GEFÖRDERT VOM
Bundesministerium
für Bildung
und Forschung

SpesML - SysML Workbench for the SPES Methodology

Dr. Maximilian Junker (Qualicen / Technical University of Munich)

Dr. Wolfgang Böhm (Technical University of Munich)

Before we start: who we are

Dr. Wolfgang Böhm

- Project coordinator of the SPES series of research projects
- Core contributor to the SPES modeling framework
- 25+ years of industry experience

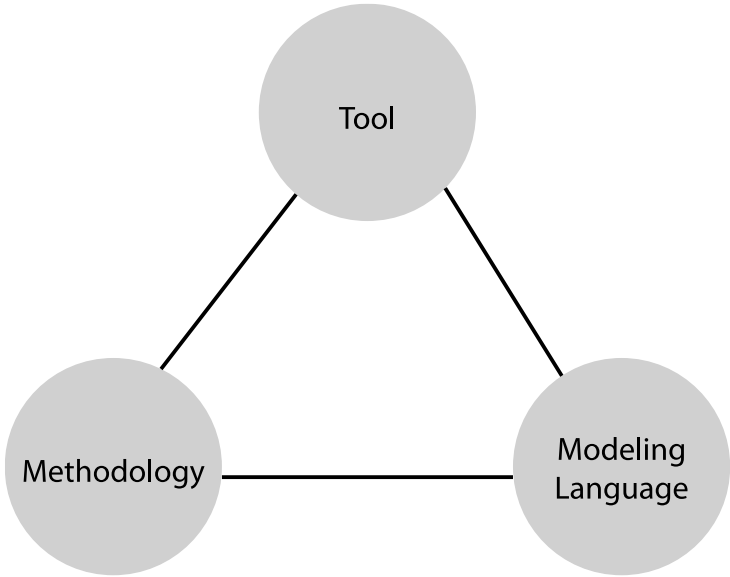
Dr. Maximilian Junker

- Co-founder of Qualicen
- Core contributor to the SPES modeling framework
- MBSE professional consulting companies during introduction of MBSE

MBSE: Technical and Methodological Challenges

Experiences from a variety of MBSE introduction projects:

- SysML was chosen as the modeling language, however
  - SysML focus is mainly on syntax
  - Lots of language features to implement a wide range of methodologies
  - No guidance which language elements to be used
- Missing precise semantic definition of concepts and models
- No consistent methodology or proprietary methodology driven by tool vendors
- Tools provide a wide range of features and no or only little support how to use these features in a given project



scientific well-founded semantics

We need

1. to choose a suitable methodology, based on a scientific foundation, a matching language and an appropriate modeling tool. These three aspects must be well coordinated with each other;
2. support to guide the MBSE introduction process.

SPES Framework Sets the Methodological Basis

GEFÖRDERT VOM
Bundesministerium für Bildung und Forschung

SPES ML

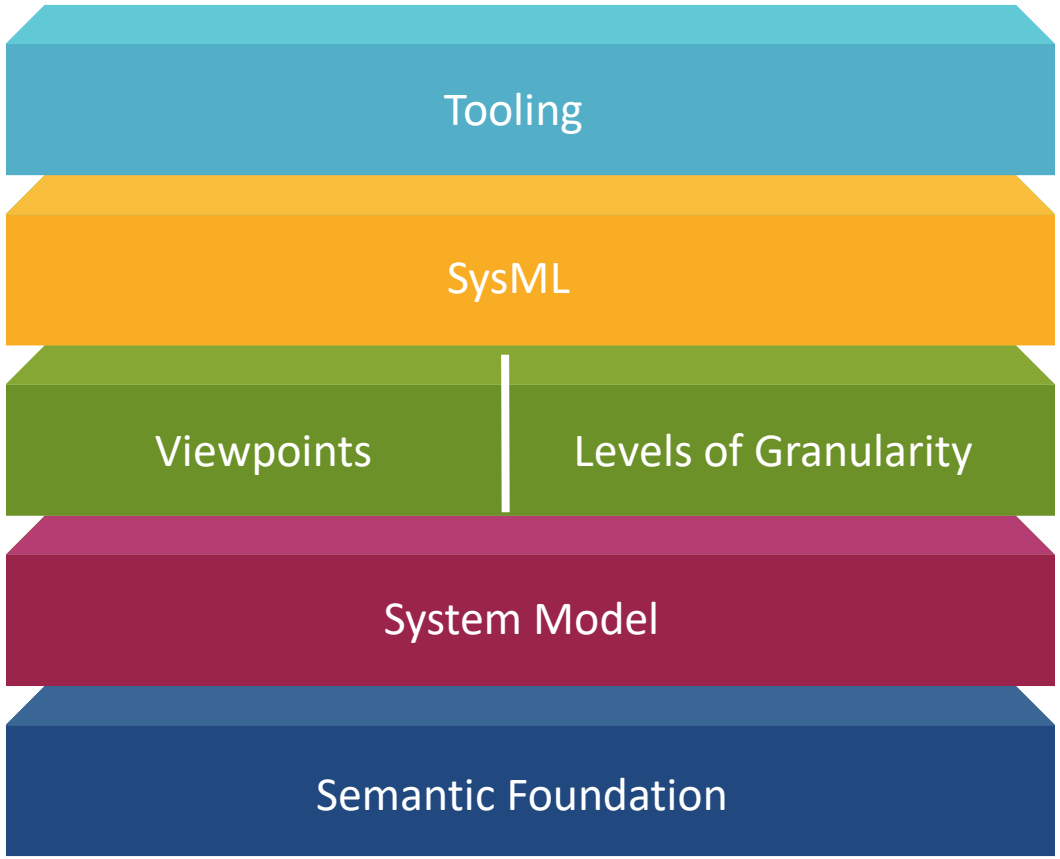
- "**S**oftware **P**latform for **E**mbedded **S**ystems" - the SPES Series of Research Projects
  - SPES2020 – setting the basis
  - SPES_XT – extensions of the SPES framework
  - CrESt – including modeling of collaborative embedded systems
  - SPEDiT – tutorials to support transfer into practical application
- 45+ partner from industry and academia contributed to the research (only a few shown here)

The SpesML Project: SysML Workbench for the SPES Methodology

- Goals
  - Focus on implementation of a concrete modeling methodology (SPES) with SysML language elements to allow contextual system specification → Mapping of the SPES modeling framework to SysML
  - Consistent specification of SPES concepts using SysML language elements and (maybe) extensions of the language
  - Semantic and methodological foundation
  - Development of concrete analyses of the execution semantics
  - Prototypical implementation of a lightweight tool environment
- Project duration
  - 2021/01/01 – 2022/12/31

- Partner

The Building Blocks of SPES ML



Tooling: Tooling based on CATIA Magic Draw enables efficient modeling

SysML: SpesML provides a mapping of the SPES modelling method to SysML

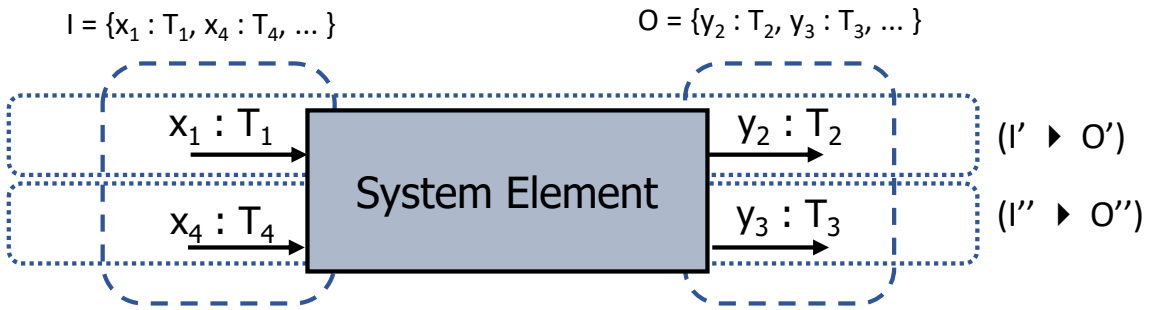
Viewpoints: SPES provides a set of models for system modelling structured in viewpoints. The basic viewpoints are Requirements VP, Functional VP, Logical VP, Technical VP

Levels of Granularity: Systems can be de-composed into subsystems and the structed SPES engineering method can be applied recursively.

System Model: The SPES system model builds on the semantic foundation and provides the basic concepts of system, operational context and subsystems.

Semantic Foundation: The Focus-Theory provides a solid mathematical base for all elements and concepts in SpesML including interfaces, behavior and composition.

Semantic Foundation – Focus Theory



Sets of typed channels

$I = \{x_1 : T_1, x_4 : T_4, \ldots\}$
$O = \{y_2 : T_2, y_3 : T_3, \ldots\}$

Syntactic interface

$(I \blacktriangleright O)$

Syntactic sub-interface

$(I' \blacktriangleright O') = (\{x_1 : T_1\} \blacktriangleright \{y_2 : T_2\})$

Stream of type T

$STREAM[T] = \{\mathbb{N}\backslash\{0\} \rightarrow T^*\}$

Valuation (history) of channel C

$\mathbb{H}[C] = \{C \rightarrow STREAM[T]\}$

Interface behavior for syntactic interface (I ▸ O)

$[I \blacktriangleright O] = \{\mathbb{H}[I] \rightarrow \wp(\mathbb{H}[O])\}$

Interface specification

$S: I \cup O \rightarrow \mathbb{B}$

represented as interface assertion S (logical formula with channel names as variables for streams)

For details see:

M. Broy: A Logical Basis for Component-Oriented Software and Systems Engineering. The Computer Journal: Vol. 53, No. 10, 2010, S. 1758-1782

Semantic Foundation – Behavior Specification

Different Specification Techniques for Behavior

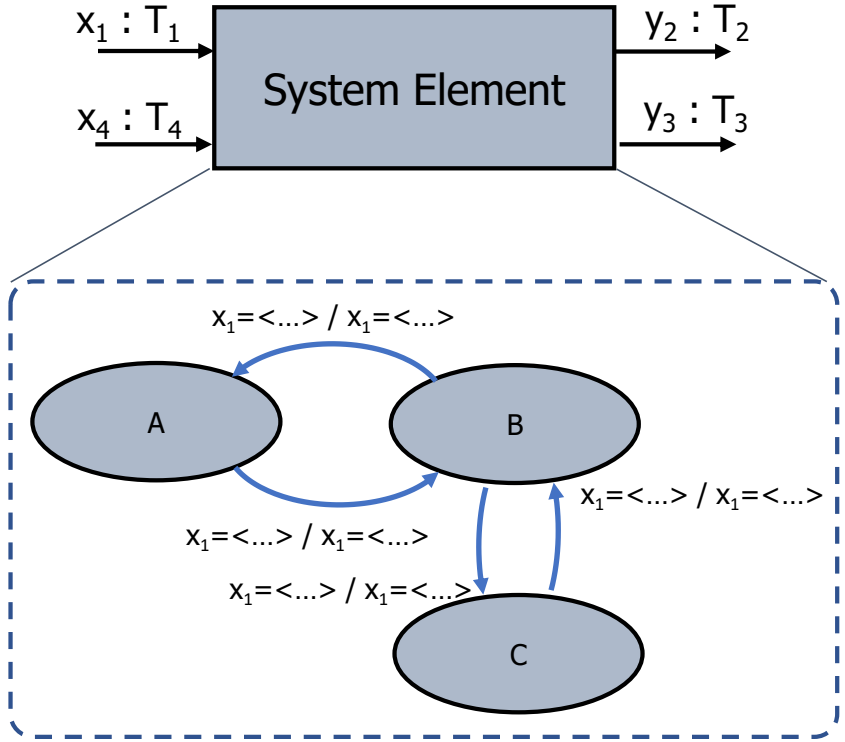
- Interface Assertions
- Assumption / Guarantee
- **State Machines**

Handling of Time

- Timed
- Untimed

Different Variants of State Machines

- Event-based State Machines
- Time-Slice-based State Machines



Semantic Foundation – Composition

Interface Behaviors

F: [I_1 ▸ O_1]

G: [I_2 ▸ O_2]

Syntactic Interface Composition

$F \otimes G$: (I ▸ O)

where:

$I = (I_1 \cup I_2) \setminus C$

$O = (O_1 \cup O_2) \setminus C$

$C = (I_1 \cup I_2) \cap (O_1 \cup O_2)$

Composed Behavior

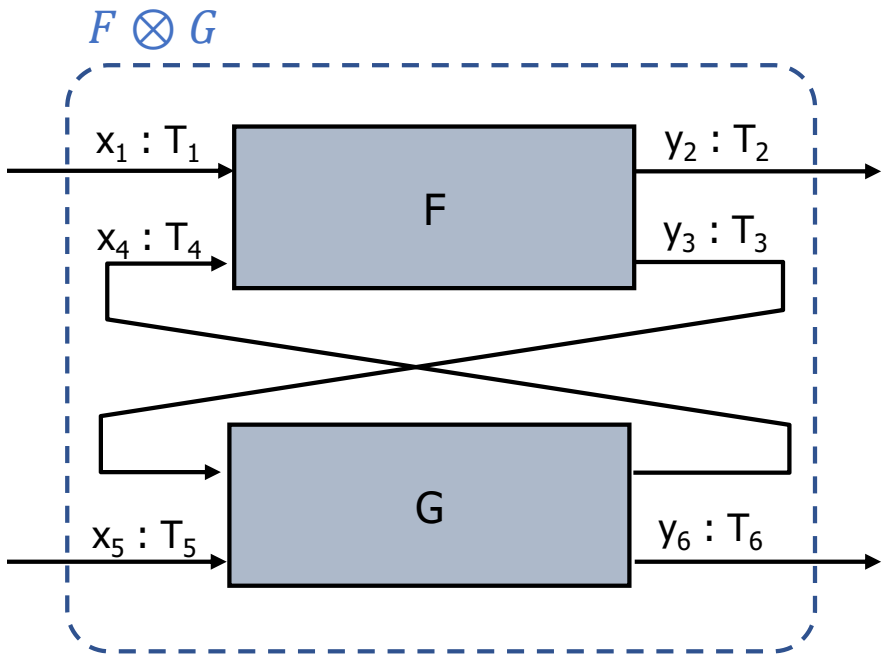
$(F \otimes G)(x) = \{y|O : y \in \mathbb{H}[Z] \wedge y|I = x \wedge y|O_1 \in F(y|I_1) \wedge y|O_2 \in G(y|I_2)\}$

where:

$Z = I_1 \cup O_1 \cup I_2 \cup O_2$

Further Topics

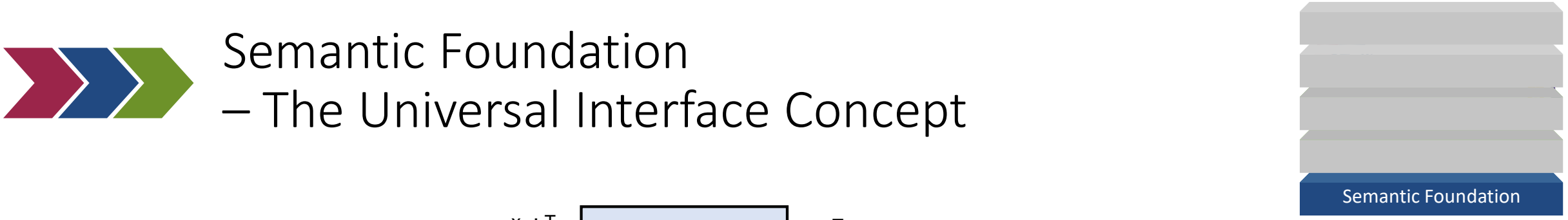
- (Weak/Strong) Causality
- Realizability

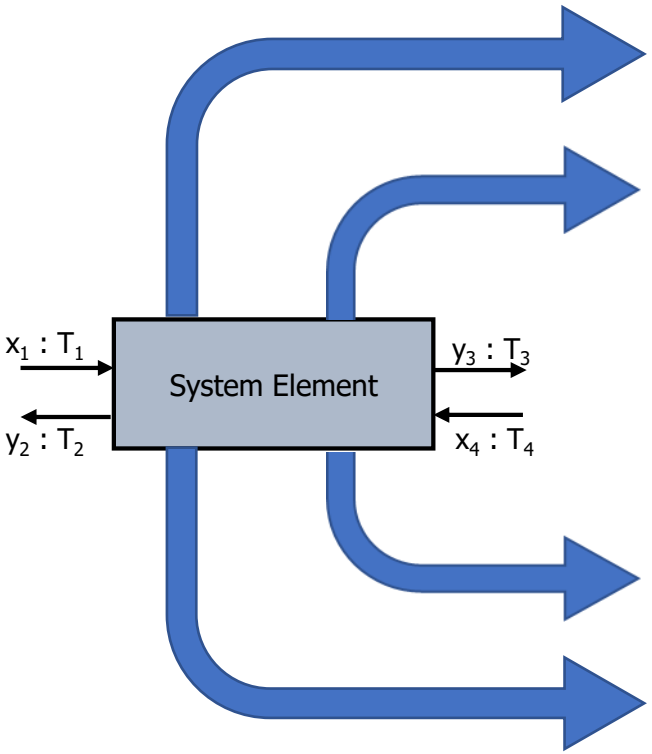


For details see:

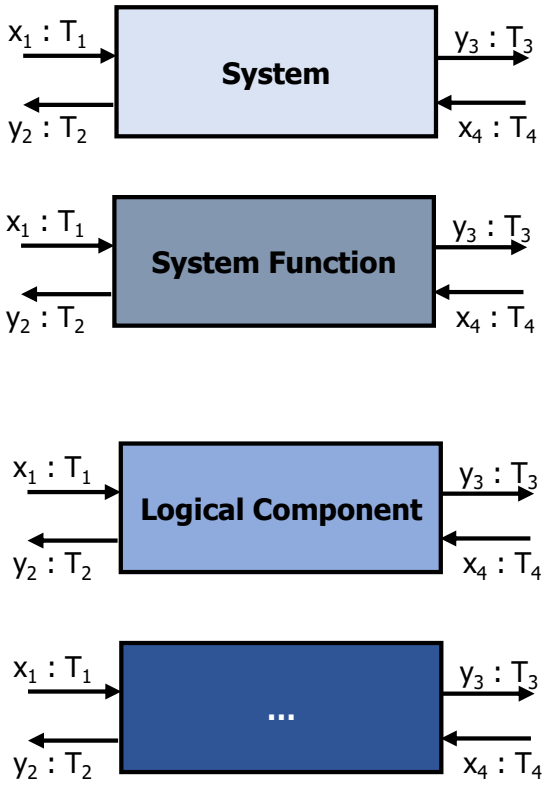
M. Broy: A Logical Basis for Component-Oriented Software and Systems Engineering. The Computer Journal: Vol. 53, No. 10, 2010, S. 1758-1782

Semantic Foundation – The Universal Interface Concept





Abstract Universal Interface Concept



Application for different types of System Elements

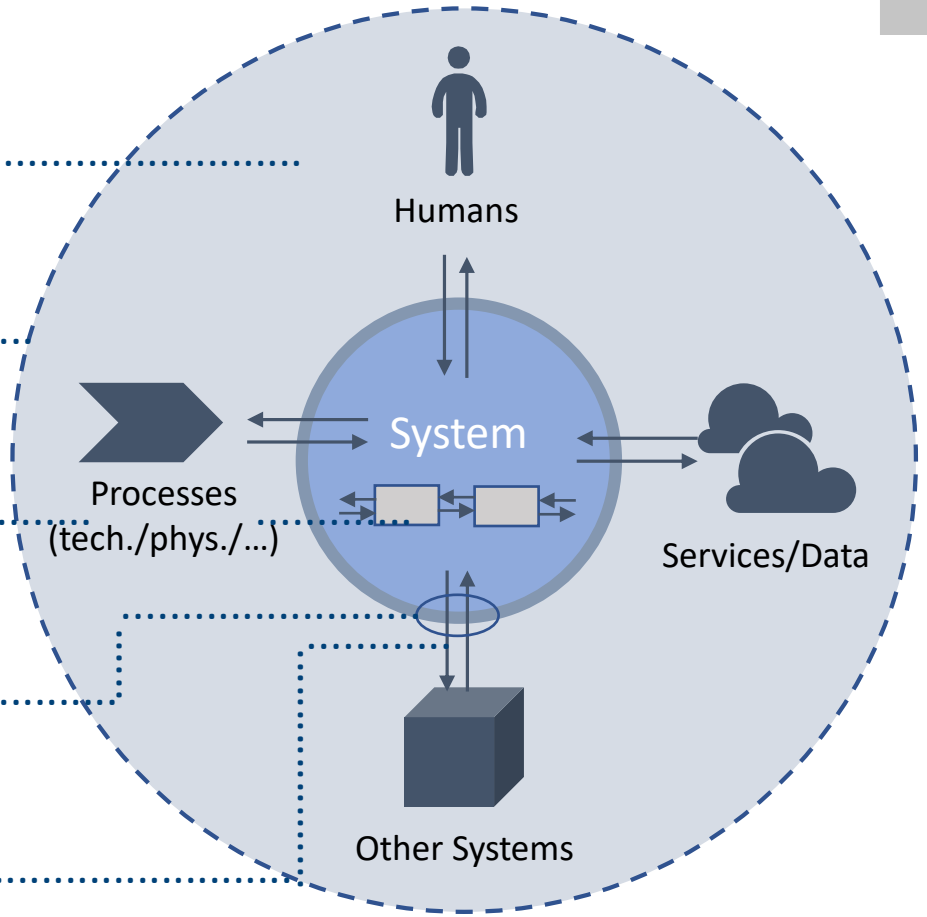
Advantages

- Compatible interface and behavior concept throughout the whole framework
- Allows for simulation and analysis across different models and viewpoints
- Enables specification of refinement between related system elements in different models and viewpoints
- Enables mode reuse

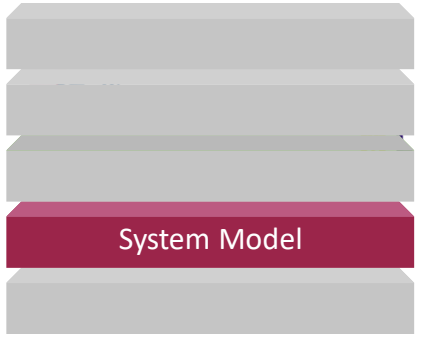
The SPES System Model

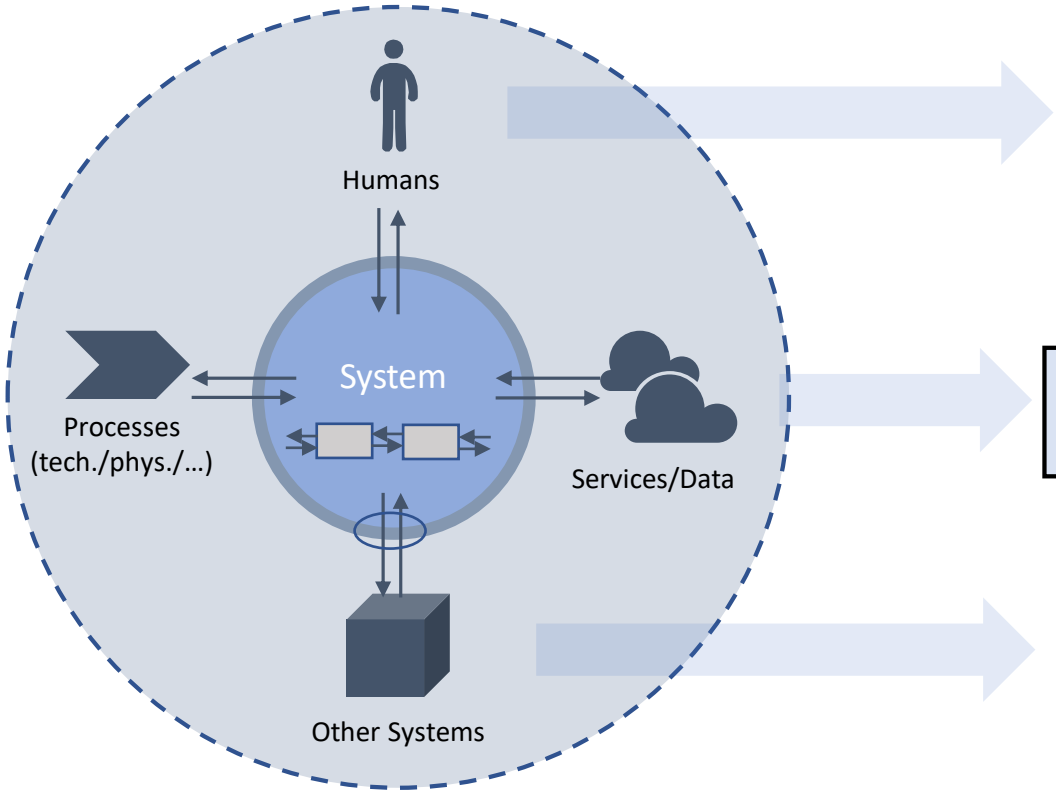
A **System** (System under Development - SuD) has:

1. An **operational context** that influences or is influenced by the SuD
2. The **context boundary** separates the operational context from non relevant environment
3. An **inner structure** of related elements
4. **Interfaces** that clearly distinguish the SuD from its context and define the **system boundary**
5. **Behavior** that is observable at these interfaces

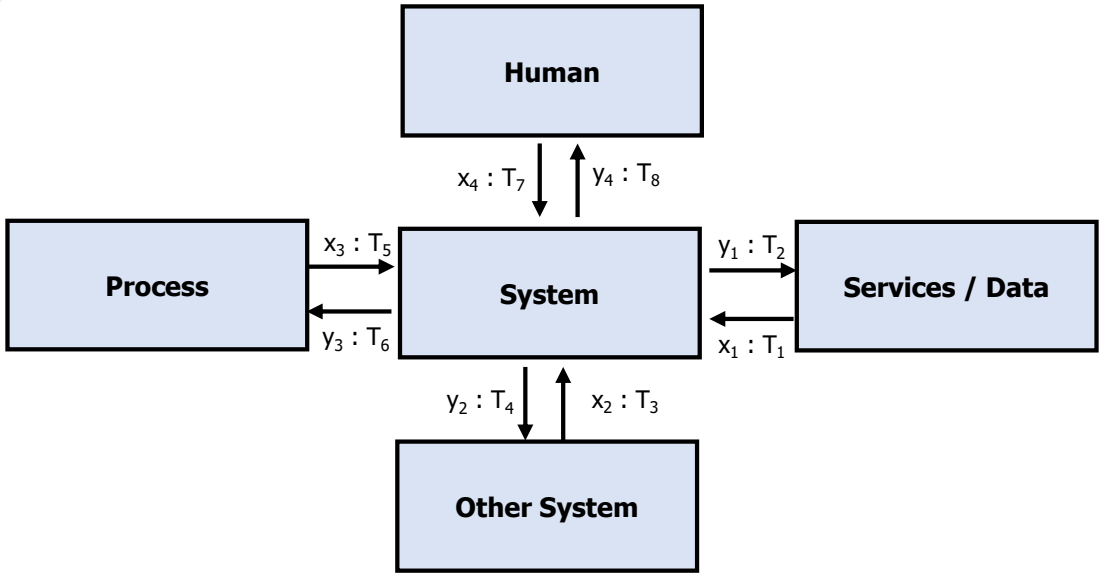


The SPES System Model – Representation within the Universal System Model



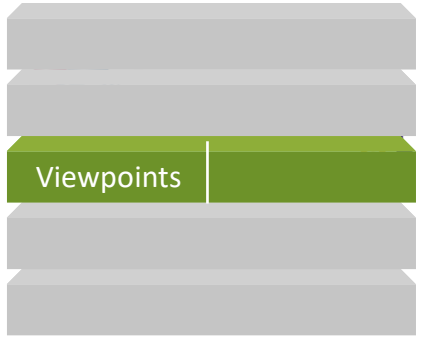


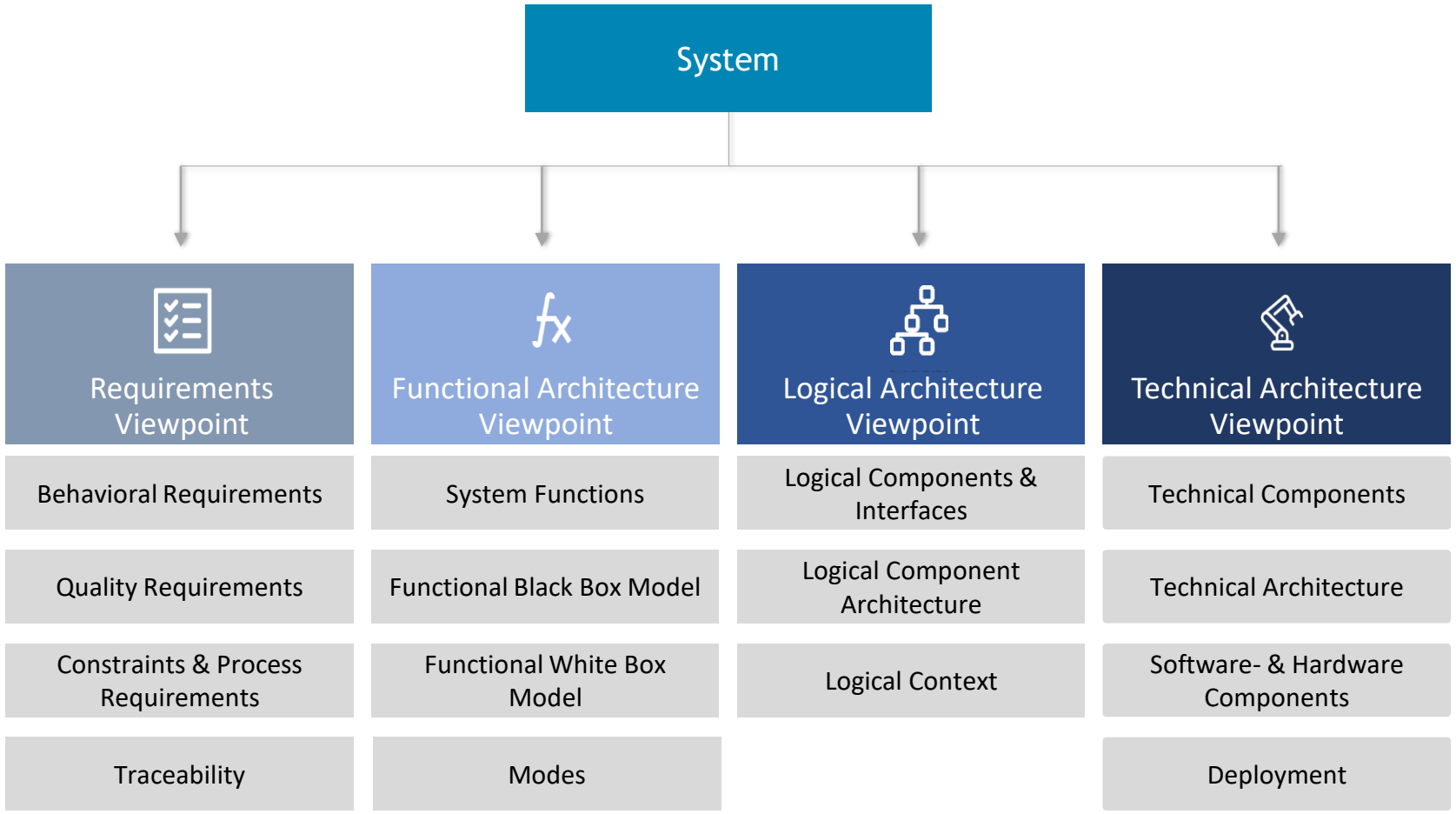
The SPES System Model



Representation within the Universal Interface Concept

SPES - Viewpoints

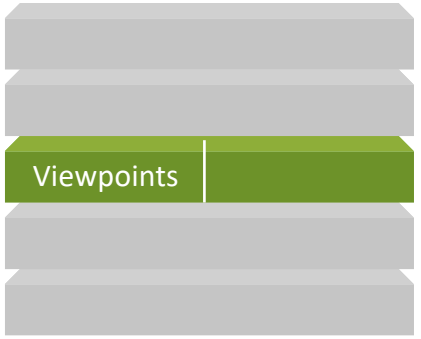




SPES Viewpoints

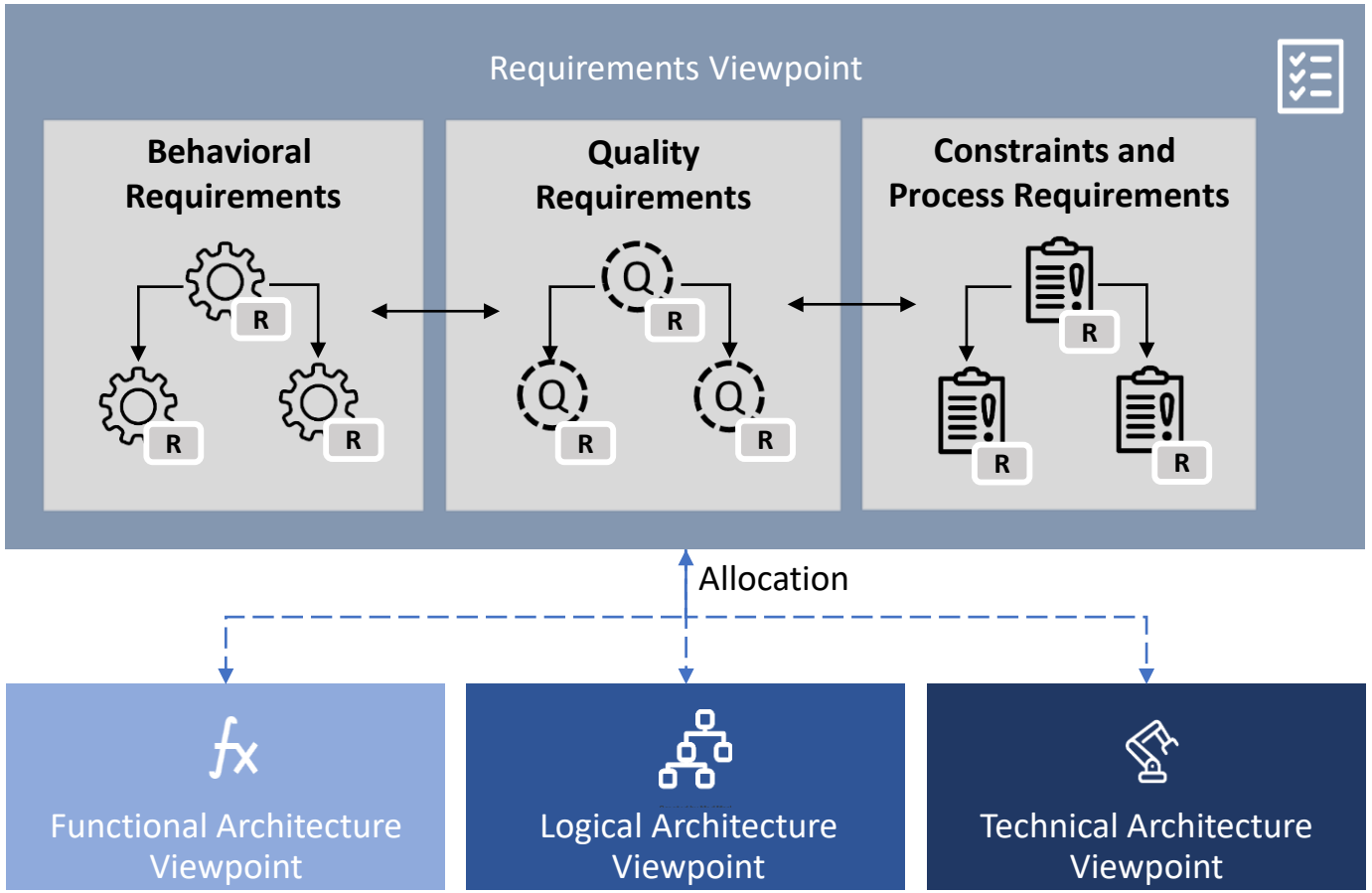
- Four viewpoints following ISO 42010
- Providing different abstractions
- Addressing different stakeholder concerns
- SPES Framework allows for adding more viewpoints

Requirements Viewpoint

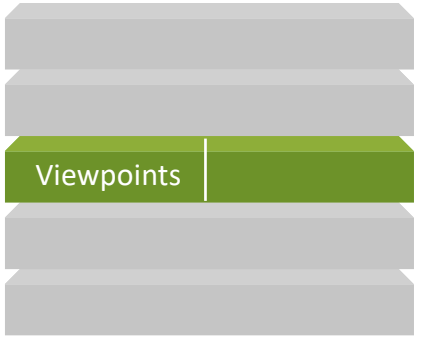


Modeling Goals

- Definition of (behavioral) requirements
- Definition of quality requirements
- Documenting system constraints and process requirements

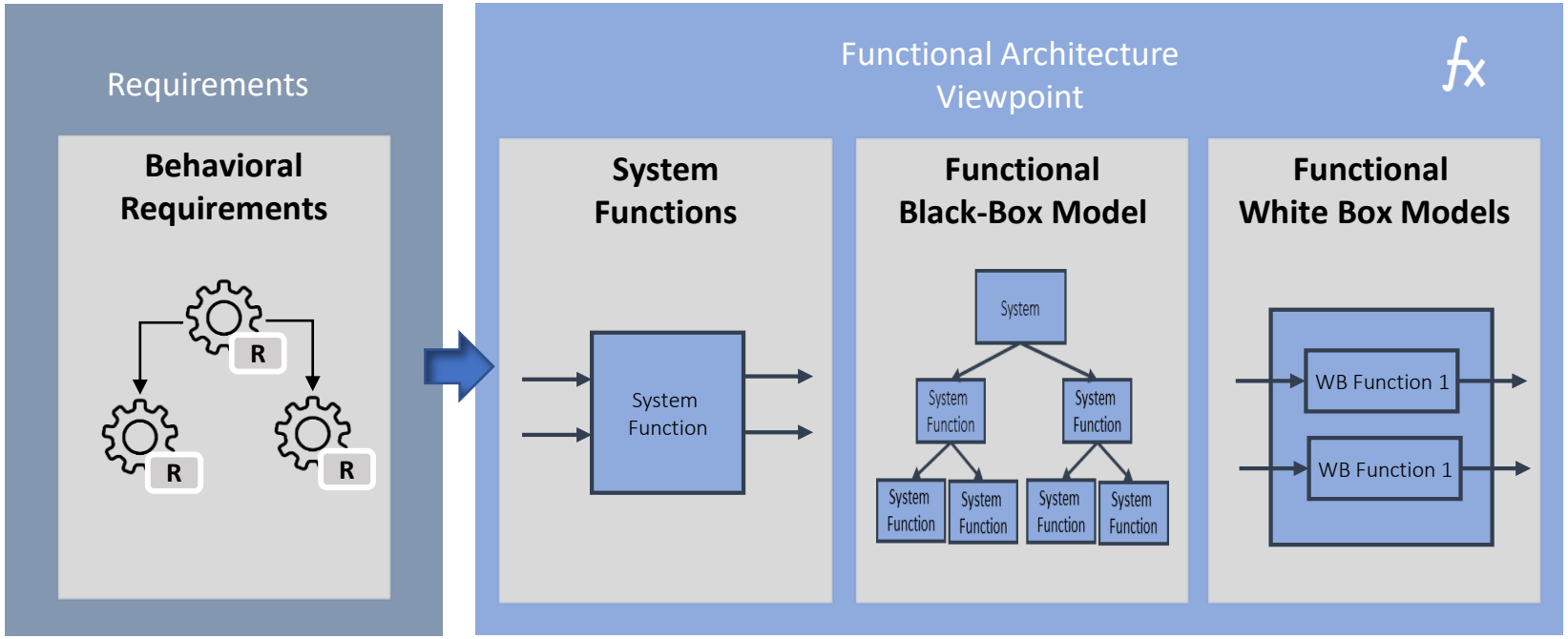


Functional Viewpoint

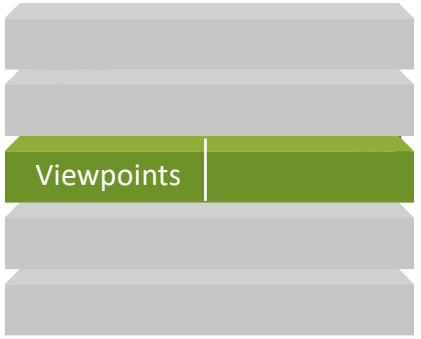


Modeling Goals

- Integration of behavioral requirements into a **system specification**
- Precise modeling of functional **black box behavior**
- Modeling of **dependencies** between behavioral requirements (System Functions) - Modes
- Decomposing System Functions to enable mapping to logical components (White Box Models).

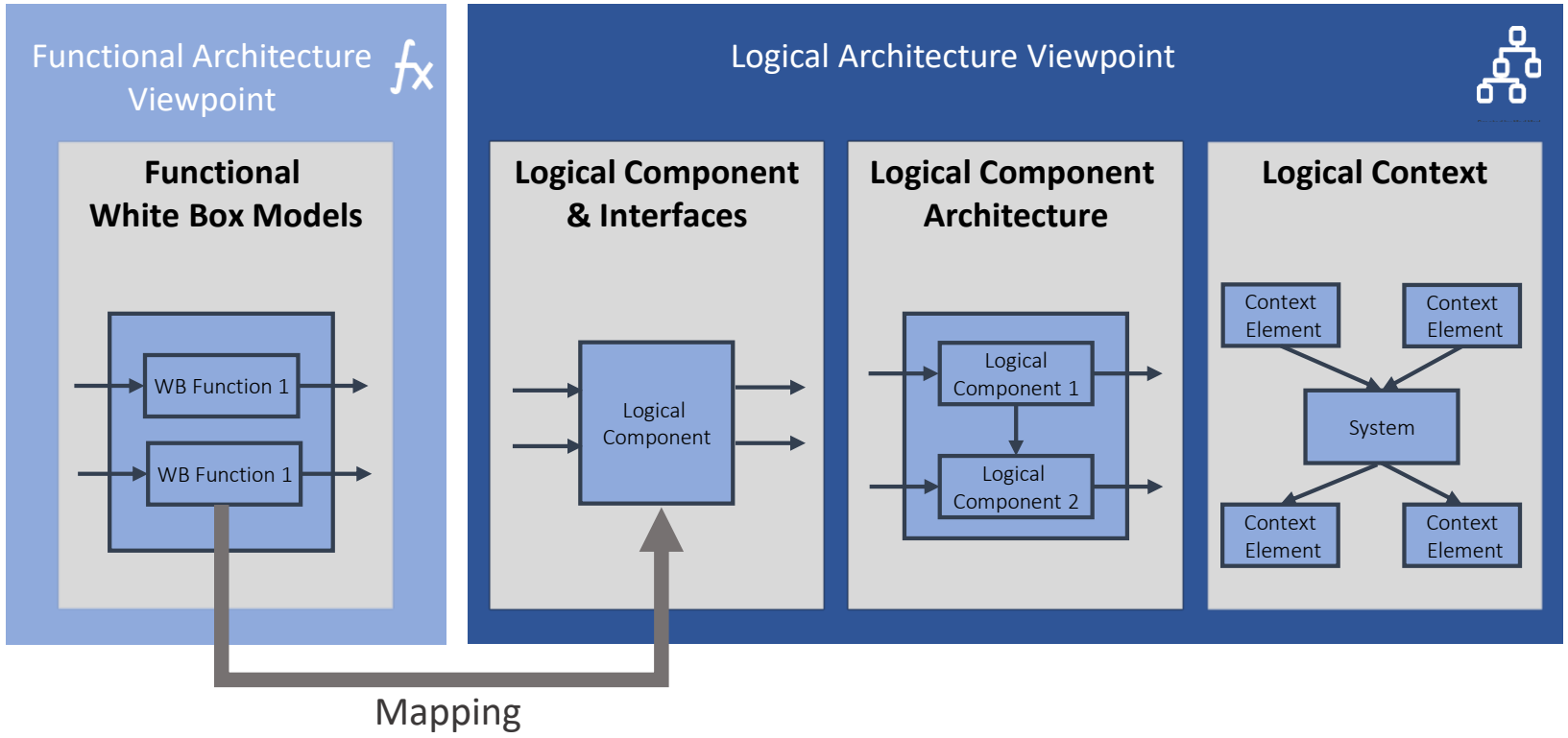


Logical Viewpoint

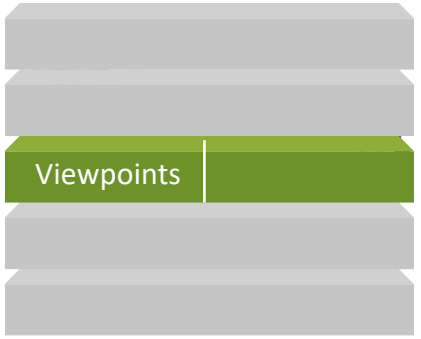


Modeling Goals

- Description of the internal logical (platform-independent) structure of the SUD by decomposing it into logical components
- Allocation of white-box functions to logical components
- Definition of the total behavior of the system
- Reuse of existing logical components
- Isolation of software subsystem

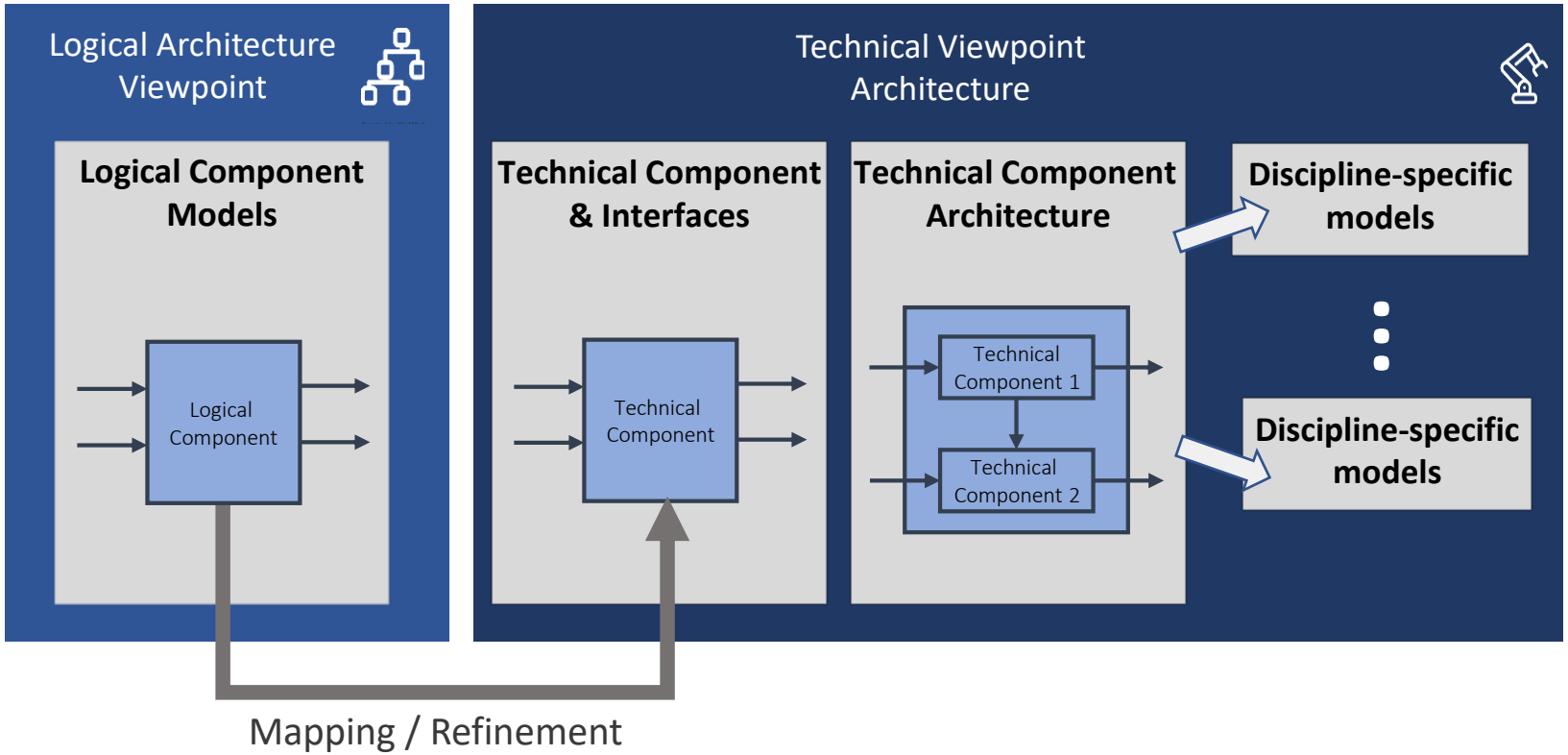


Technical Viewpoint

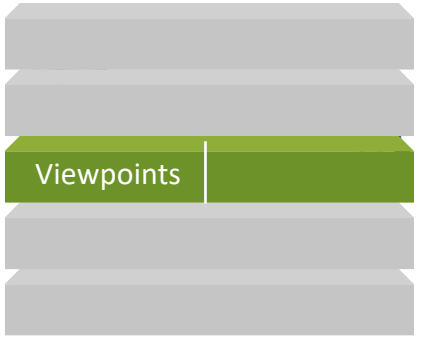


Modeling Goals

- Platform-specific refinement of the logical components (technical components)
  - Mechanical Components
  - Electronic Components
  - Software Subsystems (Application SW **together with the execution hardware**)
- **Interfaces** between components of different engineering disciplines
- Discipline-specific models of the software subsystems

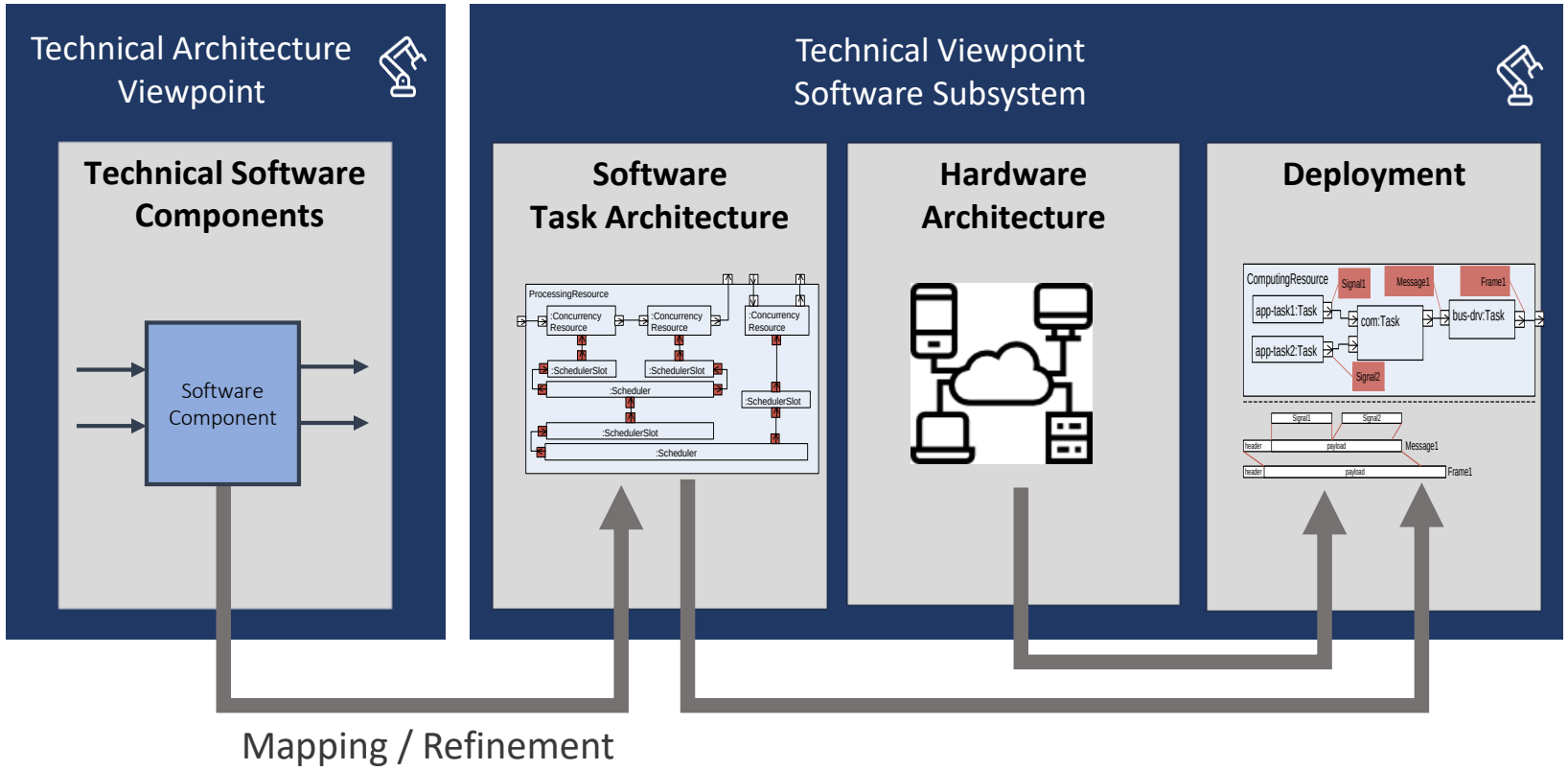


Technical viewpoint (Models of SW Subsystems)



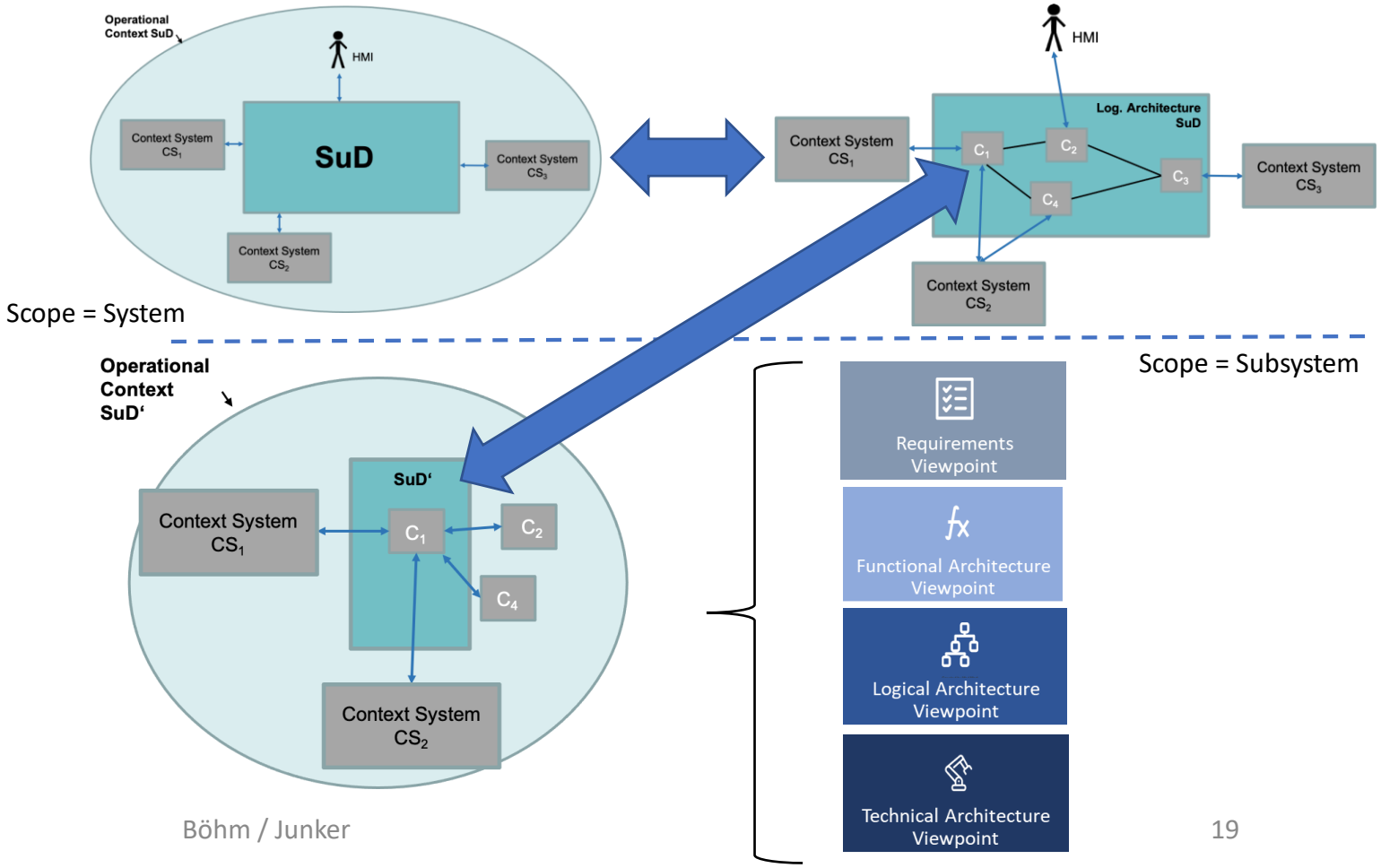
Modeling Goals

- **Platform-specific** description of the **software subsystem**
- Modeling of the **target hardware** (ECUs, buses, memory, ...)
- Definition of **(software) tasks and schedulers**
- Modeling of the **distribution-specific communication**
- **Deployment**



From System to Subsystems – Levels of Granularity

- Elements from the Logical Component Architecture can be designated as subsystems
  - Scope of the SuD changes
  - Universal Interface Concept applies at the border of the subsystem, enabling re-integration into the super-system
  - Any development framework, process, and tools may be used for the development of subsystems
  - In particular, the SPES framework may be recursively applied (e.g. for SW-subsystem)
- The concept of architecture layers (levels of granularity) enables model reuse and supplier integration



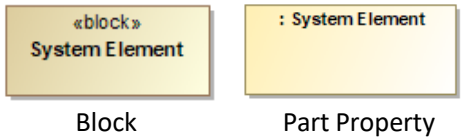
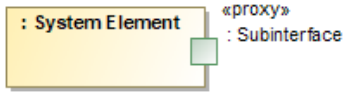
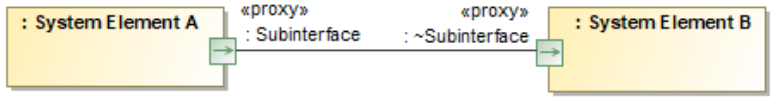
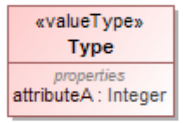
SPES ML – From Concepts to the Tool

For implementing SPES ML in a tool, we proceed in two steps:

Disclaimer: This is still work in progress

SPES ML – The SPES Method in SysML

Universal Interface Concept in SysML

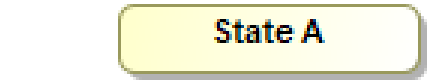
Concept in SPES	SysML Concept	
System Element	Block Part Property	
Syntactical (Sub-)Interface	Proxy Port Interface Block	
Channel	Flow Property	
Channel Matching/Renaming	Connector	
Data Types	Value Type	
…	…	

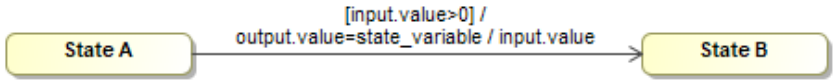
+ Constraints

SPES ML – The SPES Method in SysML

Interface Behavior Concepts in SysML

Concept in SPES	SysML Concept
State Symbol	State
Extended State Variable	Value Property in the owning block
State Transition	Transition Guard Effect + Opaque Behavior
…	…



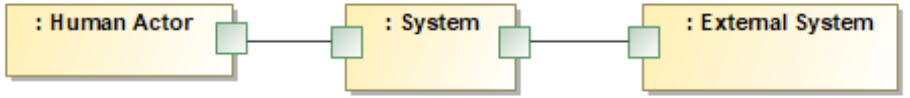


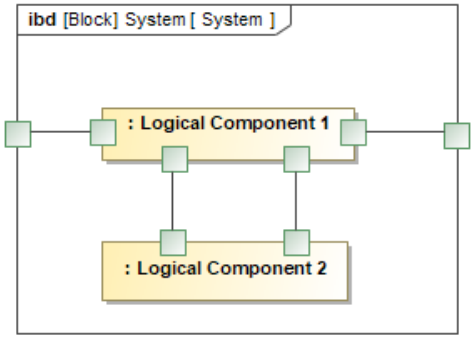
+ Constraints
+ Expression Language

SPES ML – The SPES Method in SysML

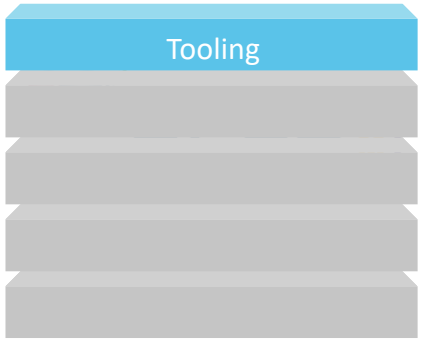
Logical Viewpoint in SysML

Concept in SPES	SysML Concept
Logical Viewpoint	Package
Logical Context	Internal Block Diagram
Logical Component Architecture	Internal Block Diagram
…	





SpesML Workbench for the SPES methodology



~Target for SpesML

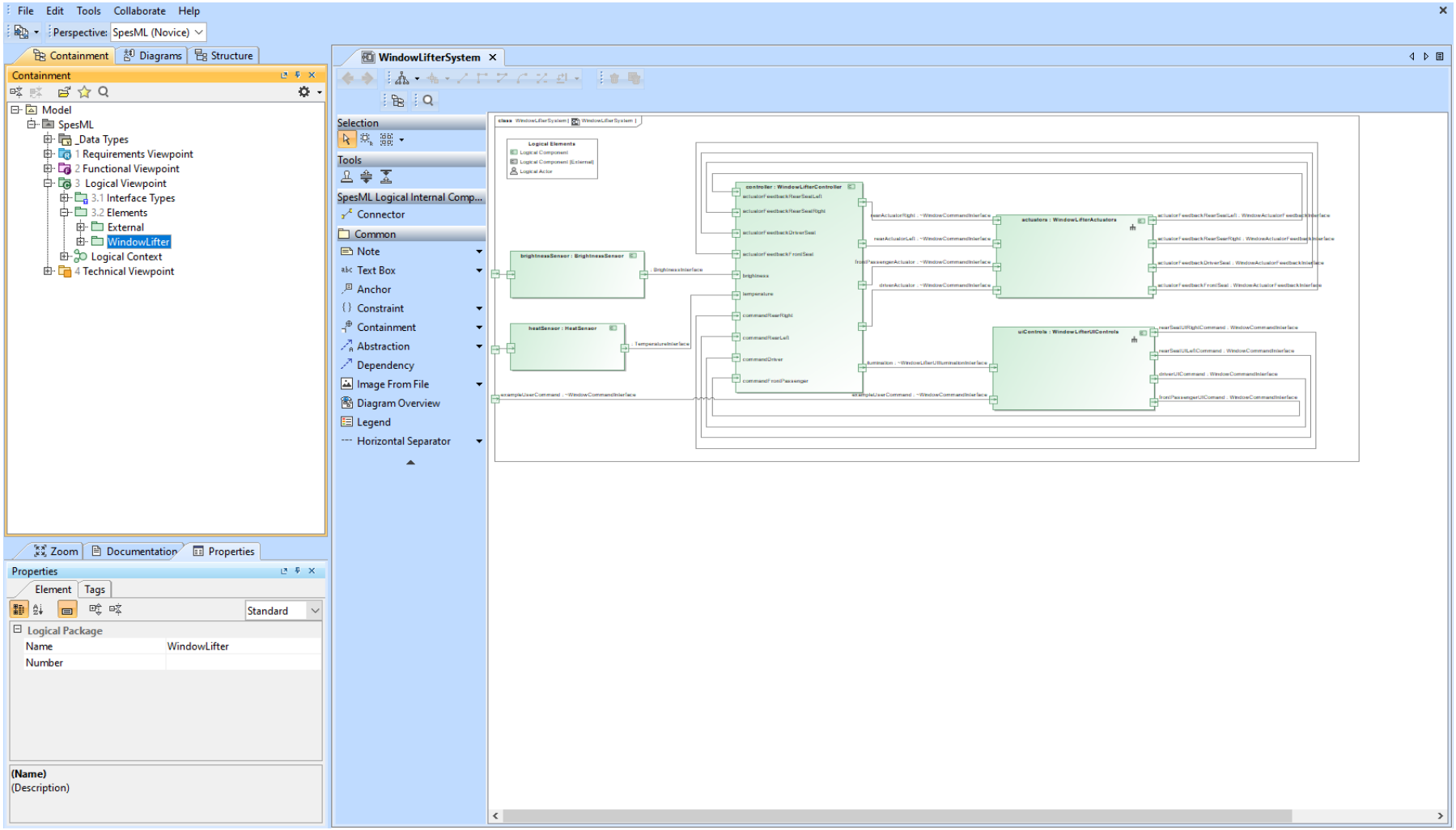
- Very simple SpesML profile
  - Usage of SysML elements
  - Usage of SysML diagrams
  - Manual application of stereotypes

- Very close to SysML standard
- Lower usability
- Lower user guidance
- No tight integration between language, method and tool

- Highly customized SpesML profile
  - Custom Structure
  - Custom Elements
  - Custom Diagrams
  - …

- Further away from SysML standard
- High usability
- High user guidance
- Better integration between language, method and tool

Demo – Window Lifter



GEFÖRDERT VOM
Bundesministerium
für Bildung
und Forschung

https://spesml.github.io/index.html

Email: maximilian.junker@qualicen.de
boehmw@in.tum.de

Questions ?