Software Modelling

HUSZERL Gábor

huszerl@mit.bme.hu

MŰEGYETEM 1782

BME VILLAMOSMÉRNÖKI ÉS INFORMATIKAI KAR

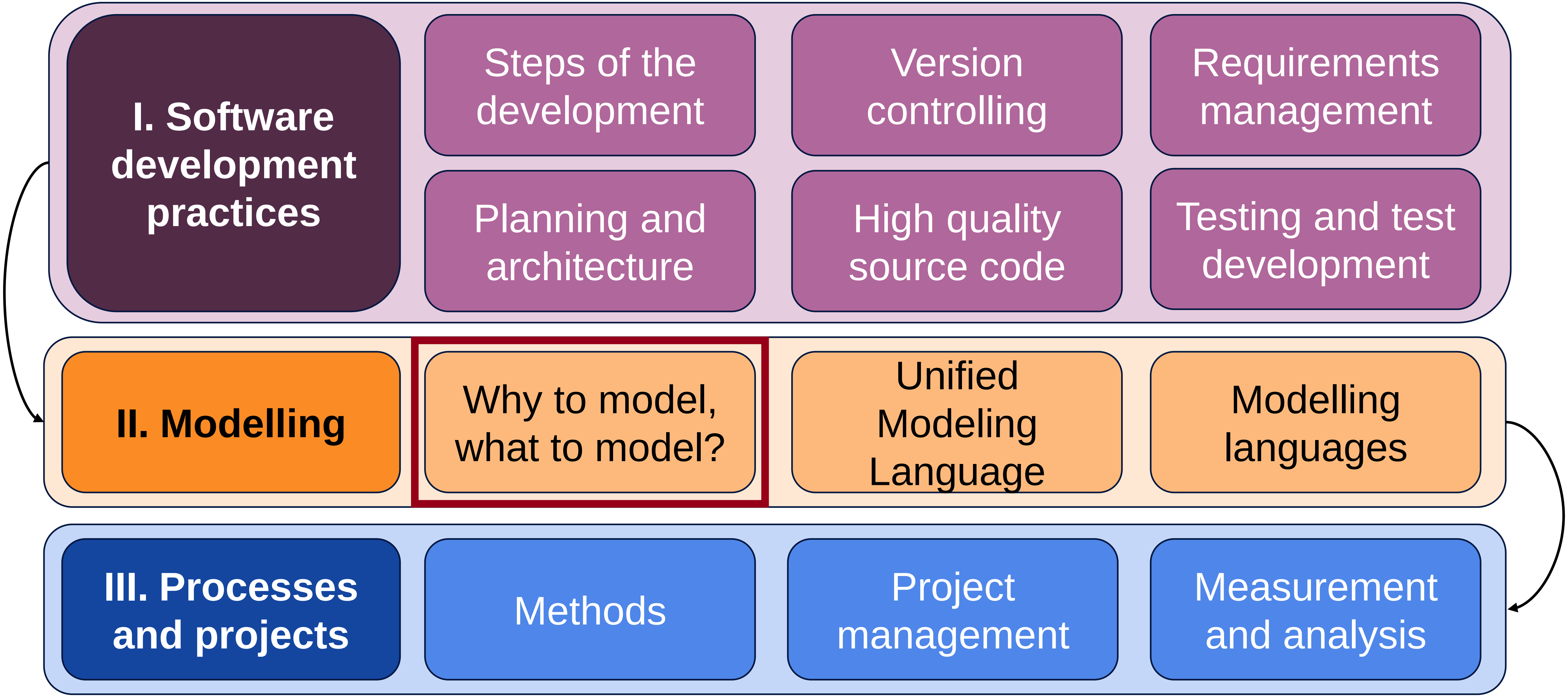
Méréstechnika és Információs Rendszerek Tanszék

ftsrg Critical Systems Research Group

Learning Outcomes

- At the end of the lecture the students are expected to be able to
- (K2) summarize the purpose and characteristics of the models,
- (K1) list modelling languages in computer engineering.

Further Topics of the Subject



I. Software development practices

- Steps of the development
- Version controlling
- Requirements management
- Planning and architecture
- High quality source code
- Testing and test development

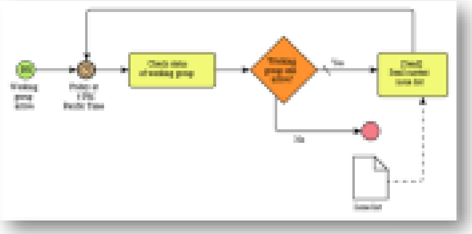
II. Modelling

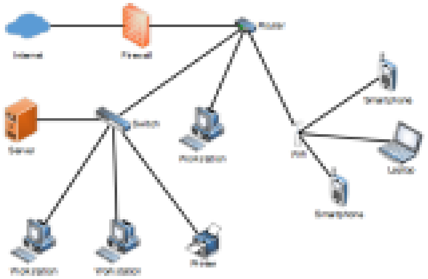
- Why to model, what to model?
- Unified Modeling Language
- Modelling languages

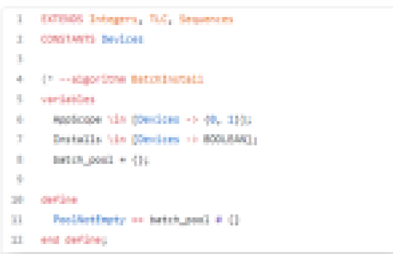
III. Processes and projects

- Methods
- Project management
- Measurement and analysis

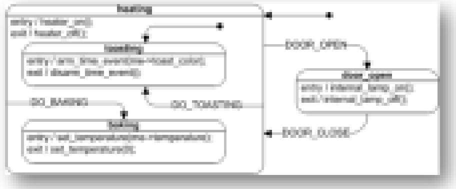
Modelling Basics







```
EXTENDS Integers, TLC, Sequences
CONSTANTS Devices

(* --algorithm BatchInstall
variables
  AppScope \in [Devices -> {0, 1}];
  Installs \in [Devices -> BOOLEAN];
  batch_pool = {};

define
  PoolNotEmpty == batch_pool # {}
end define;
```


heating
entry / heater_on();
exit / heater_off();
toasting
entry / arm_time_event(me->toast_color);
exit / disarm_time_event();
GO_BAKING
GO_TOASTING
baking
entry / set_temperature(me->temperature);
exit / set_temperature(0);
DOOR_OPEN
door_open
entry / internal_lamp_on();
exit / internal_lamp_off();
DOOR_CLOSE

Why? How? When?

What Modelling Languages Do You Know?

- Already known modelling languages:
  - Final state machines
  - Verilog
  - Graphs
  - Flow charts
  - E-R diagrams
  - Class diagrams
  - …

What Is a Model?

- Model is a partial and simplified depiction of the real or a hypothetical world (the „system”), of which can be substituted for specific viewpoints
- Decisions:
  - Which part of the world?
  - What is simplified?
  - How can it be mapped to the world?

When is it possible or recommended to use?

- Benefits
  - Smaller (finite)
  - Clearer

Scientific vs. Engineering Models

Scientific model
descriptive

- Made to understand the real world („the system”)
- Value: How well the model fits to the modelled system
- The Newtonian Laws do not apply inside a black hole

  → the model is wrong

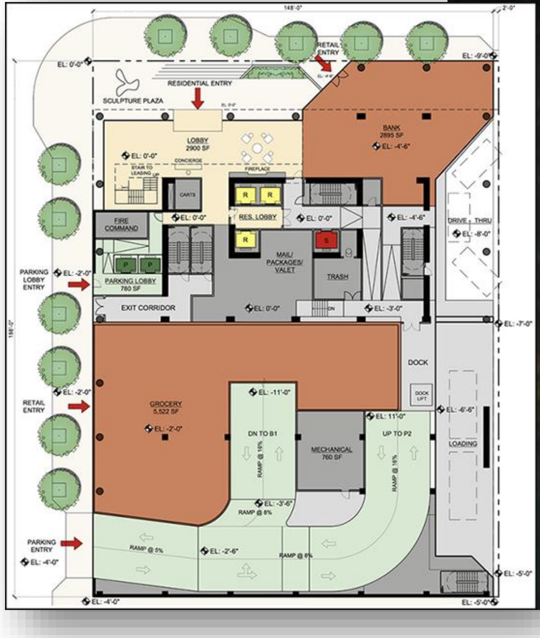
Engineering (design) model
prescriptive

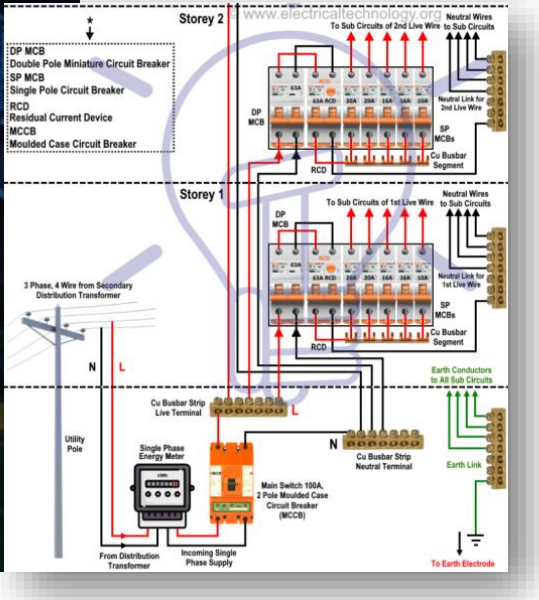
- Made to specify, design a system
- Value: How well the implemented system fits to the model
- The implemented PCB does not fit to the design (to the model)

  → the „system" is wrong

E. A. Lee. Modeling in engineering and science. DOI

Creating and Using Engineering Models



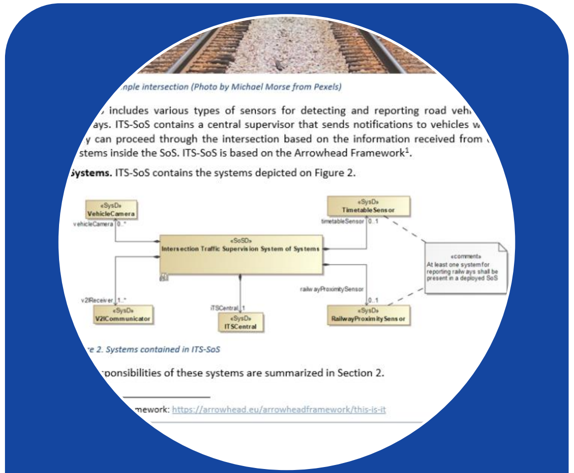

www.electricaltechnology.org
Storey 2
DP MCB
Double Pole Miniature Circuit Breaker
SP MCB
Single Pole Circuit Breaker
RCD
Residual Current Device
MCCB
Moulded Case Circuit Breaker
To Sub Circuits of 2nd Live Wire
Neutral Wires to Sub Circuits
Neutral Link for 2nd Live Wire
SP MCBs
Cu Busbar Segment
Storey 1
To Sub Circuits of 1st Live Wire
Neutral Link for 1st Live Wire
3 Phase, 4 Wire from Secondary Distribution Transformer
Earth Conductors to All Sub Circuits
Cu Busbar Strip Live Terminal
Utility Pole
Single Phase Energy Meter
Main Switch 100A, 2 Pole Moulded Case Circuit Breaker (MCCB)
Cu Busbar Strip Neutral Terminal
Earth Link
From Distribution Transformer
Incoming Single Phase Supply
To Earth Electrode
N
L

What Is the Point of Modelling?

- I am developing software. Should I do modelling?
  - Yes! In fact, you already do!
    - (The source code is a kind of model…)
    - Most importantly: **mental models**
- When should be these models documented?
  - The role of the model: **communication**
    - Human → Human
    - Human → Machine
    - Machine → Machine
    - Human → Human (themselves in the future)
      - E.g. Why we have designed/implemented this algorithm this way some years ago?

Styles of Modelling

Brainstorming



…nple intersection (Photo by Michael Morse from Pexels)

… includes various types of sensors for detecting and reporting road veh… …ays. ITS-SoS contains a central supervisor that sends notifications to vehicles w… …y can proceed through the intersection based on the information received from … …stems inside the SoS. ITS-SoS is based on the Arrowhead Framework[1].

…ystems. ITS-SoS contains the systems depicted on Figure 2.

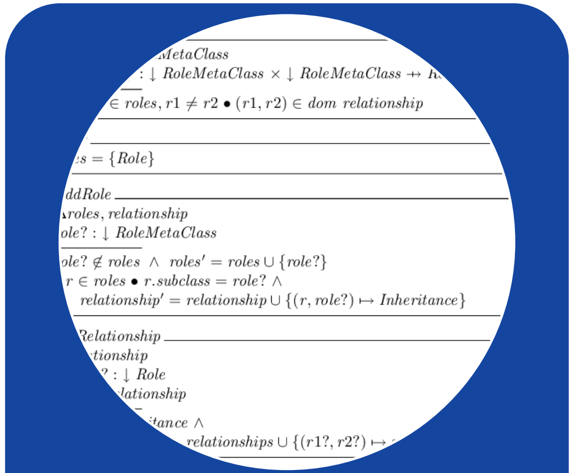
«SysD» VehicleCamera
vehicleCamera 0..*
«SysD» TimetableSensor
timetableSensor 0..1
«SoSD» Intersection Traffic Supervision System of Systems
«comment» At least one system for reporting railways shall be present in a deployed SoS
railwayProximitySensor
0..1
v2IReceiver 1..*
«SysD» V2ICommunicator
ITSCentral 1
«SysD» ITSCentral
«SysD» RailwayProximitySensor

…re 2. Systems contained in ITS-SoS

…ponsibilities of these systems are summarized in Section 2.

…mework: https://arrowhead.eu/arrowheadframework/this-is-it

Detailed specification



$\ldots MetaClass$

$\ldots : \downarrow RoleMetaClass \times \downarrow RoleMetaClass \nrightarrow R\ldots$

$\ldots \in roles, r1 \neq r2 \bullet (r1, r2) \in dom\ relationship$

$\ldots s = \{Role\}$

$\ldots ddRole$

$\ldots roles, relationship$

$\ldots ole? : \downarrow RoleMetaClass$

$\ldots ole? \notin roles \ \wedge \ roles' = roles \cup \{role?\}$

$\ldots r \in roles \bullet r.subclass = role? \ \wedge$

$relationship' = relationship \cup \{(r, role?) \mapsto Inheritance\}$

$\ldots Relationship$

$\ldots tionship$

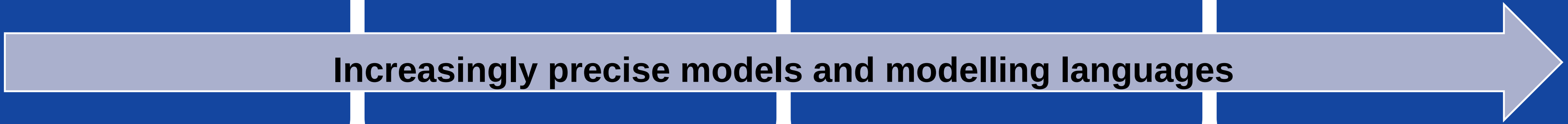
$\ldots ? : \downarrow Role$

$\ldots lationship$

$\ldots tance \ \wedge$

$\ldots relationships \cup \{(r1?, r2?) \mapsto \ldots$

Formal model



Increasingly precise models and modelling languages

Basic Concepts: Refinement and Abstraction

- *Refinement*: adding further details to the model …

  e.g. breaking down the system into subsystems

  … while keeping the input model an abstraction of it

  e.g. the subsystems together meet the same requirements

- Its inverse: *abstraction*

  e.g. merging subsystems

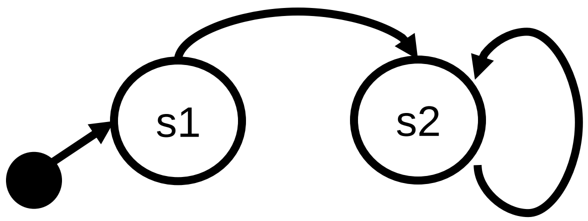
- Different kinds of refinement, e.g.
  - Hierarchical refinement: „ unpacking the boxes”
  - Set refinement: partitioning of a set (variable domain refinement)
    - **good** / **bad** → **fast** / **average** / **slow** / **incomplete** / **dangerous**

Modelling Language

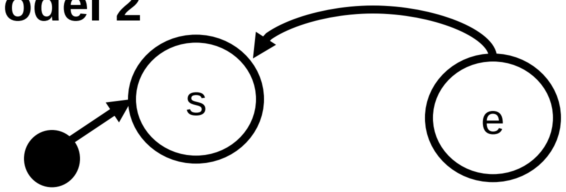
- Defines what are the well-formed models

- Required to define a language:
  - **Syntax**: what rules can be used to build a valid model?
  - **Semantics**: what does the model mean? What domain can we map it to?

Model 1



Model 2



Modelling language

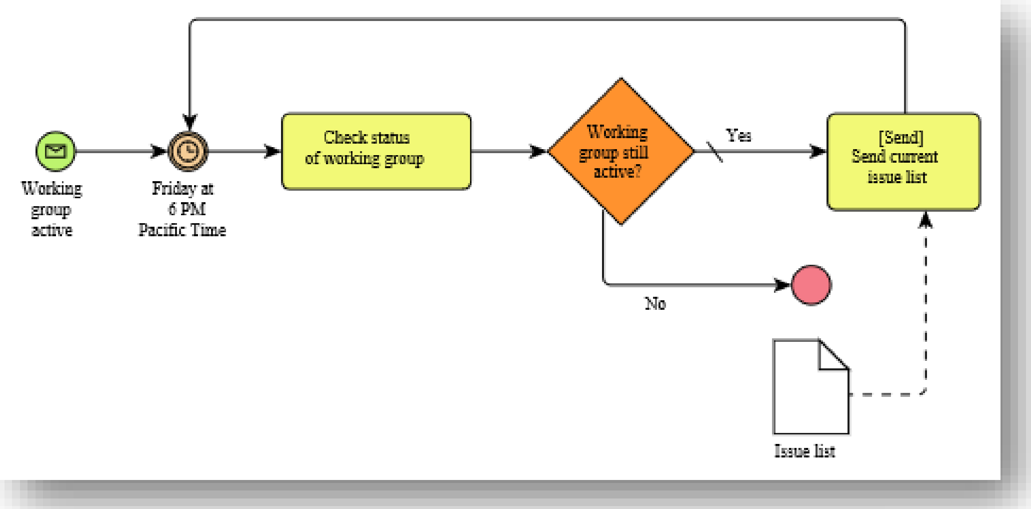
Syntax

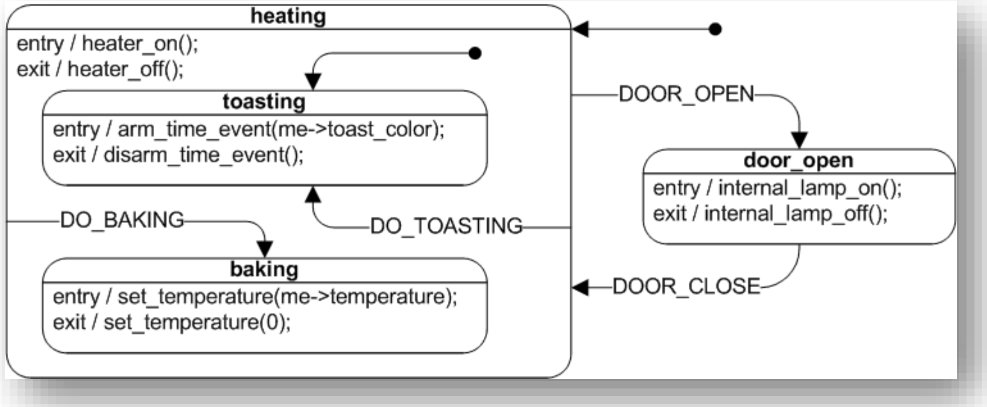
- Transition (arrow) connects source and target states (circle).
- The source and target states can be the same.

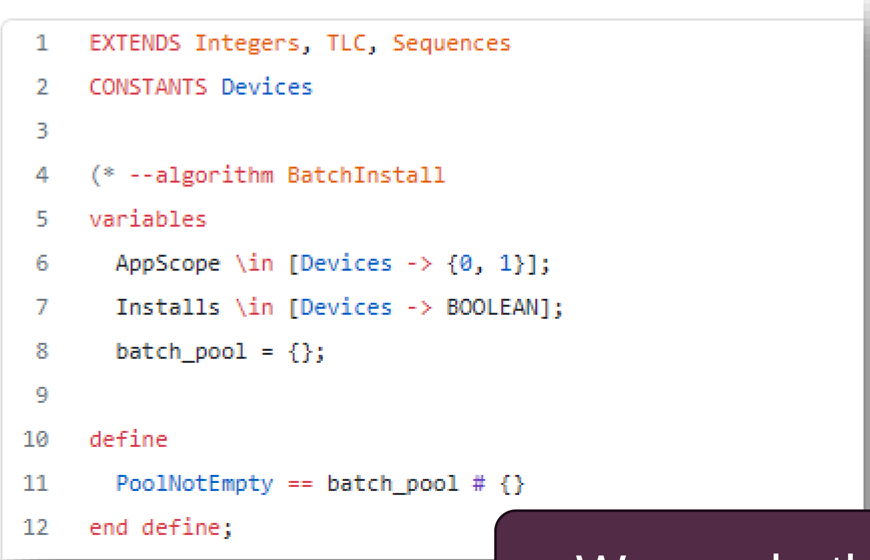
Semantics

- Transitions represent state changes in response to events
- Model's meaning is a sequence of state changes

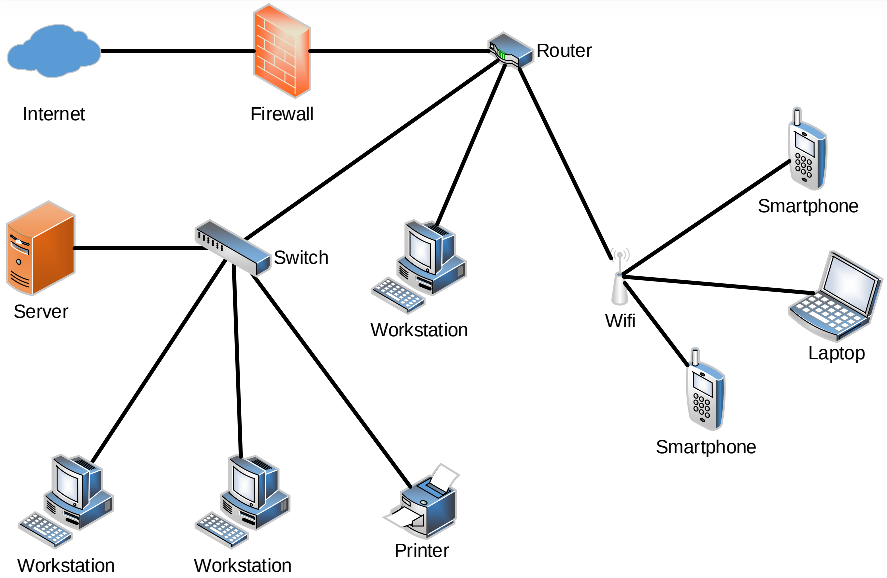
Typical Models and Languages in Comp.Eng.


Working group active
Friday at 6 PM Pacific Time
Check status of working group
Working group still active?
Yes
No
[Send] Send current issue list
Issue list


heating
entry / heater_on();
exit / heater_off();
toasting
entry / arm_time_event(me->toast_color);
exit / disarm_time_event();
DO_BAKING
DO_TOASTING
baking
entry / set_temperature(me->temperature);
exit / set_temperature(0);
DOOR_OPEN
door_open
entry / internal_lamp_on();
exit / internal_lamp_off();
DOOR_CLOSE



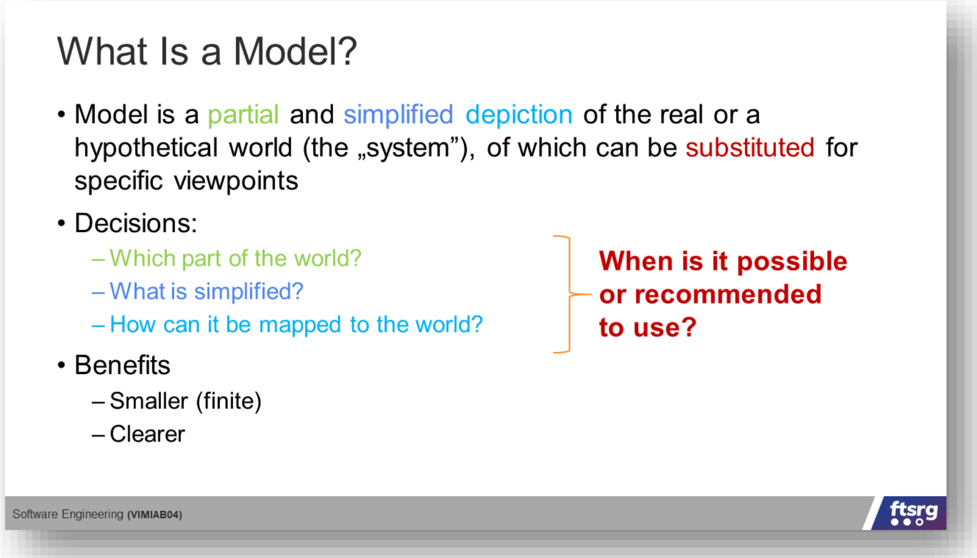
```
EXTENDS Integers, TLC, Sequences
CONSTANTS Devices

(* --algorithm BatchInstall
variables
  AppScope \in [Devices -> {0, 1}];
  Installs \in [Devices -> BOOLEAN];
  batch_pool = {};

define
  PoolNotEmpty == batch_pool # {}
end define;
```



We use both textual and graphical languages.

Summary

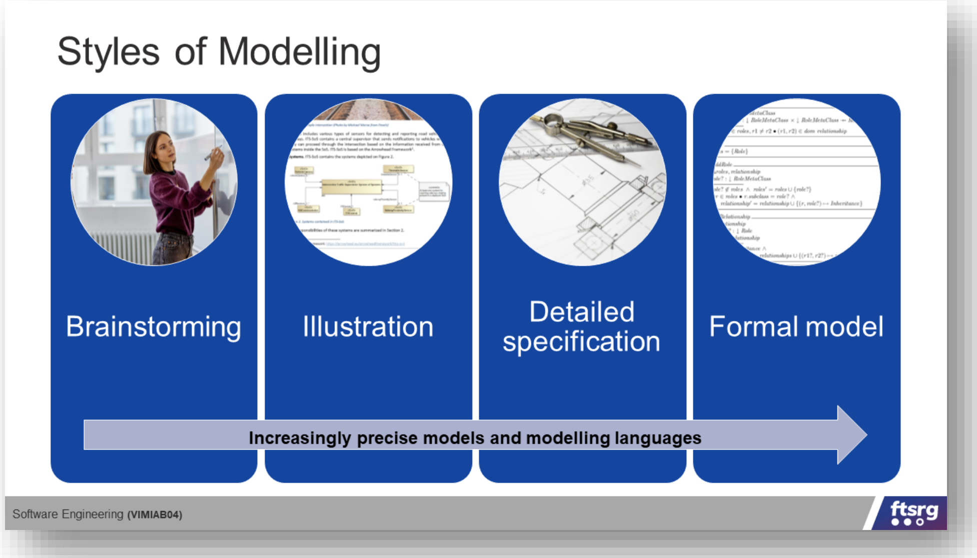
Summary



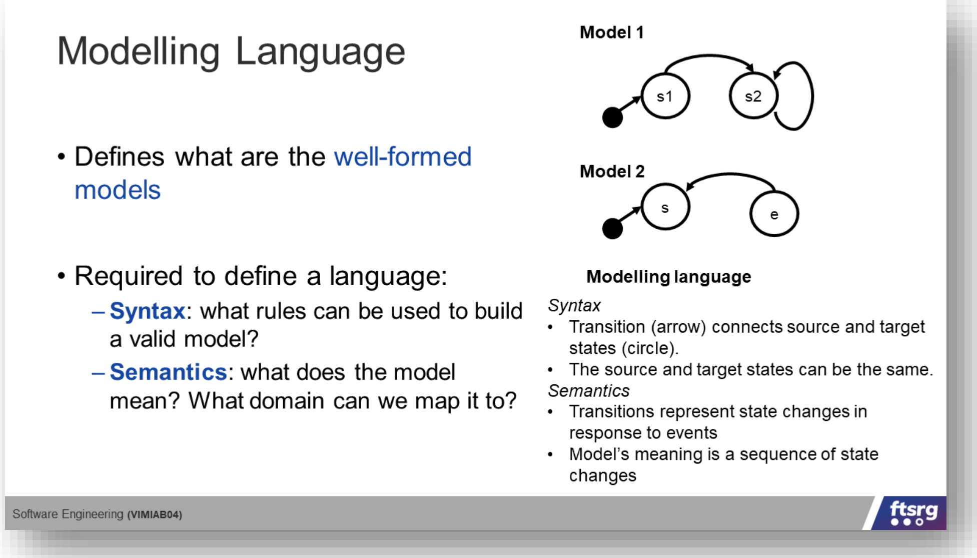
What Is a Model?

- Model is a partial and simplified depiction of the real or a hypothetical world (the „system”), of which can be substituted for specific viewpoints
- Decisions:
  - Which part of the world?
  - What is simplified?
  - How can it be mapped to the world?
- Benefits
  - Smaller (finite)
  - Clearer

When is it possible or recommended to use?



Styles of Modelling



Modelling Language

- Defines what are the well-formed models
- Required to define a language:
  - **Syntax**: what rules can be used to build a valid model?
  - **Semantics**: what does the model mean? What domain can we map it to?

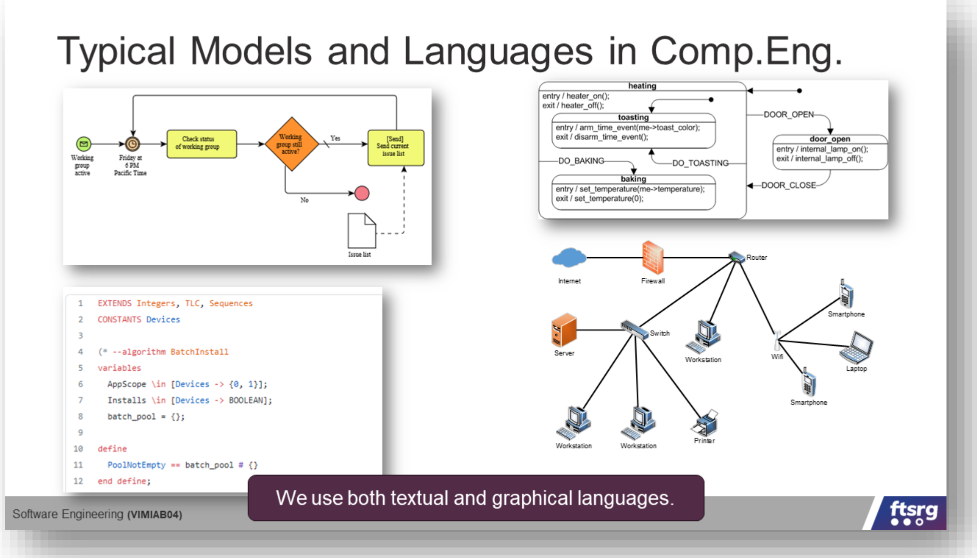
Modelling language

Syntax

- Transition (arrow) connects source and target states (circle).
- The source and target states can be the same.

Semantics

- Transitions represent state changes in response to events
- Model's meaning is a sequence of state changes



Typical Models and Languages in Comp.Eng.

Working group active
Friday at 6 PM Pacific Time
Check status of working group
Working group still active?
Yes
No
[Send] Send current issue list
Issue list
heating
entry / heater_on();
exit / heater_off();
toasting
entry / arm_time_event(me->toast_color);
exit / disarm_time_event();
DO_BAKING
DO_TOASTING
baking
entry / set_temperature(me->temperature);
exit / set_temperature(0);
DOOR_OPEN
door_open
entry / internal_lamp_on();
exit / internal_lamp_off();
DOOR_CLOSE
Internet
Firewall
Router
Server
Switch
Workstation
Wifi
Smartphone
Laptop
Smartphone
Workstation
Workstation
Printer

```
EXTENDS Integers, TLC, Sequences
CONSTANTS Devices

(* --algorithm BatchInstall
variables
  AppScope \in [Devices -> {0, 1}];
  Installs \in [Devices -> BOOLEAN];
  batch_pool = {};

define
  PoolNotEmpty == batch_pool # {}
end define;
```

We use both textual and graphical languages.