGAF…
- DYA…
- Zachman…
- Standard framework does not cover unique properties
  - Use specific viewpoints
- Views can be related via the architecture domain model.

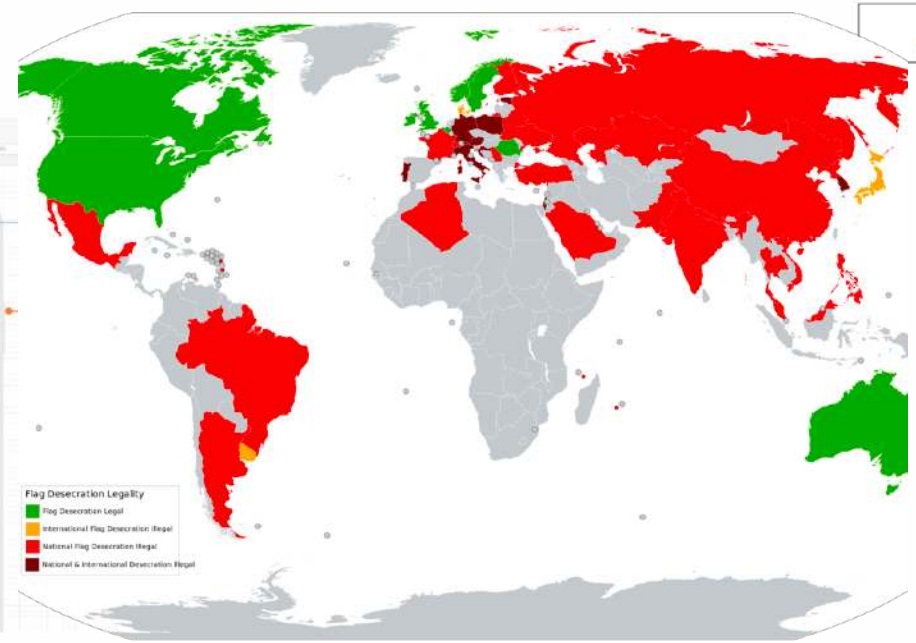
Model ← View → Viewpoint

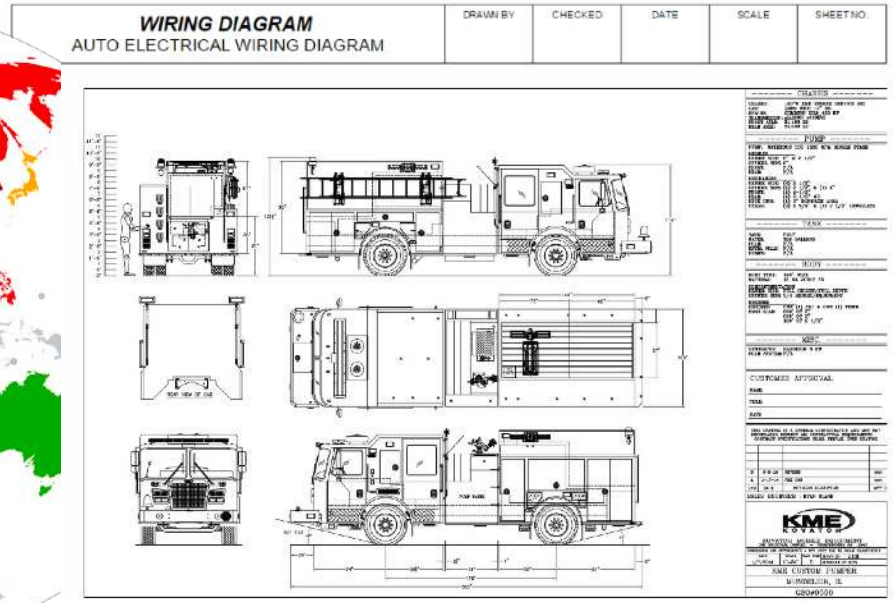


Model? View? Viewpoint?









Specify domains, identify aspects

- What are your domains?
- What are aspects relevant for your project?
- Think of:
  - Application
  - Design
  - Realisation
- Just list them and give a short definition

Select viewpoints, clean up views

- Which stakeholder(s)? Which concerns?
- Decide on the language:
  - Concepts and notation
  - Visualisation (depends on communication context)
- Specify how to translate the model(s) into a view.
- Pitfalls:
  - Too little views (e.g., just the decomposition)
  - Overloaded views (everything in one picture)

UNIVERSITY OF TWENTE.