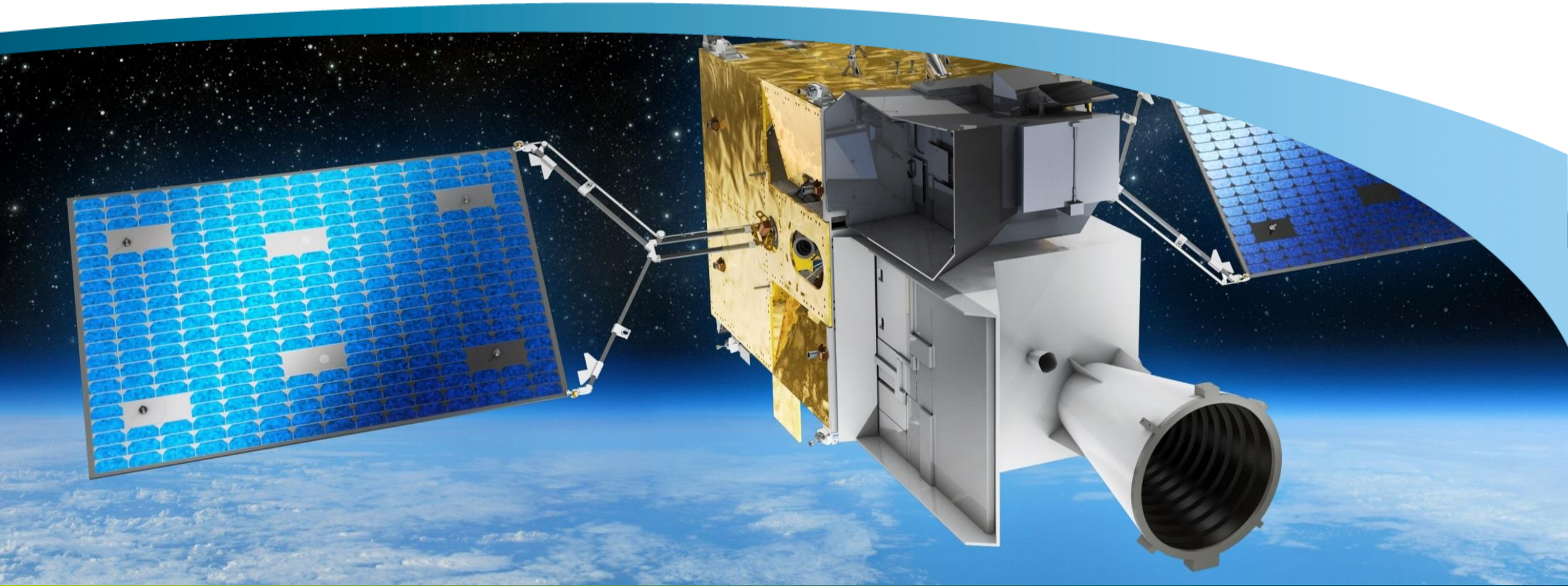
OHB System AG
Andreas Wortmann
18.10.2016, ADCSS



SPACE SYSTEMS

Rationale, recent projects and future options for Model based Developments

We. Create. Space.

Agenda

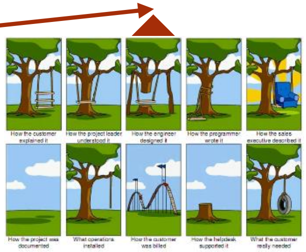
- Why (not) Model based Engineering/Development
- Application of Model Based Approaches – Recent Projects
- Studies and Long–Term Options

Why Model based Engineering

- Very brief development life cycle

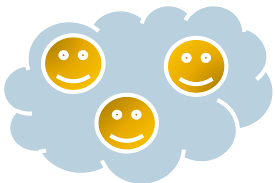
Reality

Intention



Gap:
The wrong system has been built

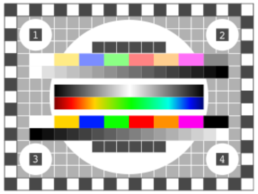
Goal:
Minimize gap



Discussion
(reveal, listen, understand …)

Natural Language Requirements

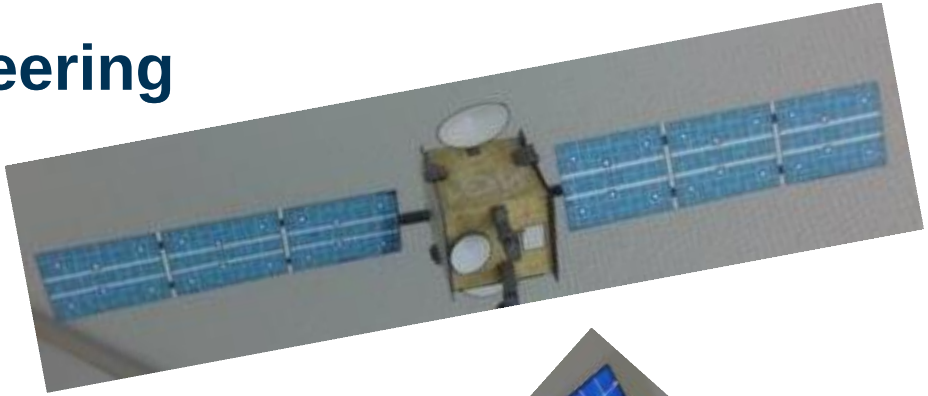
Implementation



Test,
& Verification

What we need here is:
An efficient objective alignment in the various steps of the life-cycle to support communication.

What is Model based Engineering



- This objective language is modelling
  (→ Model centric engineering)

  - Ensure a common/understanding
  - A model is a virtual representation of something
    - Interdisciplinary
    - Usually sufficiently complex
    - Contains interdependent parts

  - Example (entry) languages to : UML, SysML …

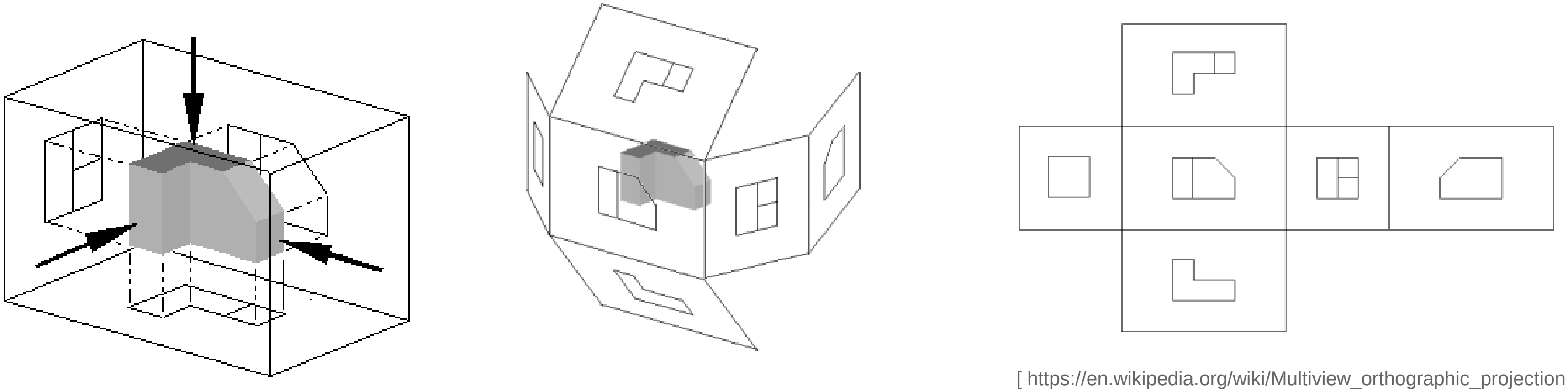
  - Different views on the same model supported
    - A view has a limited scope and purpose
    - Allows different specialists to work in their domain
    - Gather information to ease analyses and documentation

What is Model based Engineering

- Model

A **conceptual model** is a representation of a system, made of the composition of concepts which are used to help people know, understand, or simulate a subject the model represents. [Wikipedia]

- Different views of the same model



[https://en.wikipedia.org/wiki/Multiview_orthographic_projection]

So … it‘s more than just drawing UML figures

Benefits of Model based Engineering

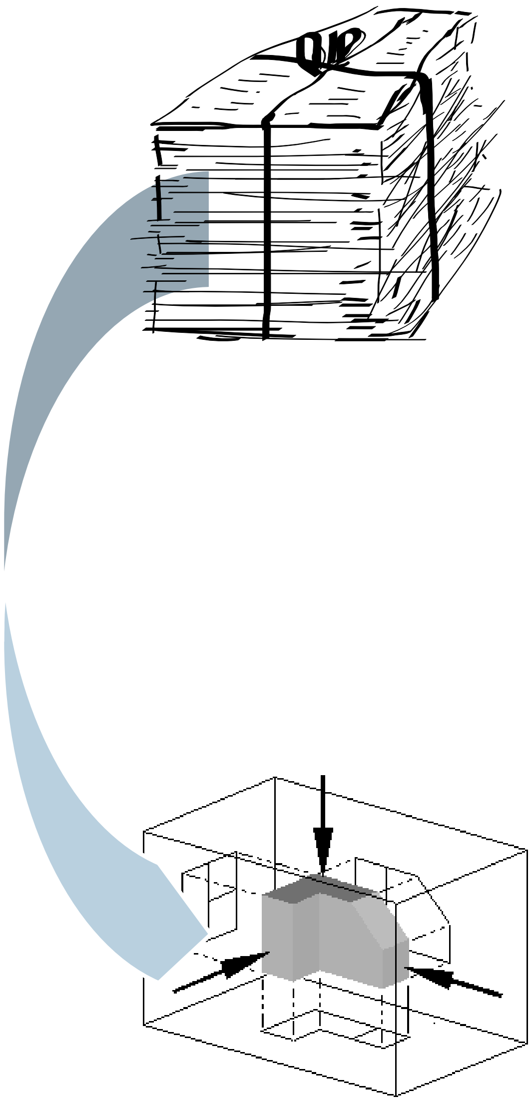
- The model is objective and has well-defined semantics
  - The model is machine-readable
  - Benefits:

    → Consistent representation of the (desired) system

    → (Parts of) the implementation can be generated

    → Checks and Analyses can be carried out on the model

    → …

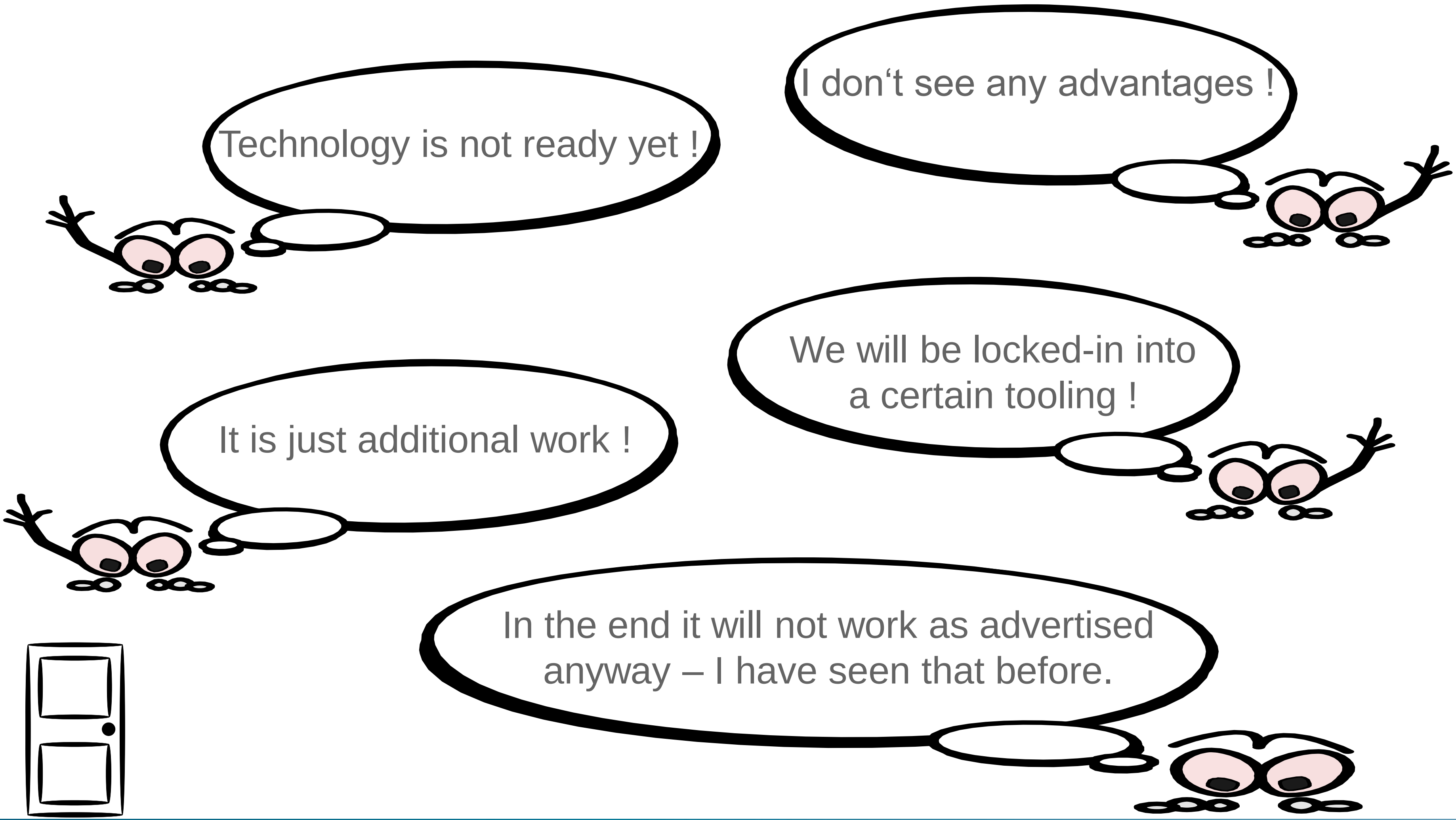


- Document based Engineering → Model based Engineering
- Diagram-centric → Model-centric "thinking"

… Where is, will and might it be useful …

- Mechanical, Thermal, Structure (MTS) Design
  - CAD tools → fully applied, virtually self-evident
  - Analyses → corrective actions → model validation
- AOCS (Algorithms are exercised on a modeled environment)
- Word Processor (Office suites …)
- …
- System Engineering (SysML …)
- Requirements Engineering
- Software
- FDIR
- Operational and Test Procedures
- Different versions for analysis/evaluation or product line variability
- …

… Real-Life Hesitance / Counter-Arguments



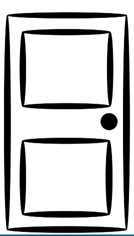
Motivations to apply model based approaches

New technology/tooling has evolved and we want to investigate it – it's model based

There is a issue which is worth analysis – define a mathematical model

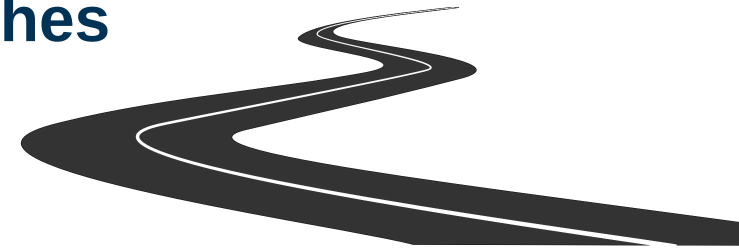
"… model based …"

In my project I have a specific challenge to master – the solution we came up with involves a model



We have learned from previous projects and want to significantly improve – so we need a solid foundation

Application of Model Based Approaches



- AOCS Algorithms and Environmental Models
- Mechanical, Thermal, Structure, Harness …
- Concurrent Engineering Facility

Avionics:

- Model Based Software Development using IBM Rhapsody in C
- Electronic Datasheets (EDS) as part of the Satellite Reference Database (SRDB)
- Valispace (Collaborative System Engineering Tool)
- Hardware/Software Codesign
- Schedulability Analysis of Flight Software

Observation:

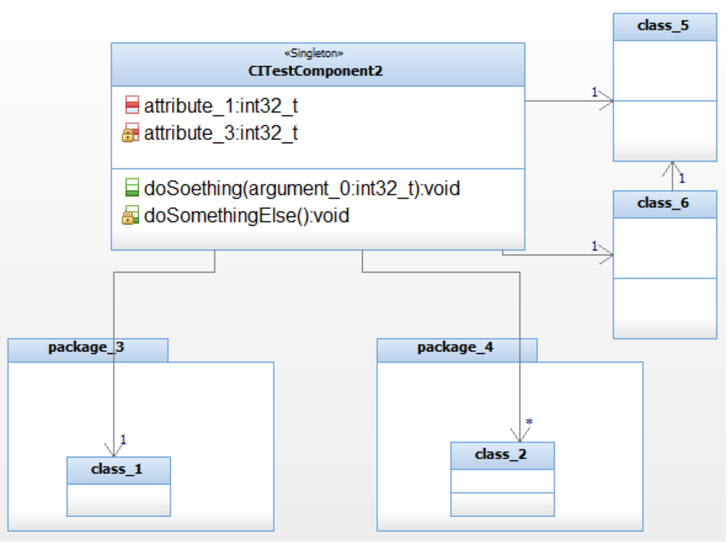
Mainly the supporters and motivators are with software background.

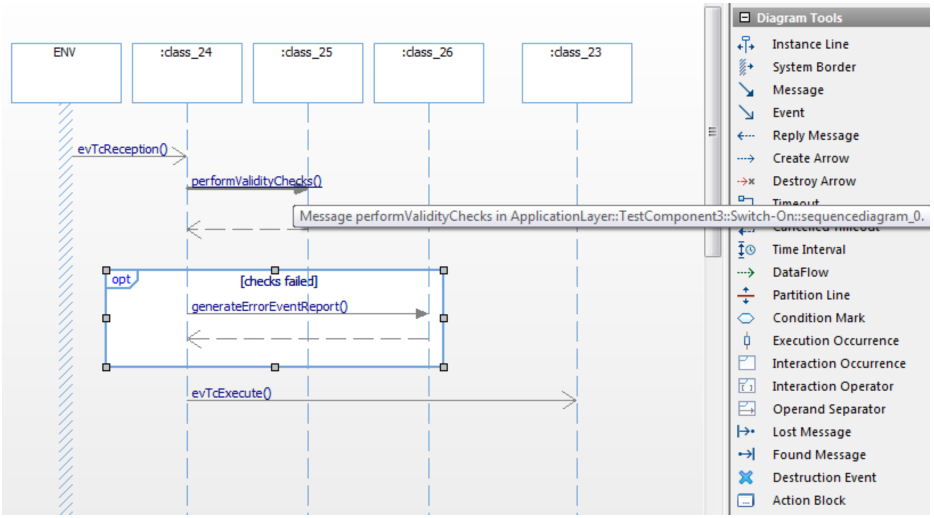
IBM Rhapsody in C

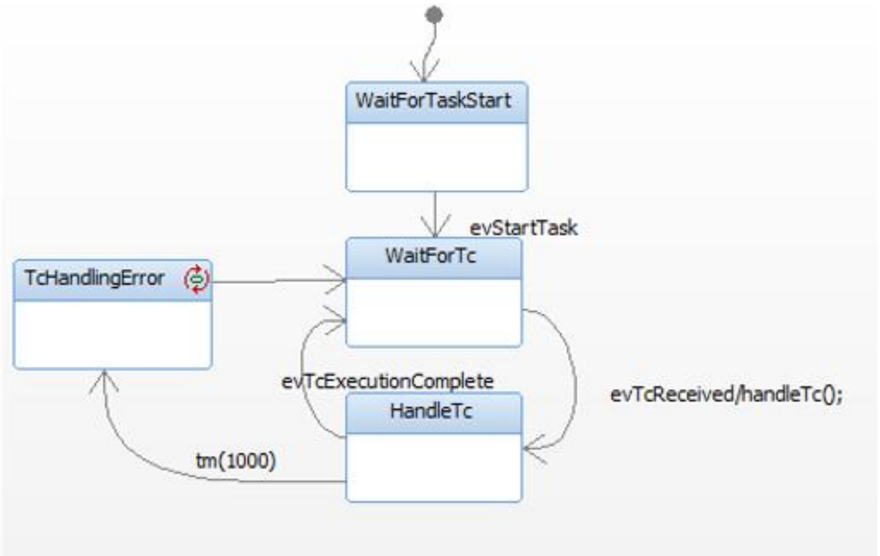
- Small GEO, Galileo, MTG

  (Approach for a common future Platform Software)

- Structural code is generated from component diagrams, state charts, sequence diagrams …
- Linking with natural language requirements in DOORS







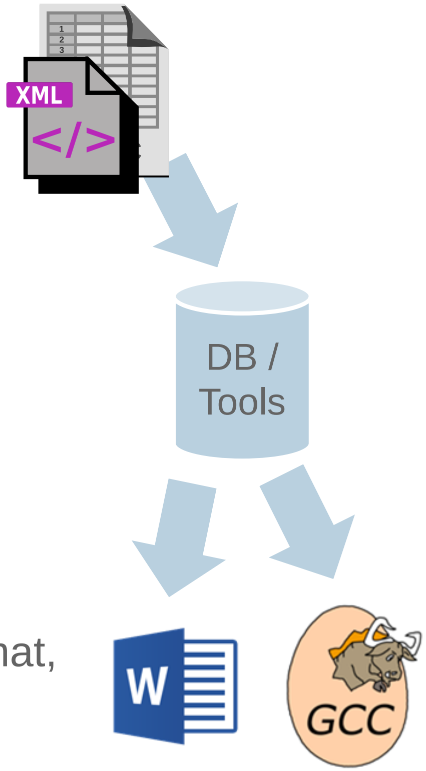
From practical experience - model based software development

- Entry Language UML / SysML
  - … is too limited
    - Need for additional domain specific elements (Attributes accessible (r/w) by ground, PUS compliant interface …)
  - … is too generic
    - An enforced restriction to established design pattern would mitigate impl. errors

  → Need to raise the abstraction level within the (space-)domain

- The “scope” of such tools is too limited to model the entire thing
  - Usually the C-level implementation is in-lined into the model
    → Functional implementation is not part of the model
  - The Exterior is not covered (avionics hardware)
    - Entire System and the deployment of the software within
    - Device Interfaces → EDS
    - …

EDS – Electronic Datasheets

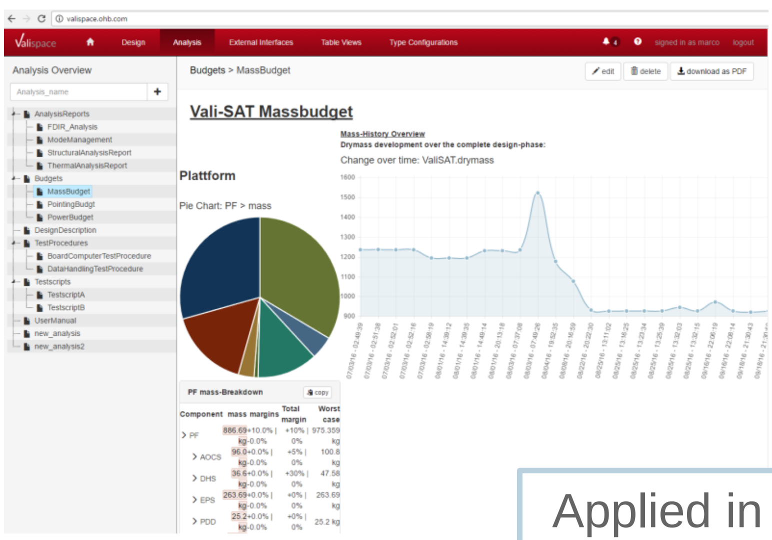


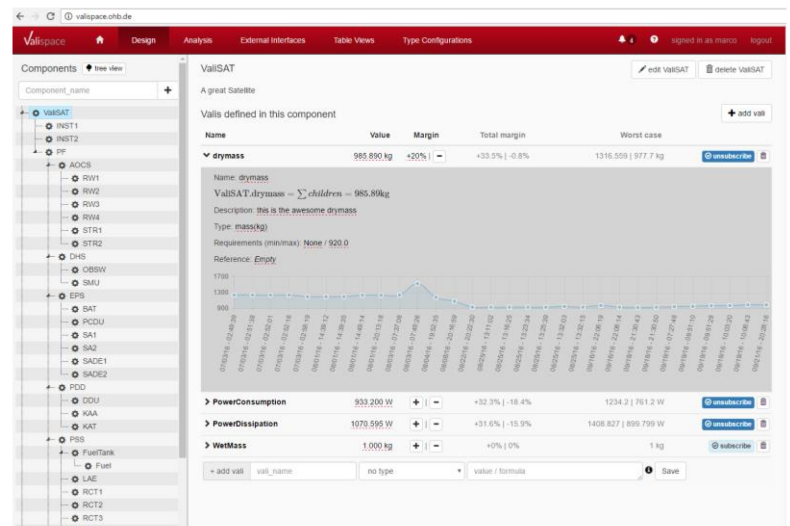
- Proprietary Approaches:
  - Based on Excel sheets
  - Based on XML
  - … reuse by different projects (to some extend)
- The issues are:
  - No Standard established
  - Supplier usually do not (correctly) provide data in the required format, resulting in additional work for the implementers
  - No elaborated model based tooling
    - Limited set of checks

    →even though it is a model we can't all the potential benefit from it

Model based Systems Engineering

- Browser based, collaborative COTS Tool for Phases A-E
- Maintains Engineering Data and Budgets
- Reports history, trends, requirement compliance
- Integrates with Word/Excel/Matlab/etc. via REST



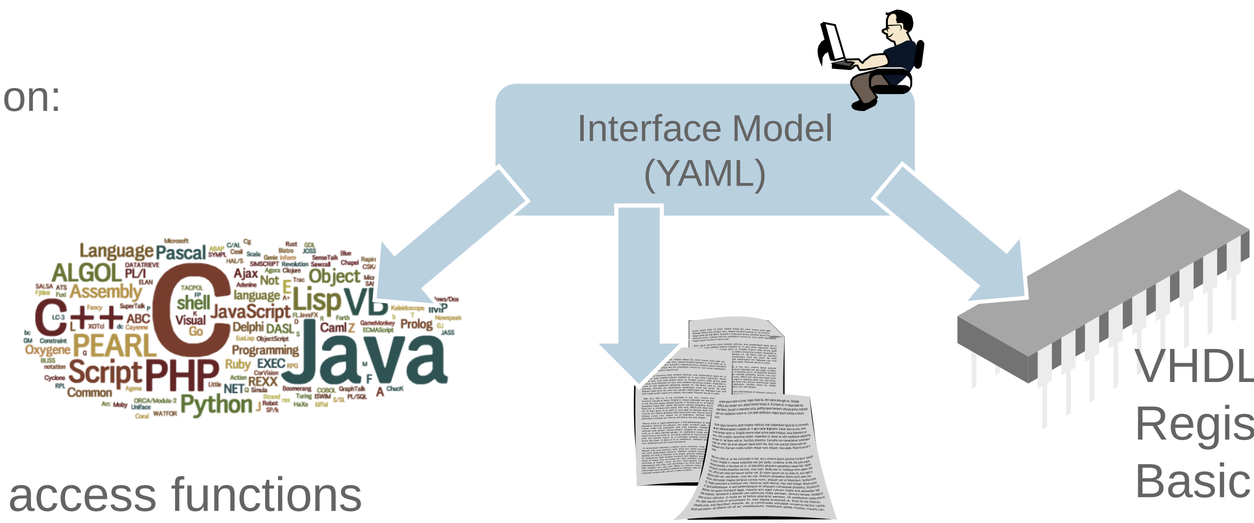


Applied in Pre-development Study – planned for C/D

Hardware/Software Codesign

- Problem:
  - Iterative Design Process between HW and SW → late design decisions
  - Precise Interface Definition needs to be communicated

- Solution:



Software access functions
Software emulation of hardware
(incl. failure injection)

Documentation
(HW/SW ICD)

VHDL interface
Register sets
Basic functionality
(reset, trigger …)

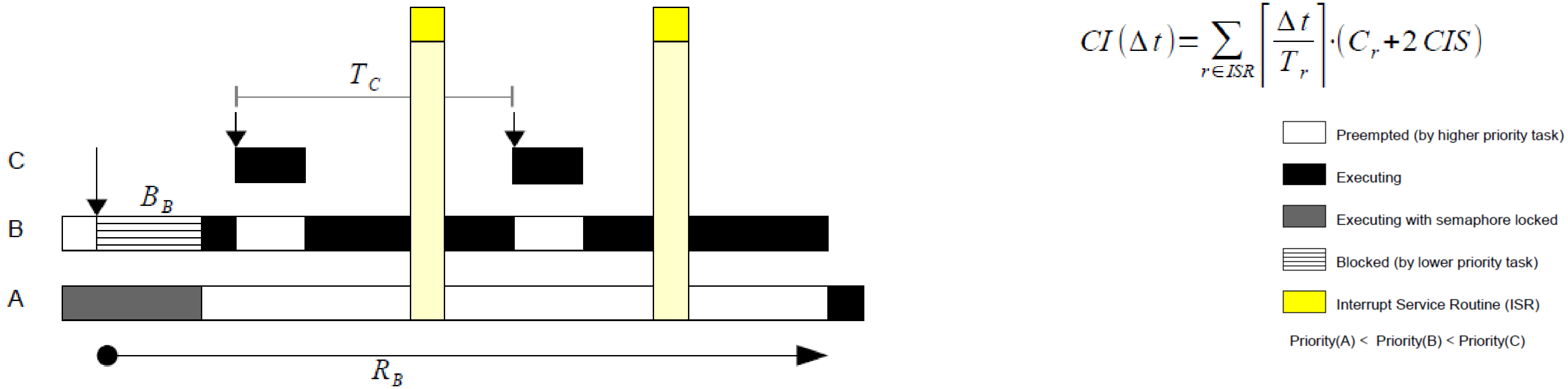
Full utilization in software and hardware development process.

Schedulability Analysis of Flight Software

- Computational Model of the Software
  - – a specific (mathematical) view of the model for timing analysis

$$R_i = CS + B_i + C_i + \sum_{j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil \cdot (C_j + 2\,CS) + CI(R_i)$$

$$CI(\Delta t) = \sum_{r \in ISR} \left\lceil \frac{\Delta t}{T_r} \right\rceil \cdot (C_r + 2\,CIS)$$



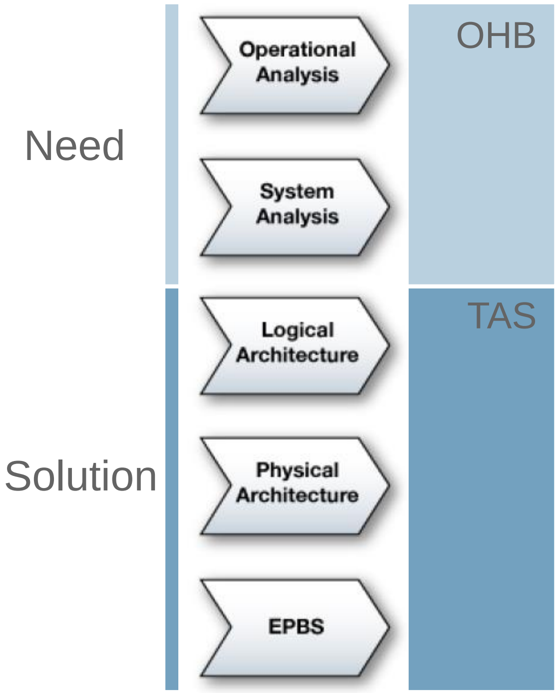
Full utilization in various commercial projects.

Ongoing Studies and Long-Term Options

- Capella in e.deorbit study (Systems-Engineering)
  - Functional Breakdown of the Mission
  - Model of Ground Segment
- Autonomous FDIR
- Model based Autonomy
- Extend the model to cover the C language

Capella used in e.deorbit study – Functional Breakdown of the Mission

- Capella implements the ARCADIA method: “Architecture Analysis and Design”



Functional Architecture described with Capella
(1) Functional breakdown of the system
(1) Functional dependencies and sequences
(2) Operational sequence diagrams
(2) Block/system diagrams
Consistency of the model checked
Derivation of functional system requirements from model

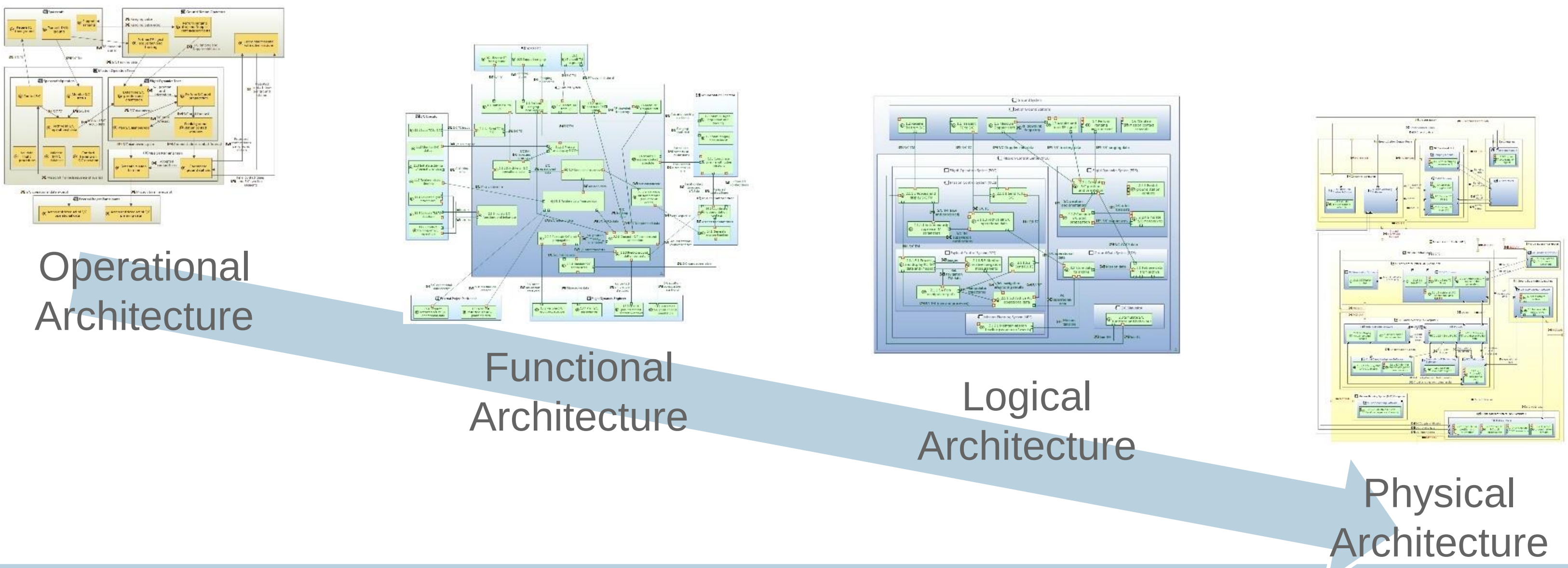
Further Refinement done by TAS
… model physical structures
… link with S/W development

Presented at SECESA, 10/2016

Capella used in e.deorbit study – Model of Ground Segment

Capella implements the Arcadia Method for model based architectural design

- Successive engineering phases by model transformations



→ Main Issues: Technical realization of tool

- Entering a model is time-consuming (proper arrangement of boxes and lines)
- Diagrams become convoluted rather quickly (when details are added)
- Modelling rules need to be obeyed (limited automatic guidance)

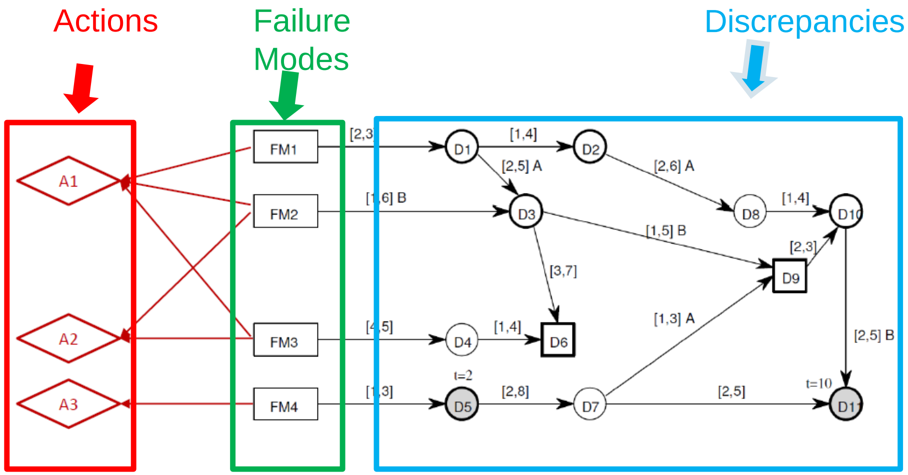
Autonomous FDIR

- (Extended) Time Failure Propagation Graph proposed as methodology for FDIR modeling
  - Model the dynamic behavior of a system when a failure has taken place
  - Formal mathematical model

  - Labeled directed graph
  - Nodes: failure modes (fault causes) and discrepancies (anomalous conditions)
  - Edges: model propagation of failures, annotated with condition and timing

- Extensions
  - Recovery actions
  - Sustainability time limit
  - Observability

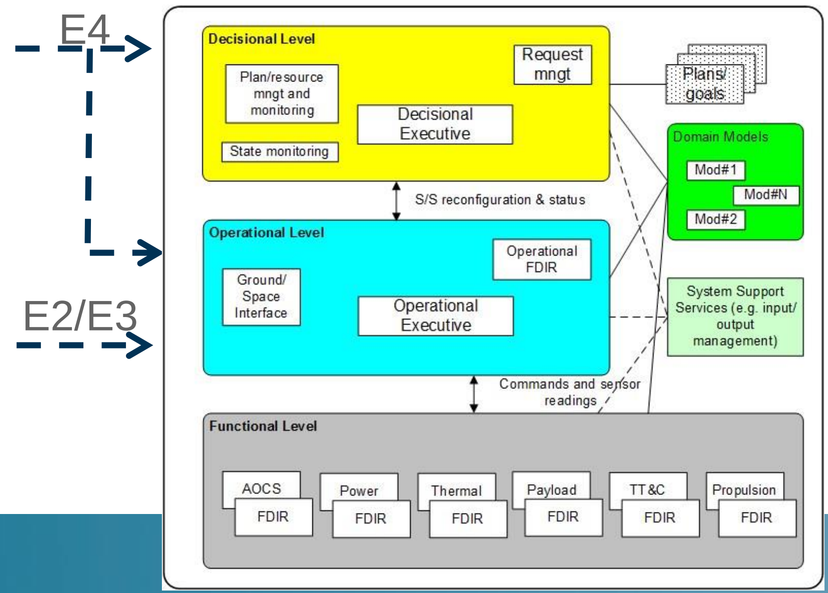
- Applications
  - Documentation and Analysis
  - Assist Development
  - Support Operations



Model based Autonomy – Study

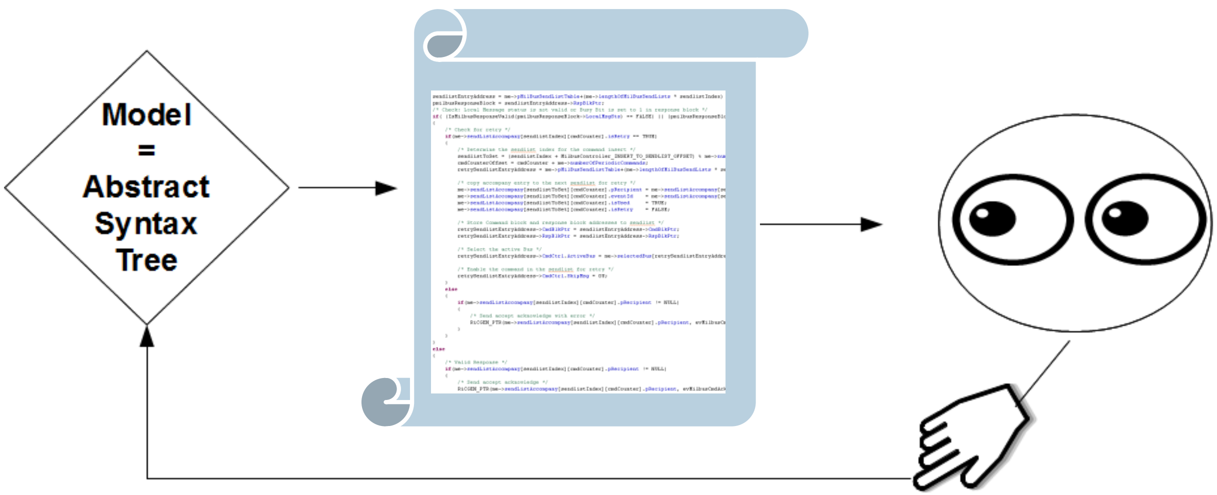
- Spacecraft autonomy: migration of functionality from the ground segment to the flight segment
- Different levels of spacecraft autonomy (ECSS-E-ST-70-11C – Space Segment Operability) based on factors such as mission types, mission objectives, operational concepts
  - Level E1: Flight Control Procedure (ground based)
  - Level E2: Mission Timeline
  - Level E3: OBCP
  - Level E4: Execution of goal oriented mission operations onboard

- Reconfiguration (decisional level) modelled via a Markov Decision Process (MDP) framework (→ mathematical model)
- MDP based approach for spacecraft autonomy at operational level



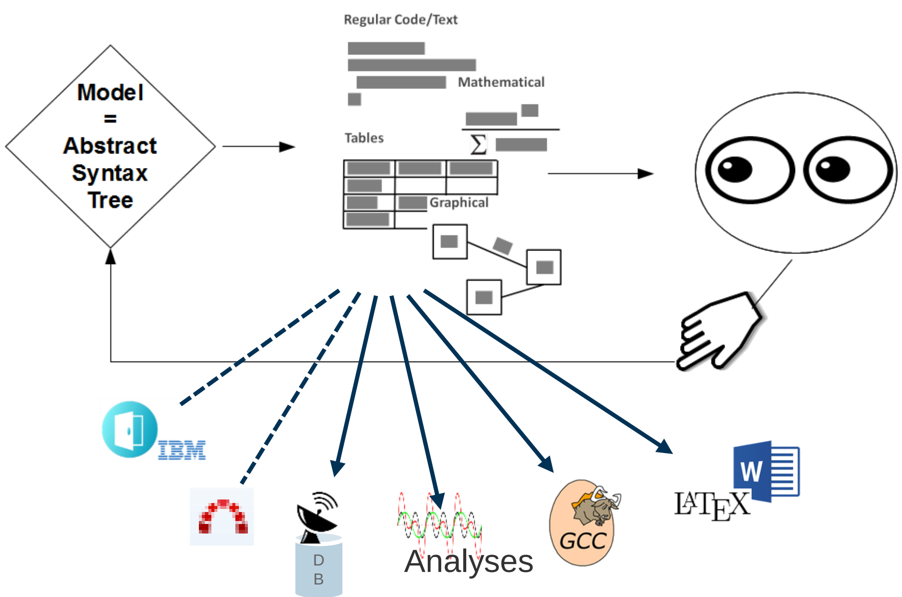
Extend the model to cover the C language

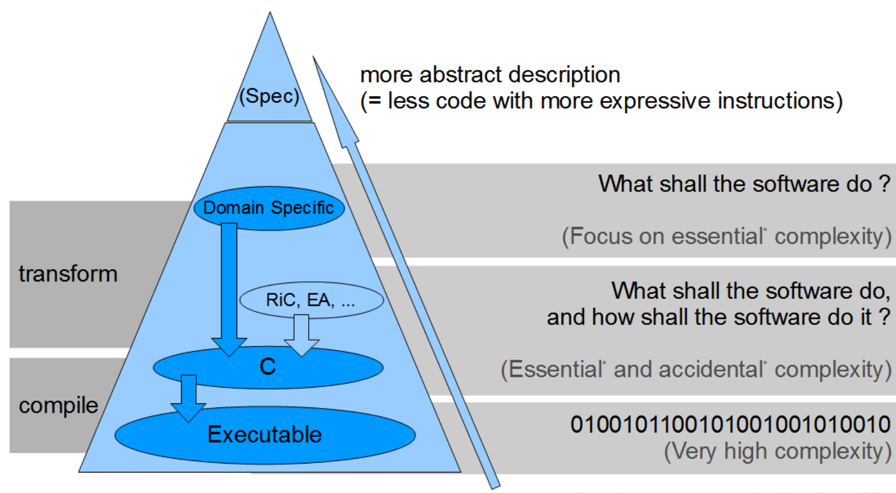
- From Lessons Learned in model based flight software development:
  - The C-level implementation is in-lined into the model → technically unrelated
  - Domain specific abstractions are missing
- Solution
  - C Language is implemented with a model based technology
    → A Domain Specific Language that looks like C and behaves like C
  - Blur the border between modeling and implementation **(Model == S/W Implementation)**



Extend the model to cover the C language

- Augment the C language with domain specific extensions
- Raise level of abstraction, respecting domain specific needs (Flight-, Simulator-, Ground- S/W)





* Essential and accidental complexity as identified by Frederick P. Brooks in 1986, „No Silver Bullet – Essence and Accidents of Software Engineering“

Summary and Vision

- There are a lot of model based activities ongoing with different approaches and intentions.

  → We need to integrate the various approaches for even better results.

- We see a strong tendency towards **model based** and **(domain) specific** tooling
  - Increasing Complexity is mitigated by
    - Shifting domain specific knowledge into the tooling
    - Specific and limited views on the model
  - Models enable
    - Single source paradigm (up-to-date documentation, configuration …)
    - Formal checks (verification, plausibility …)

- In most cases the technology (engaged) is not yet capable to deliver all the advantages that theoretically can be drawn from model based approaches.

OHB System AG
Andreas Wortmann
18.10.2016, ADCSS

Thank you !

SPACE SYSTEMS

Rationale, recent projects and future options for Model based Developments

We. Create. Space.