The Challenge of Model-Based Integration for Cyber-Physical Systems

MPM4CPS Conference, Pisa, Italy, 18-23 November 2018

Holger Giese
System Analysis & Modeling Group,
Hasso Plattner Institute, University of Potsdam, Germany
holger.giese@hpi.uni-potsdam.de

Prelude

Internet of Things

Definition of Multi-Paradigm Modeling:

-
-
-
-
-

Ok, then let's talk about what is really interesting for MPM in our CPSLab ...

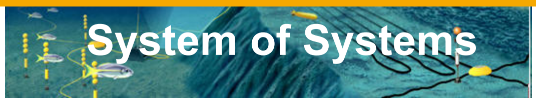
but then I realized ...

(Networked)
Cyber-Physical S

ge-Scale Systems

Smart Home

Smart Factory
E.g. Industry 4.0



E-Health

Smart Logistic

http://oceanservice.noaa.gov/news/weeklynews/nov13/ioos-awards.html

Micro Grids

Ambient
Assisted Living

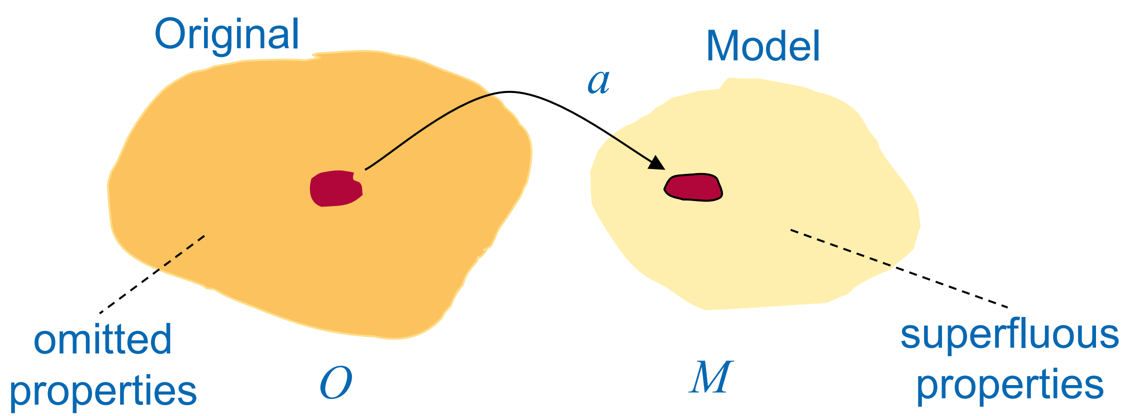
Outline

1. Foundations
(1) What are Models?

Herbert Stachowiak; *Allgemeine Modelltheorie*, Springer-Verlag, Wien 1973.

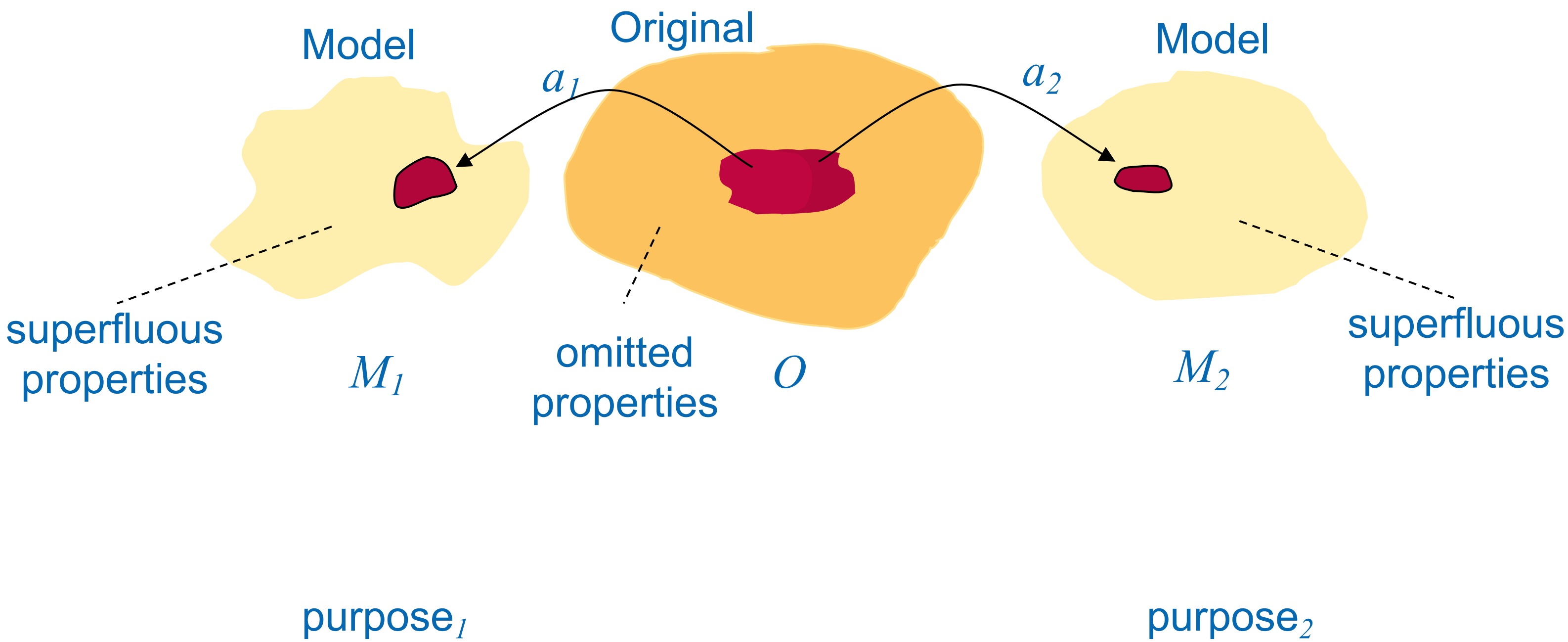
Models are in general abstract representations of existing or envisioned originals

- **Representation** of an original: it exists always a point of reference
  - A function a which assign a model M to the original O (**abstraction**).
  - A not unique backward mapping i assigns originals O to each model M (**interpretation**).
- **Reduction:** not all **properties** are represented
- **Pragmatics:** replaces the original only for a specific **purpose**



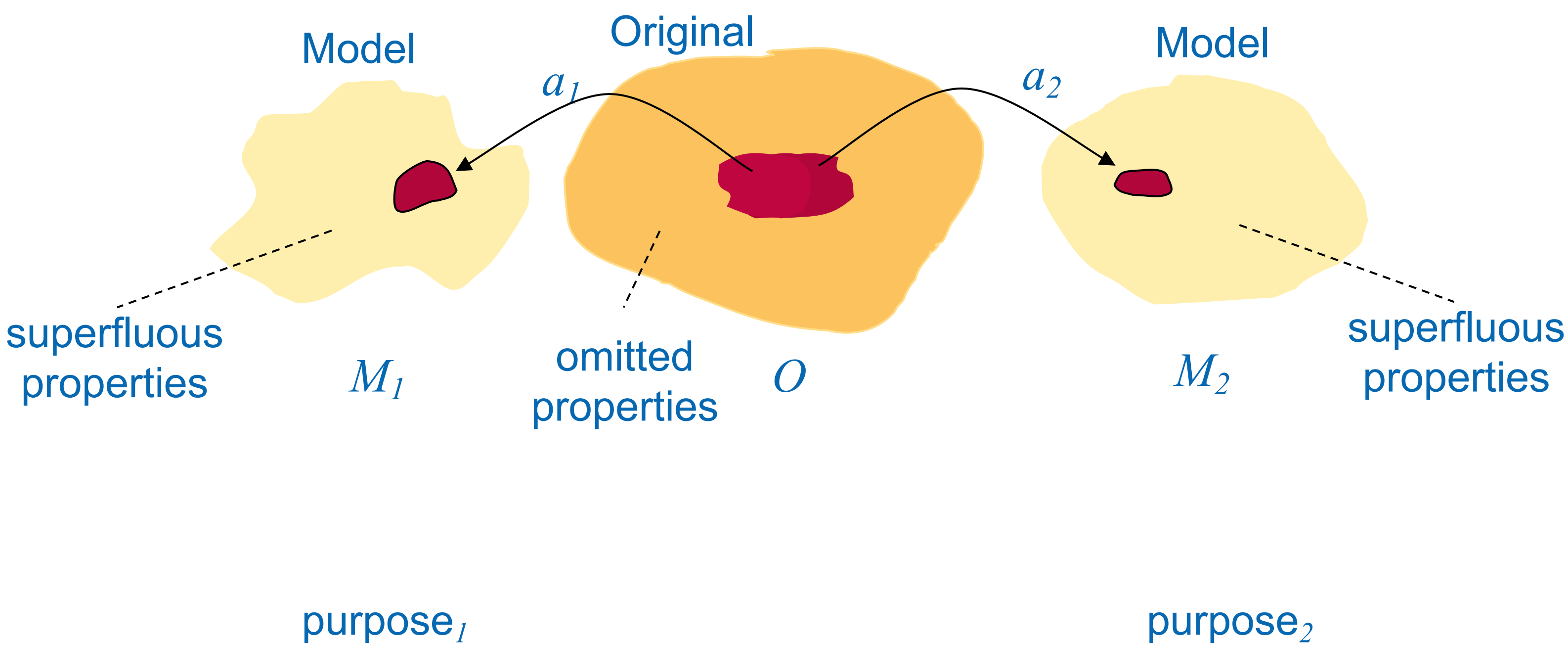
But Nowadays we often have Multiple Models?

Each **model** M_j is an abstract representations of of a part or multiple parts of an existing or envisioned original used for a specific purpose.



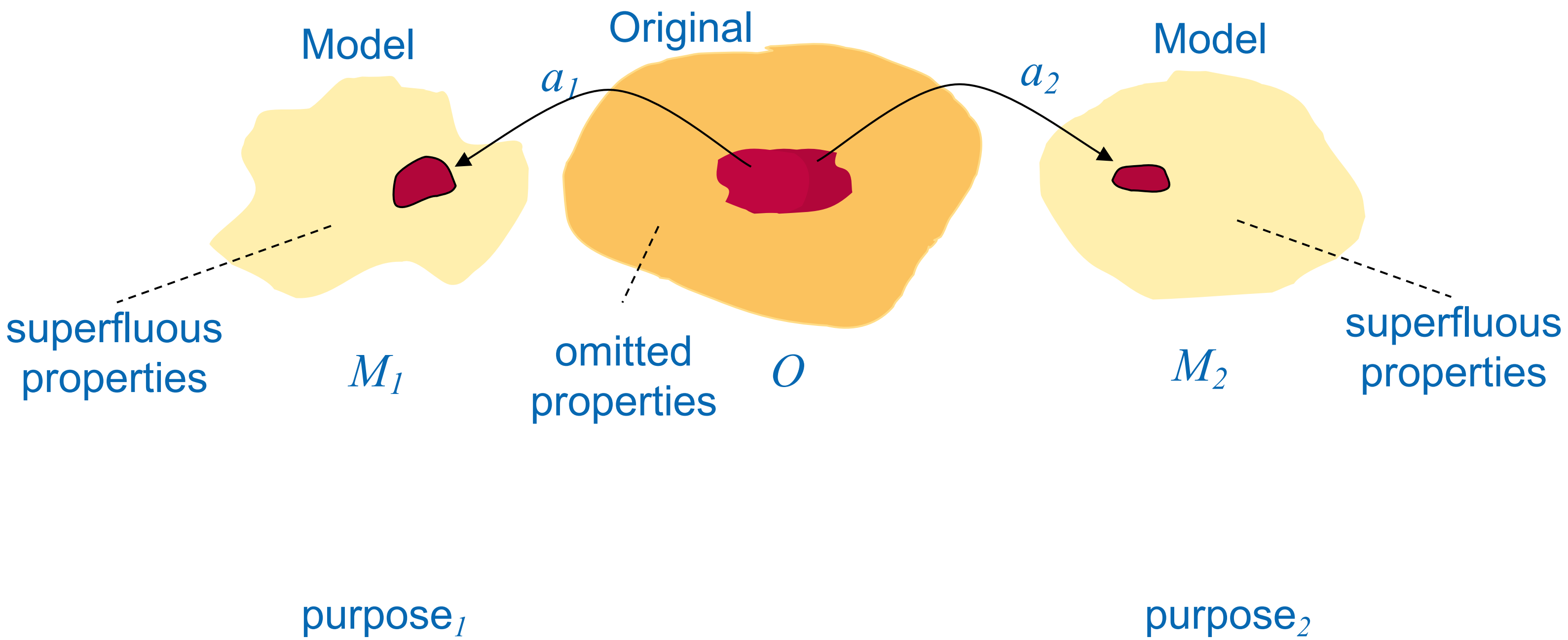
Benefits of Multiple Models?

Benefit: For purpose$_j$ we replace the original O by a suitable model M_j that does not contain any irrelevant information (**reduced complexity**!)



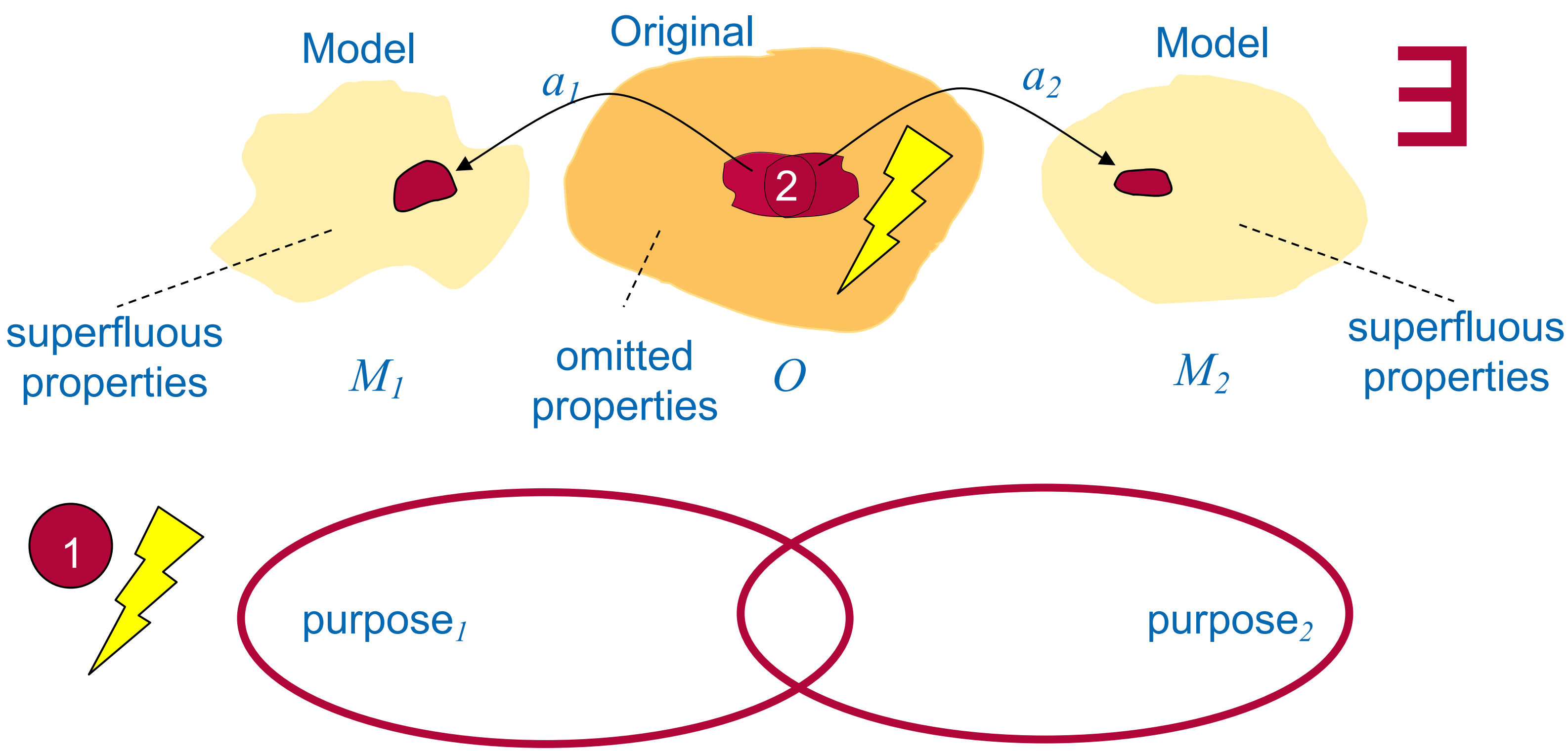
Drawback of Multiple Models?

Drawback: Does an original O consistent with both models M_1 and M_2 really exist (**consistency**)?



How to Handle Multiple Models?

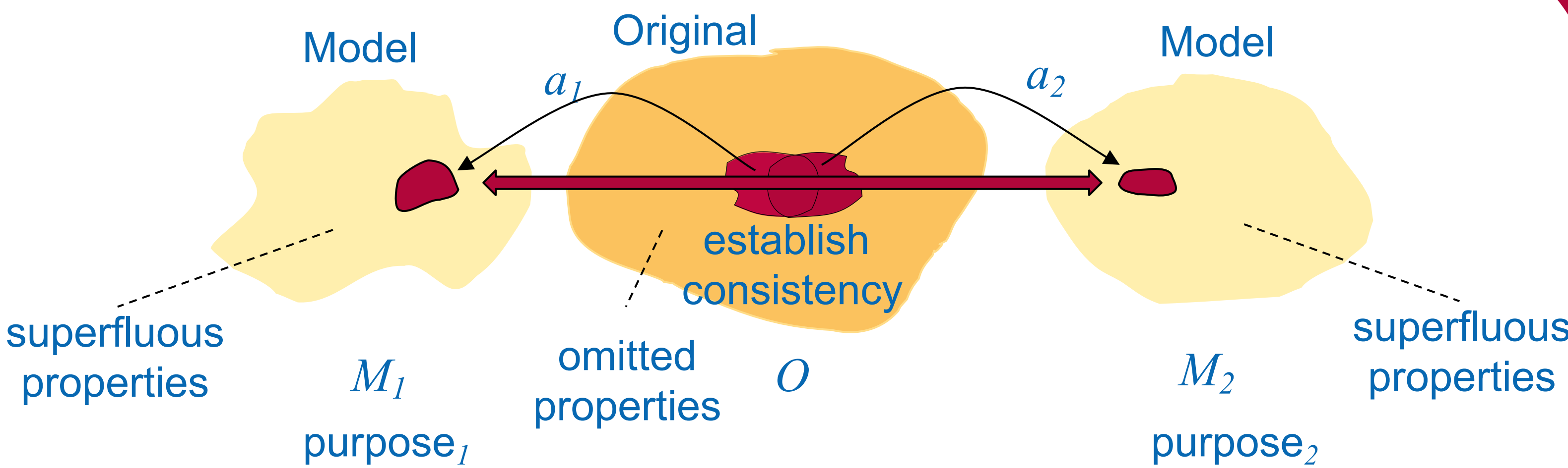
Idea 1: Try for each purposes to find a model M_j that replace the original O, does not contain any irrelevant information (reduced complexity!), and is **completely orthogonal** to all other model.



How to Handle Multiple Models?

Why this focus?
This is the heart of the matter of MPM4CPS!

Idea 2: Try for each purposes to find a model M_j that replace the original O, does not contain any irrelevant information (reduced complexity!), and **integrate** the models systematically to establish consistency.

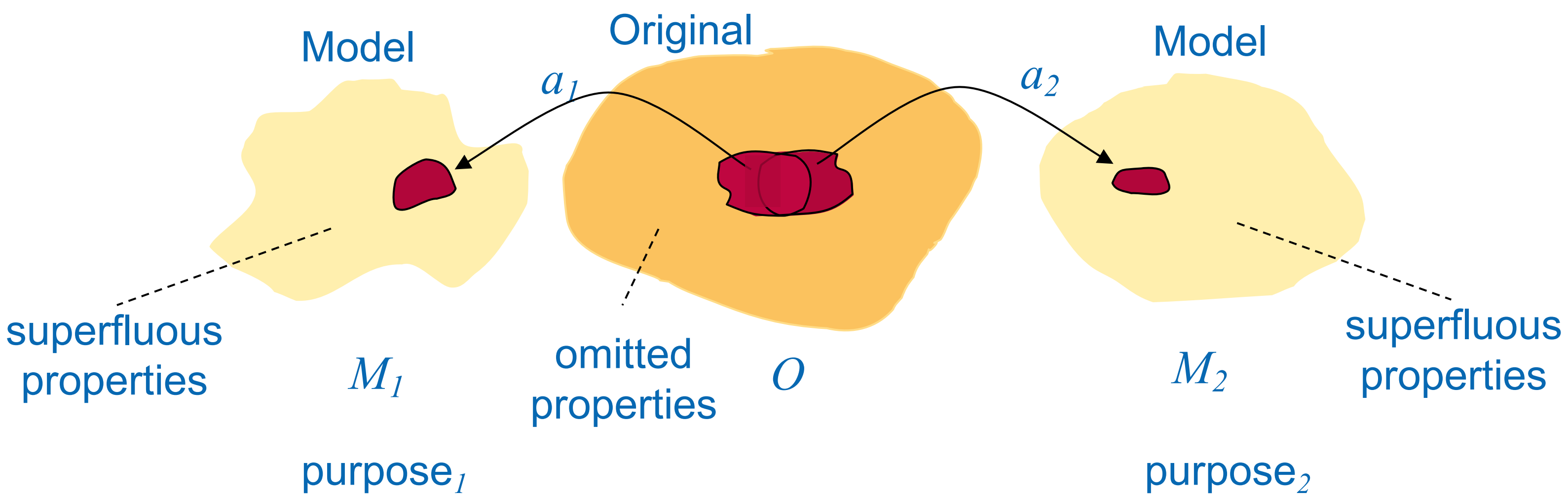


Key questions:

- How many models are helpful (tradeoff benefits vs. integration effort)?
- When and how is integration happen for these models?

(1) How Many Models? Multi-Formalisms

HPI Hasso Plattner Institut



Specific for purpose$_1$:

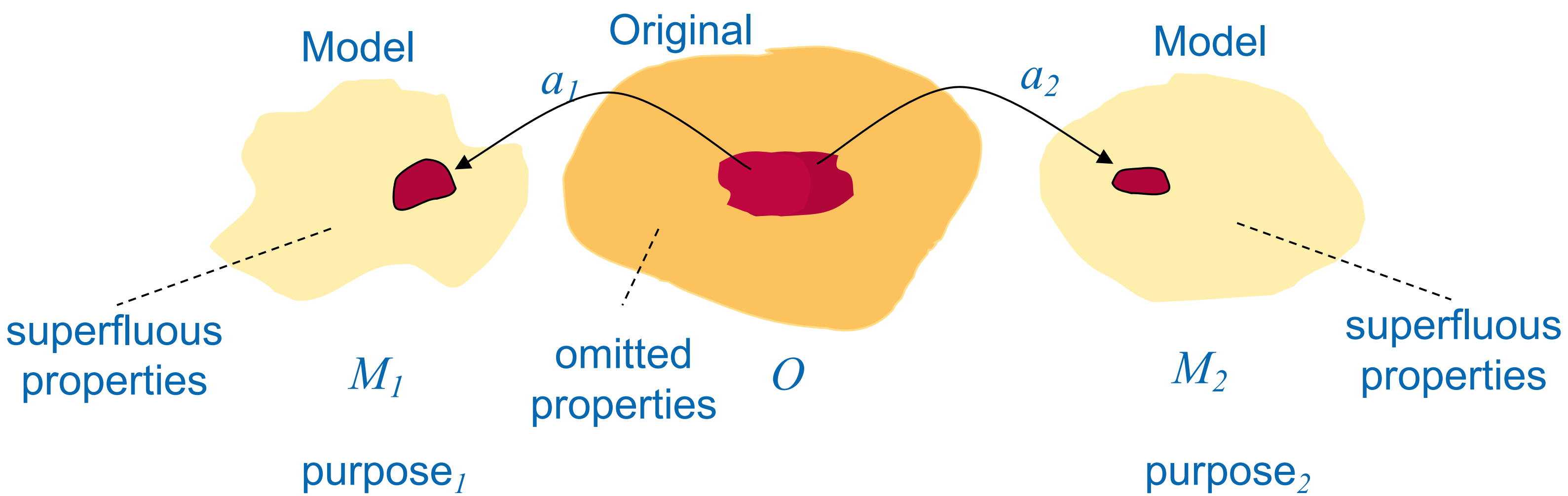
- Chosen formalism (semantics)
- Chosen level of detail

Specific for purpose$_2$:

- Chosen formalism (semantics)
- Chosen level of detail

Integration has to consider more ...

How Many Models? Multiple Paradigms



Specific for purpose$_1$:

- Chosen paradigm
  - Formalism(s) + semantics
  - Workflows and tools used
  - Local consistency needs

Specific for purpose$_2$:

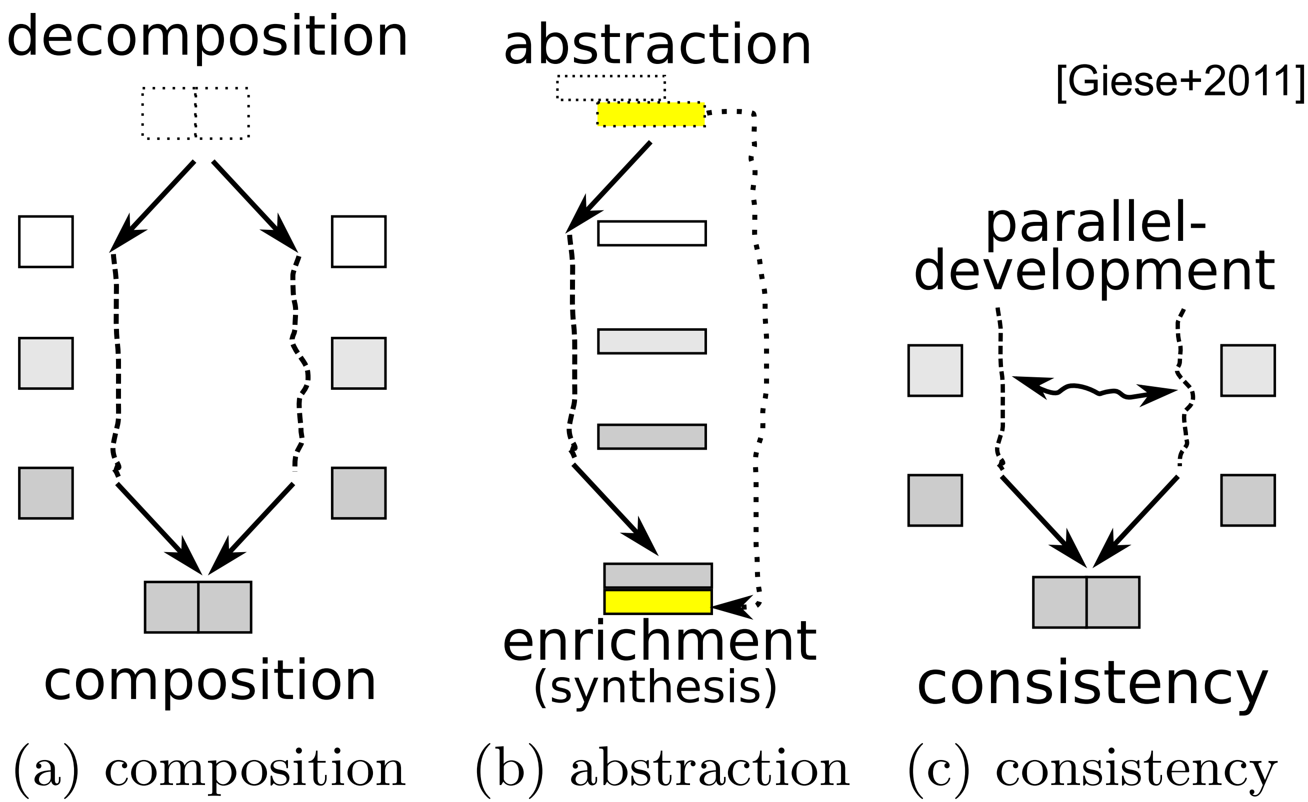
- Chosen paradigm
  - Formalism(s) + semantics
  - Workflows and tools used
  - Local consistency needs

Integration has to bridge/link the paradigms

(2) Integration: When & How

Warning: We use a less restricted notion of integration than many others ...

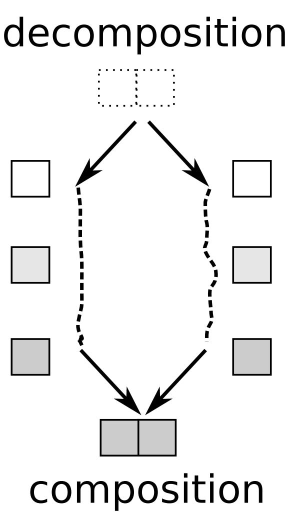
Fundamental Techniques for Integration:



Holger Giese, Stefan Neumann, Oliver Niggemann and Bernhard Schätz. **Model-Based Integration**. In *Model-Based Engineering of Embedded Real-Time Systems - International Dagstuhl Workshop, Dagstuhl Castle, Germany, November 4-9, 2007. Revised Selected Papers*, Vol. 6100:17-54 of *Lecture Notes in Computer Science*, Springer, 2011.

Integration via Explicit De-composition & Composition

[Giese+2011]



(a) composition

- explicit (horizontal) decomposition
- constant level of abstraction
- subsystems can be developed in parallel

Point in time:

- **Decomposition**: interfaces guarantees integration during the later composition (e.g., syntax-level for programming languages)
- **Composition**: risk that integration problems are detected rather late
- **Synthesis**: automated techniques that can generate a solution that solves the integration problem (if possible)

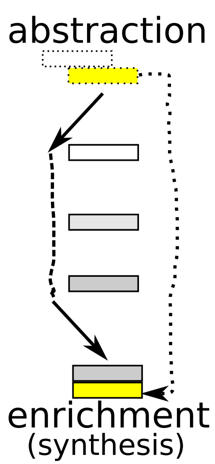
Remarks:

- Ideal case: all relevant characteristics are guaranteed for composition
- Real case: only a few relevant characteristics are guaranteed for composition
- Separation of concerns may not be enough to exclude that concern span multiple models
- Emergent phenomena can only be observed for the composition (e.g., deadlock)

Integration via Vertical Abstraction & Enrichment

[Giese+2011]

- change of the abstraction level
- implicit separation by omitting the details for a certain time
- Vertical enrichment can happen in two fundamentally different forms:
  - unconstrained enrichment (orthogonal characteristics)
  - constrained enrichment (refinement/approximation)



(b) abstraction

Point in time:

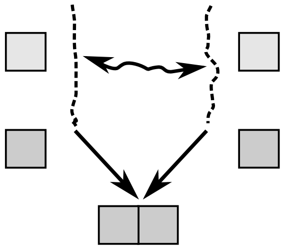
- **During abstraction**: can be employed to ease development when there is only a unidirectional dependency between the upfront-addressed details and the omitted ones
- **During enrichment**: the integration problem has to be addressed late when the enrichment happens, as the initial abstraction step does not provide any guarantee for the later enrichments.
- **During enrichment by synthesis**: used to automatically apply enrichment (if possible)

Examples:

- Architecture layers with std. interface (operating system, hardware)

Integration via Consistency & Synchronization

[Giese+2011]



(c) consistency

- approach the dependencies between the different artifacts throughout the parallel development
  - check consistency & resolve issue immediately
  - synchronization ➜ automatically keep consistent

Point in time:

- Frequently: do a horizontal integration of models that evolve in parallel

Remarks:

- the in parallel developed models can more freely evolve
- consistency resp. synchronization covers usually not all integration problems later on (example co-simulation and scheduling)

Kind of Integration (to Bridge Paradigms)

- **Formalism-based**: Having a single formalism in a paradigm that includes multiple paradigms (e.g., hybrid automata contain differential equations and automata)
- **Composition-based**: We compose formalism supporting different paradigms into a single paradigm by a suitable model of computation that composes the multiple formalisms (e.g., Simulink/Stateflow)
- **Tool-based**: We consider formalisms supporting different paradigms together via tools (e.g., co-simulation of a Simulink model and a plant specific simulator)

Level of Integration

- **Representation-level**: integration efforts only guarantee that a joint representation is reached
- **Syntax-level**: integration efforts lead to correct syntax
- **Semantics-level**: integration efforts lead to compatibility at the level of the semantics

Examples from software engineering:

- Merge is usually only ad hoc achieving representation-level integration and compilation is expected to ensure syntax-level integration
- Continuous integration = fully automated regression testing ensures some degree of semantic-level integration (changes do not break the semantic needs encoded in the tests)

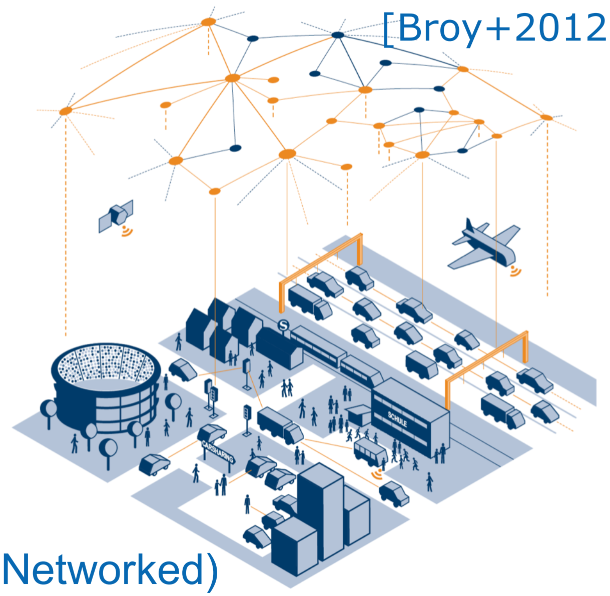
Outline

1. Foundations
2. **Cyber-Physical Systems**
3. HPI CPSLab & Integration
4. Future Needs for Integration
5. Conclusion & Outlook

3. Cyber-Physical Systems & Integration

Hasso Plattner Institut

[Broy+2012]

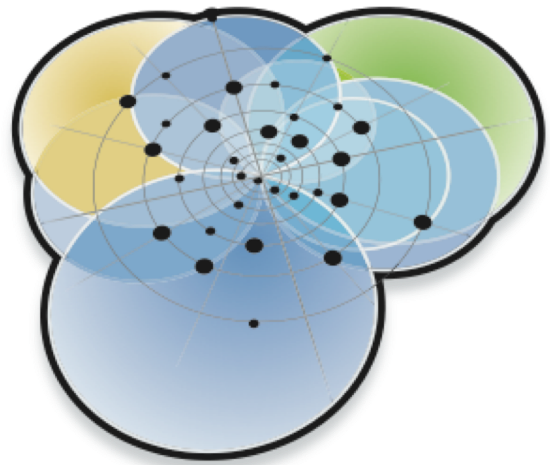


(Networked)
Cyber-Physical Systems

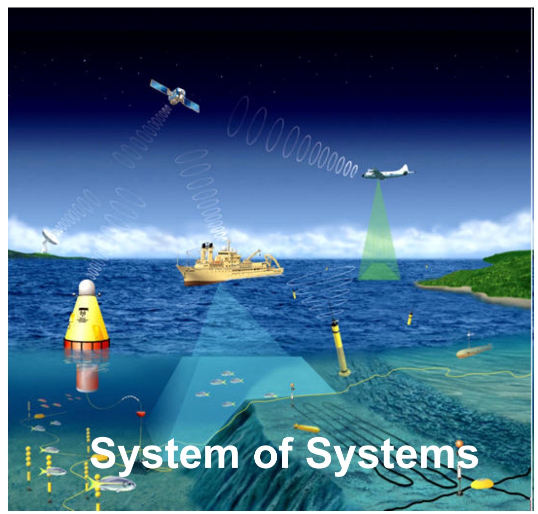
Internet of Things

Smart City

[Northrop+2006]



Ultra-Large-Scale Systems



http://oceanservice.noaa.gov/news/weeklynews/nov13/ioos-awards.html

Smart Home

E-Health

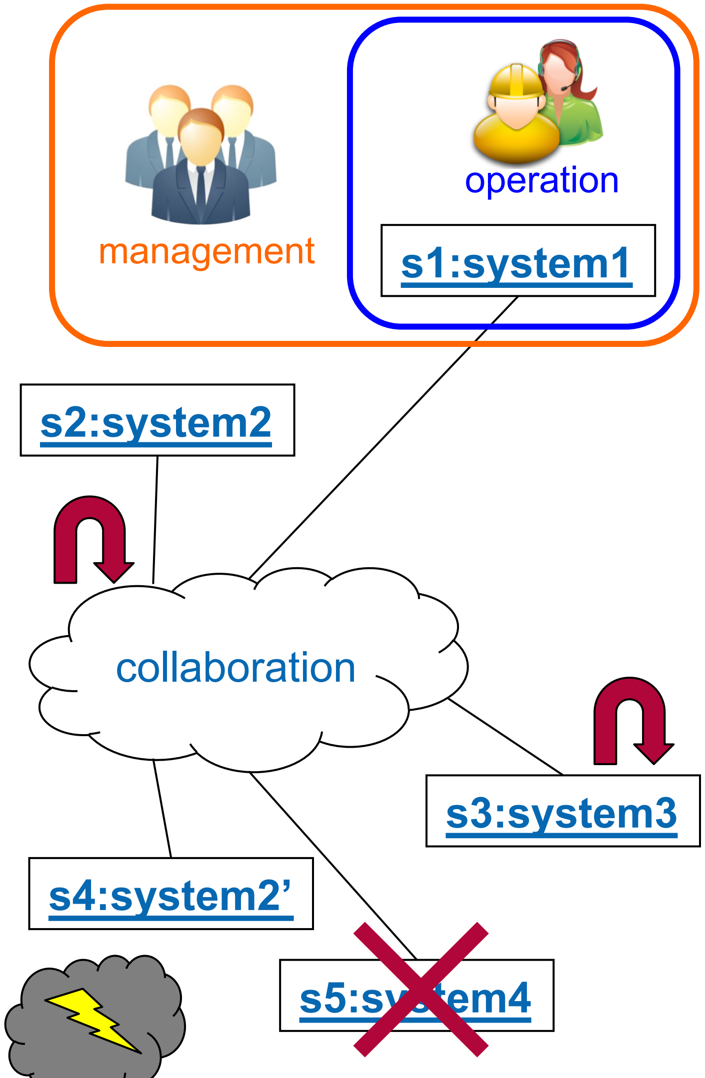
Smart Factory -
E.g. Industry 4.0

Smart Logistic

Micro Grids

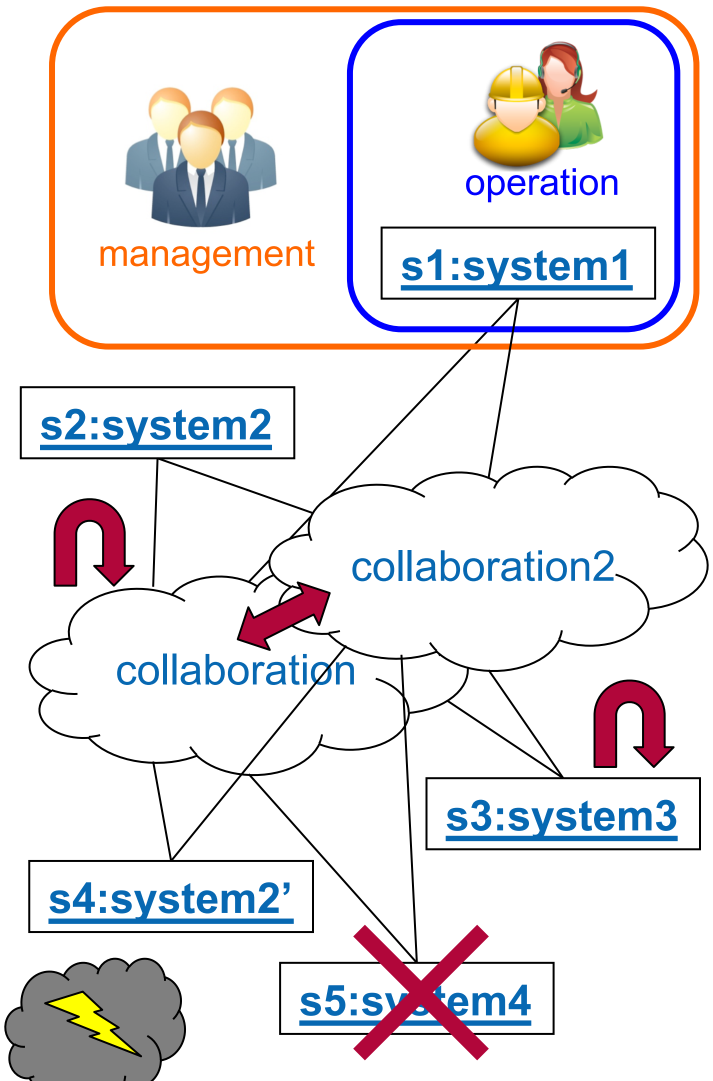
Ambient
Assisted Living

A Selection of Critical Future Challenges



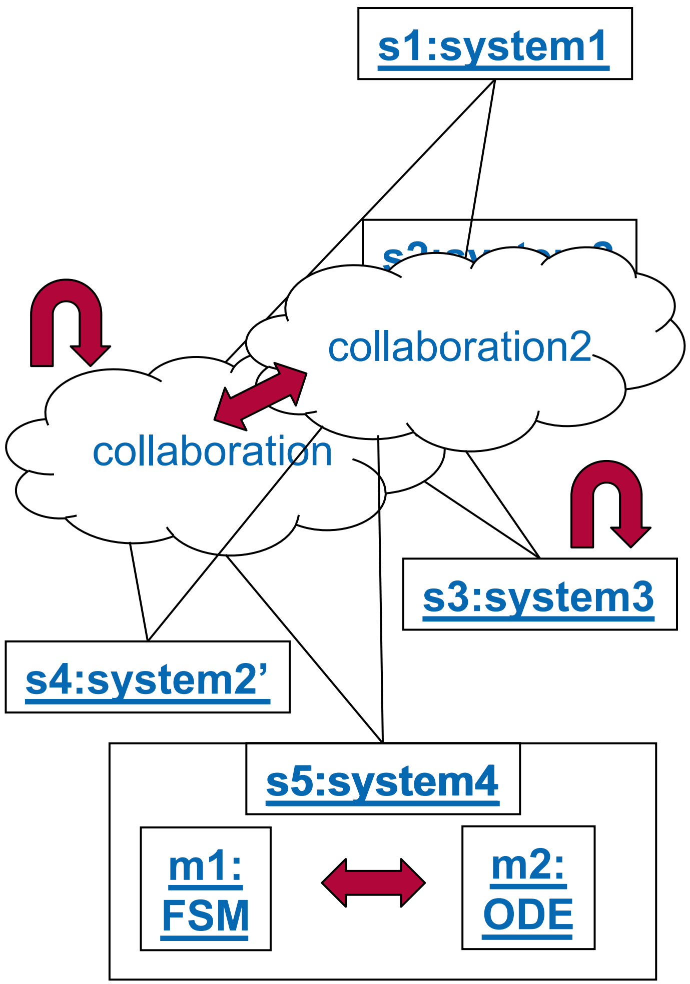
- **Operational** and **managerial independence**
  - operated independent from each other without global coordination
  - no centralized management decisions (possibly confliction decisions)
- **Dynamic architecture** and **openness**
  - must be able to dynamically adapt/absorb structural deviations
  - subsystems may join or leave over time in a not pre-planned manner
- **Advanced adaptation**
- **Resilience**

A Selection of Critical Future Challenges



- **Operational** and **managerial independence**
  - operated independent from each other without global coordination
  - no centralized management decisions (possibly confliction decisions)
- **Dynamic architecture** and **openness**
  - must be able to dynamically adapt/absorb structural deviations
  - subsystems may join or leave over time in a not pre-planned manner
- **Advanced adaptation**
- **Resilience**
- **Cross-Domain Integration**

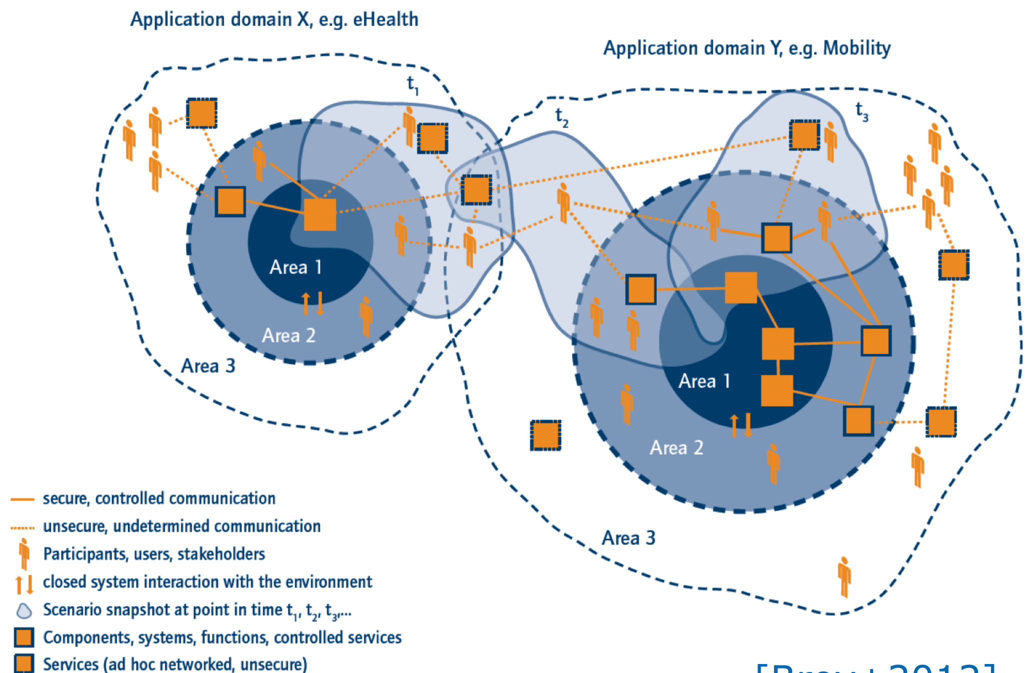
A Selection of Critical Future Challenges



- **Operational** and **managerial independence**
  - operated independent from each other without global coordination
  - no centralized management decisions (possibly confliction decisions)
- **Dynamic architecture** and **openness**
  - must be able to dynamically adapt/absorb structural deviations
  - subsystems may join or leave over time in a not pre-planned manner
- **Advanced adaptation**
- **Resilience**
- **Cross-Domain Integration**
- **Integrate Models of Computation**

Challenge: Cross-Domain Integration

Example: A convoy of fully autonomous cars abandons the premium track in order to give way to an ambulance (intersection of CPS specific for **traffic** and **health care**)



[Broy+2012]

CPS of different domains have to be connected:

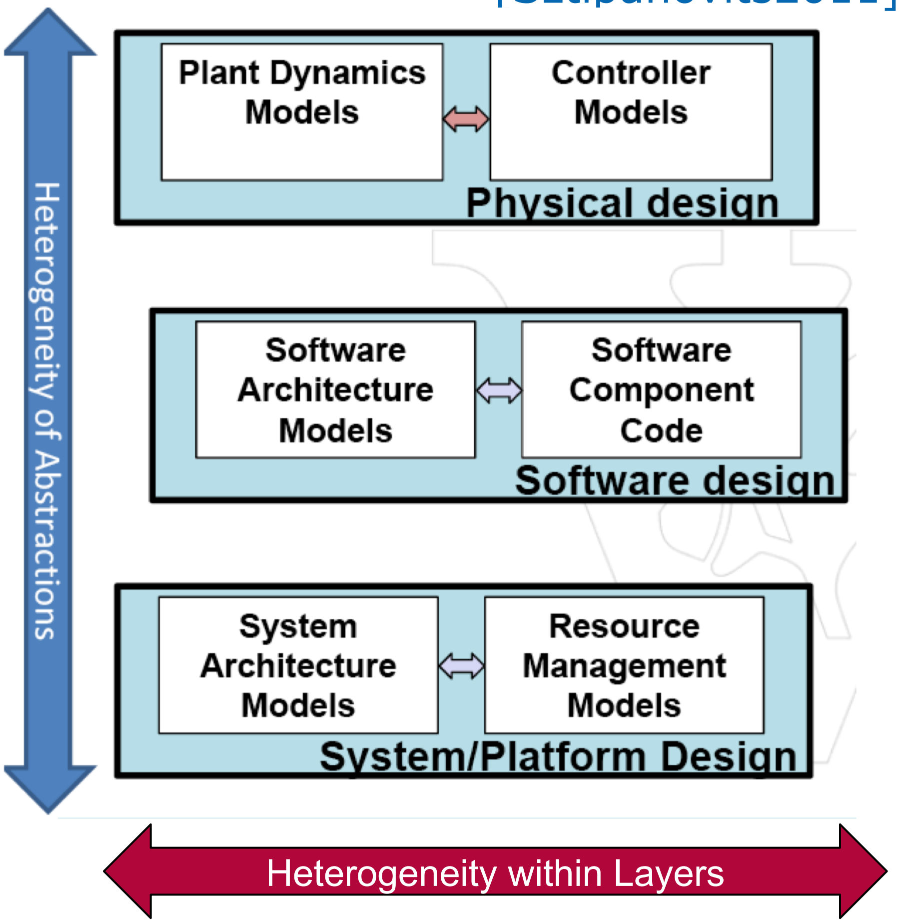
- According to social and spatial network topologies, CPS operate across different nested spheres of uncertainty
- CPS dedicated to different domains have to to interact and coordinate.

Integration has to cover multiple domains and their **paradigms**

Challenge: Integrate Models of Computation

- Problem to integrate models within one layer as different **models of computation** are employed
- Leaky abstractions are caused by lack of composability across system layers. Consequences:
  - intractable interactions
  - unpredictable system level behavior
  - full-system verification does not scale

[Sztipanovits2011]

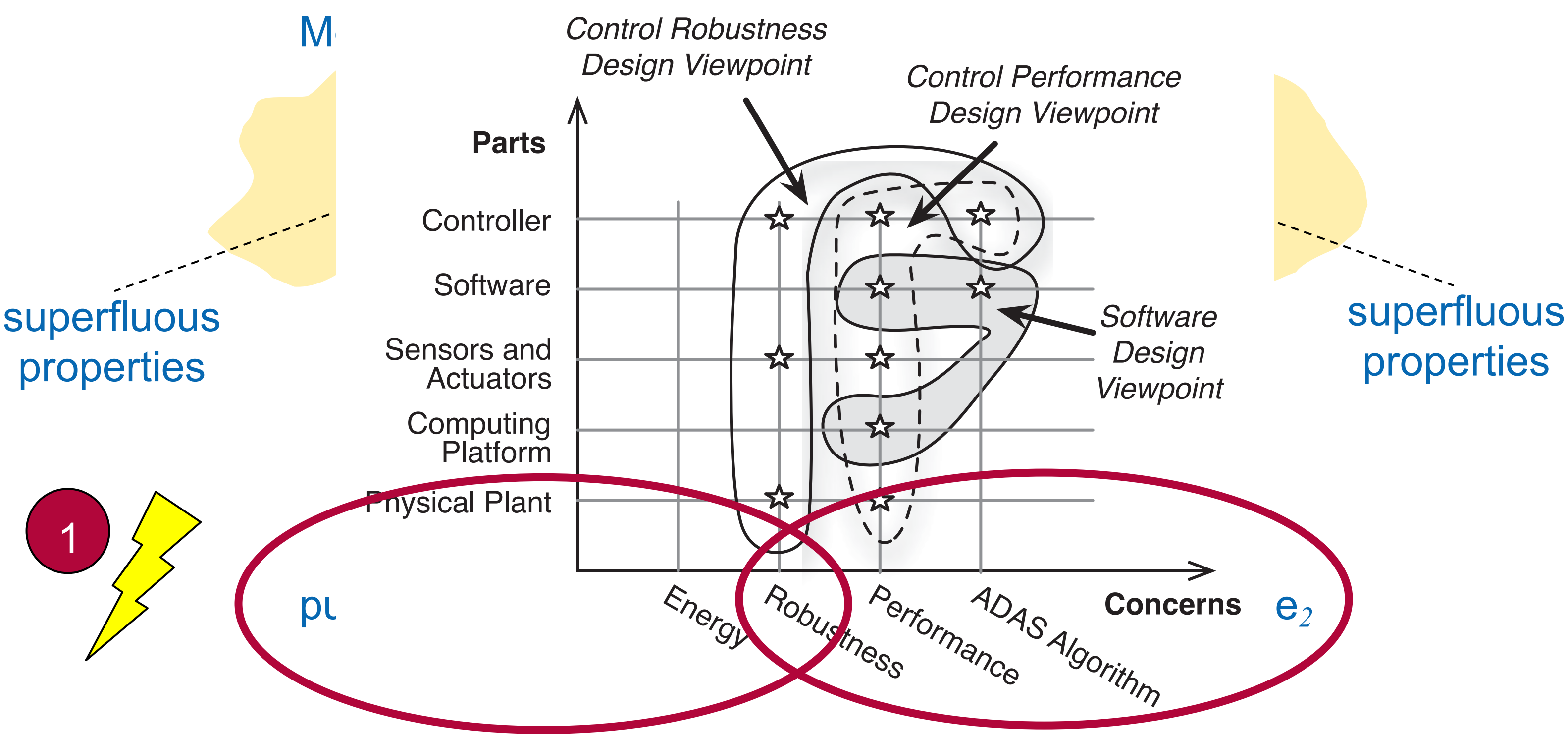


Integration has to cover multiple layers and their paradigms

How to Handle Multiple Models?

Check: Is integration unavoidable for complex CPS?

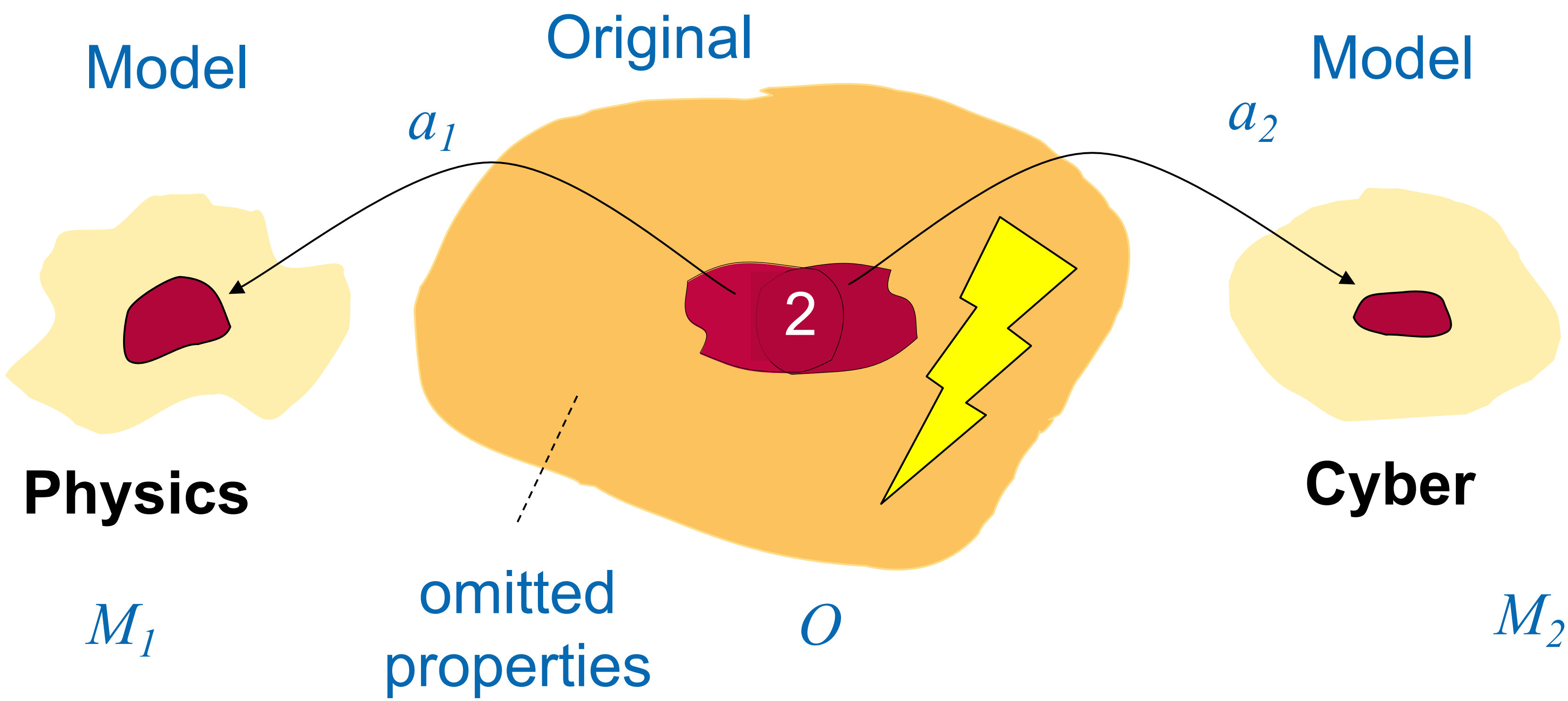
Idea 1: Try for each purposes to find a model M_j that replace the ... does not contain any irrelevant information (reduced complexity!), and is **completely orthogonal** to all other model.



[Broman+2012]

How to Handle Multiple Models?

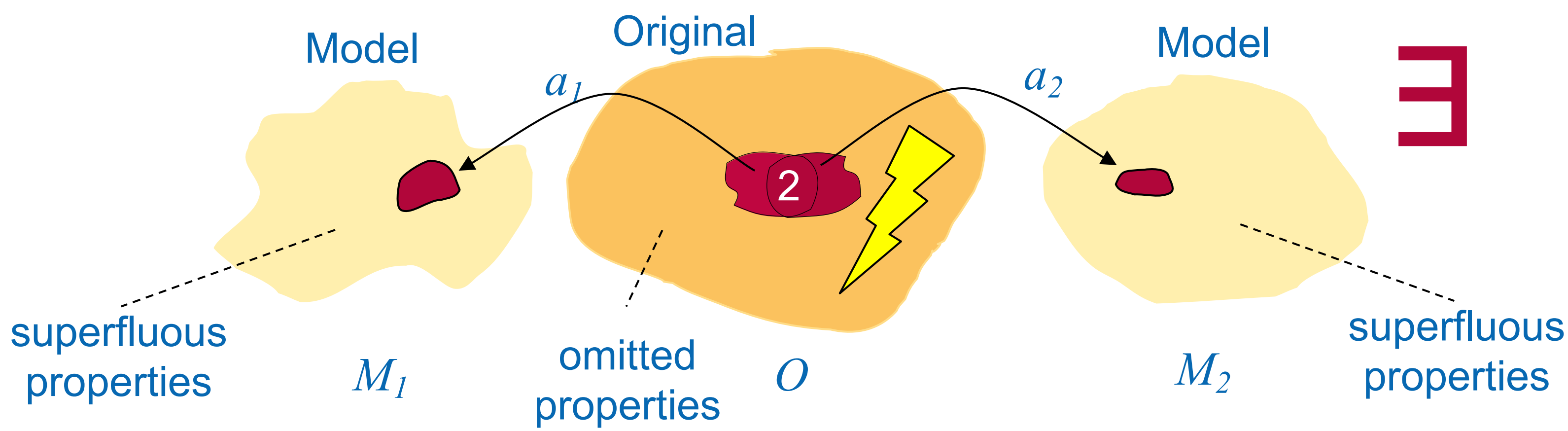
Idea 1: Try for each purposes to find a model M_j that replace the original O, does not contain any irrelevant information (reduced complexity!), and is **completely orthogonal** to all other model.

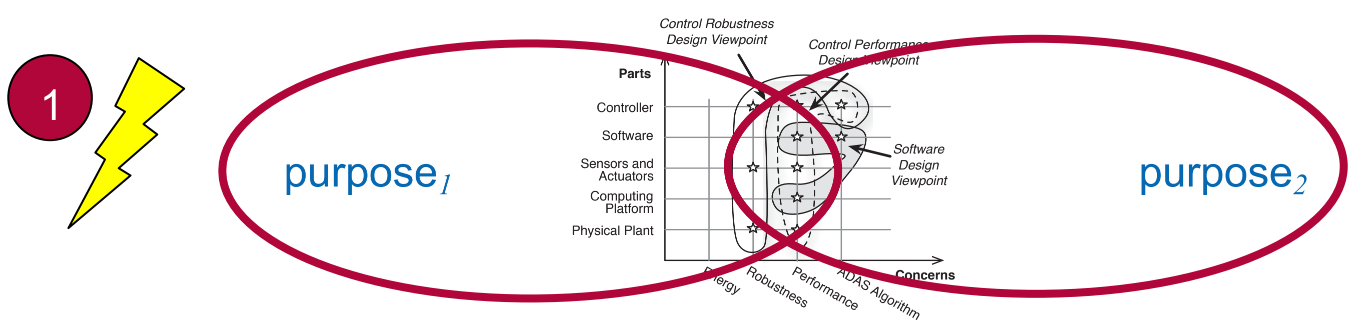


How to Handle Multiple Models?

Conclusion: Integration seems unavoidable for complex CPS!

Idea 1: Try for each purposes to find a model M_j that replace the original, does not contain any irrelevant information (reduced complexity!), and is **completely orthogonal** to all other model.





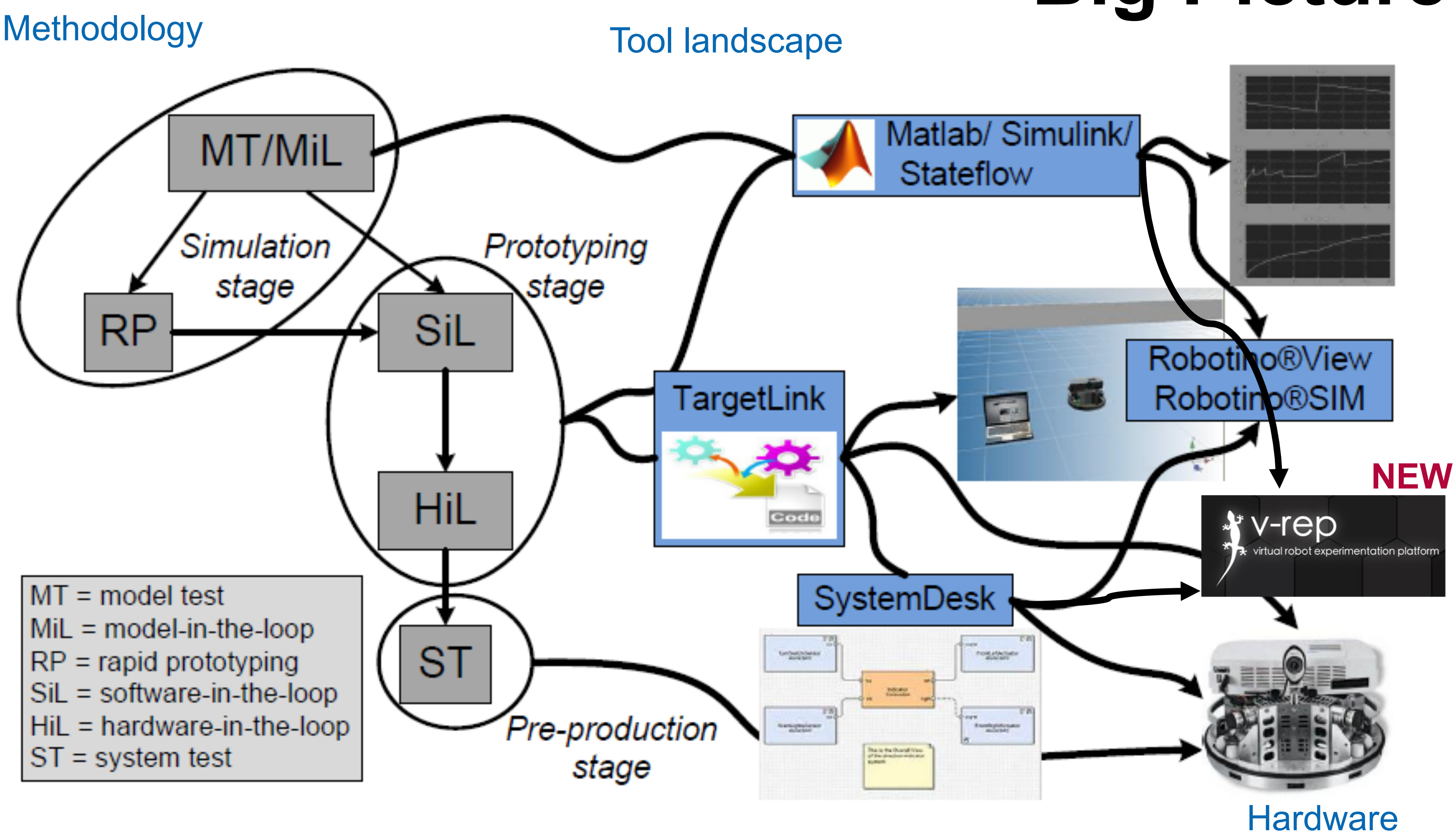
[Broman+2012]

Outline

1. Foundations
2. Cyber-Physical Systems
3. **HPI CPSLab & Integration**
4. Future Needs for Integration
5. Conclusion & Outlook

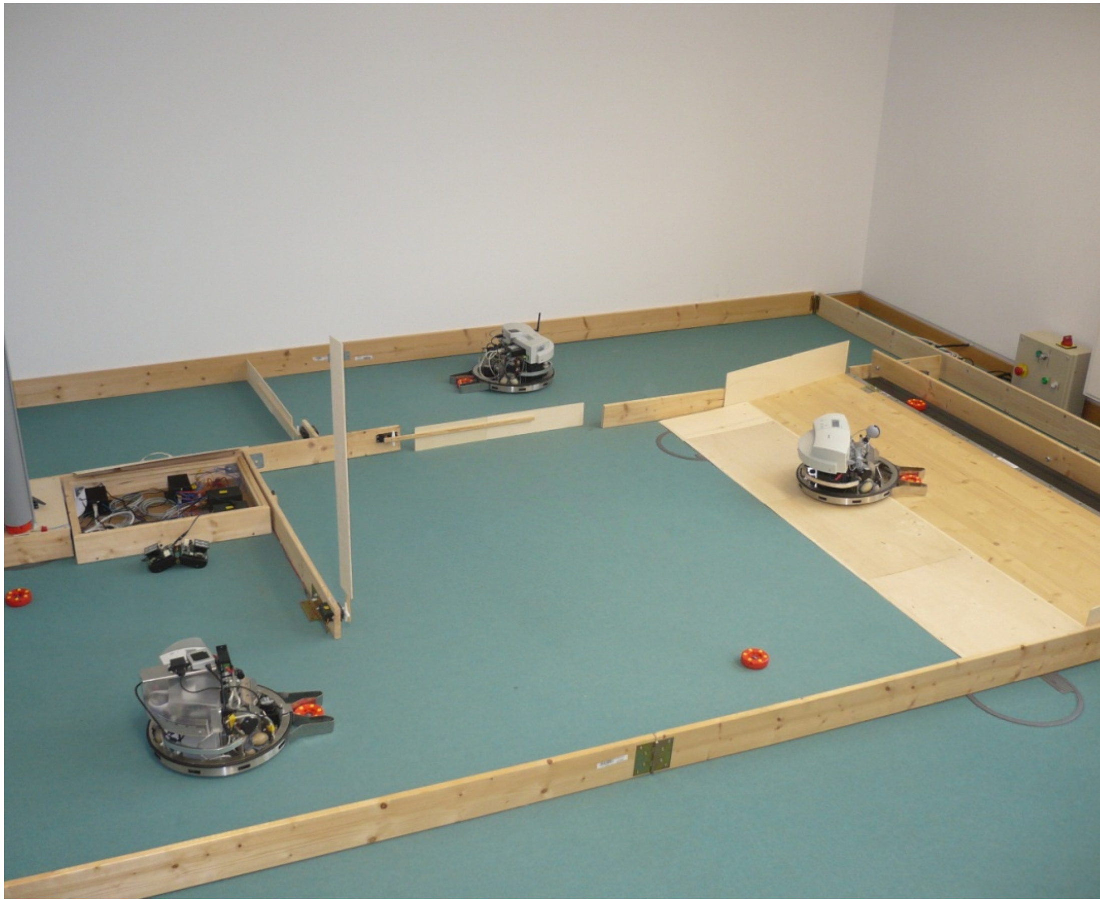
3. HPI CPSLab & Integration:

Big Picture



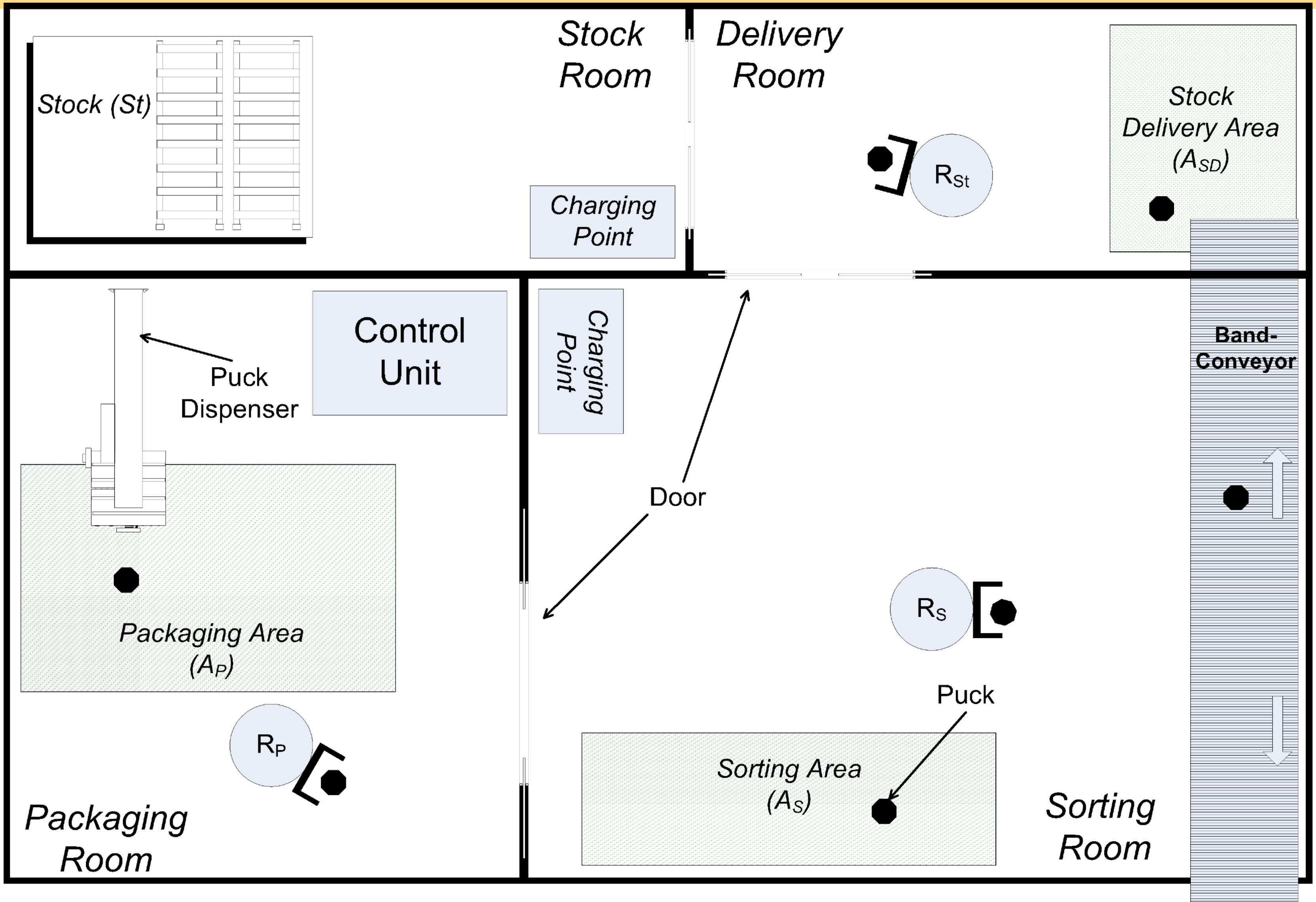
HPI CPSLab: Industry 4.0 Production

- Robots in Production Setting
- Transportation of Goods
  - represented by Pucks
- Different Production Locations
  - Puck Dispenser
  - Conveyor Belt
  - "Rooms"
- Obstacle Avoidance
  - Walls
  - Doors
  - Other Robots



HPI CPSLab: Industry 4.0 Production

Hasso Plattner Institut



HPI CPSLab: Robotino Robot - Overview

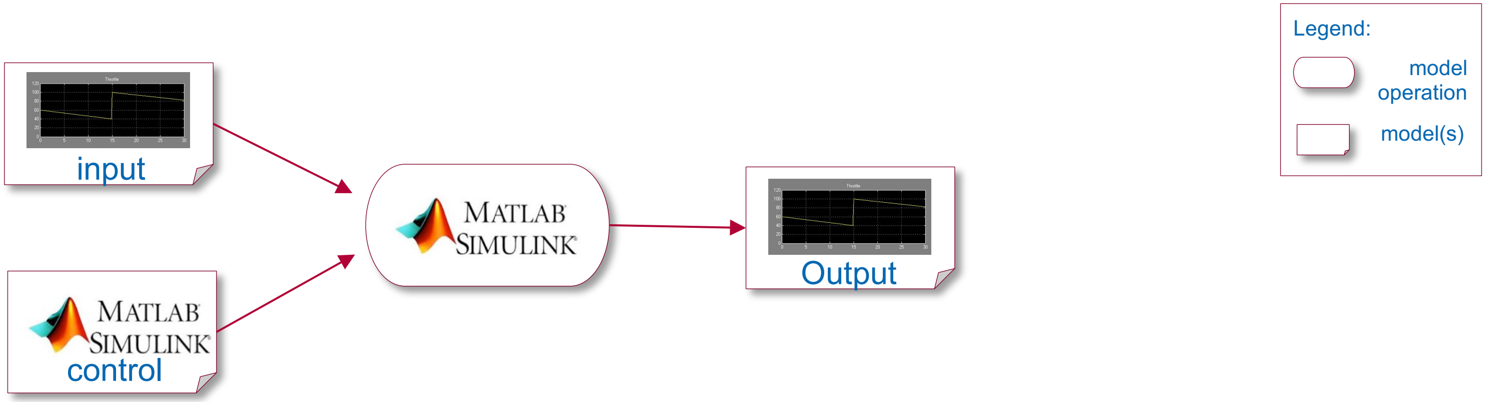
Basic Robotino Robot:

- Omni directional drive permits to move in all directions
- Distance / obstacles sensors
- Bumper to detect collisions
- Coordination via W-LAN

Extensions:

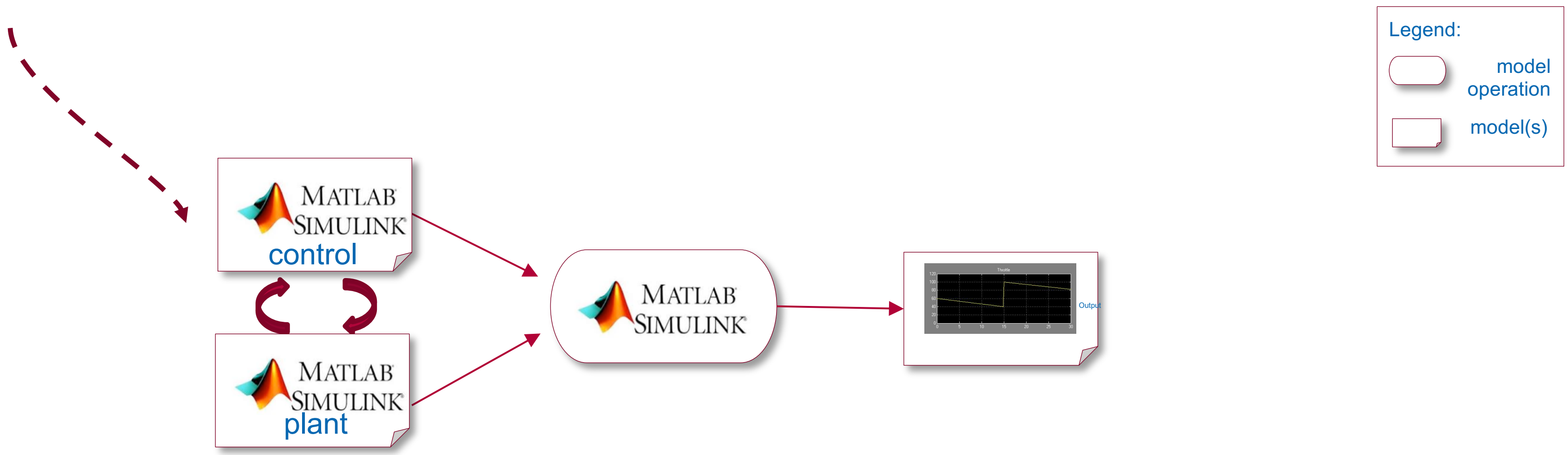
- GPS-like system: Northstar
- Camera & Vision
- Metal detector
- Gripper
- …

Model Test (MT)



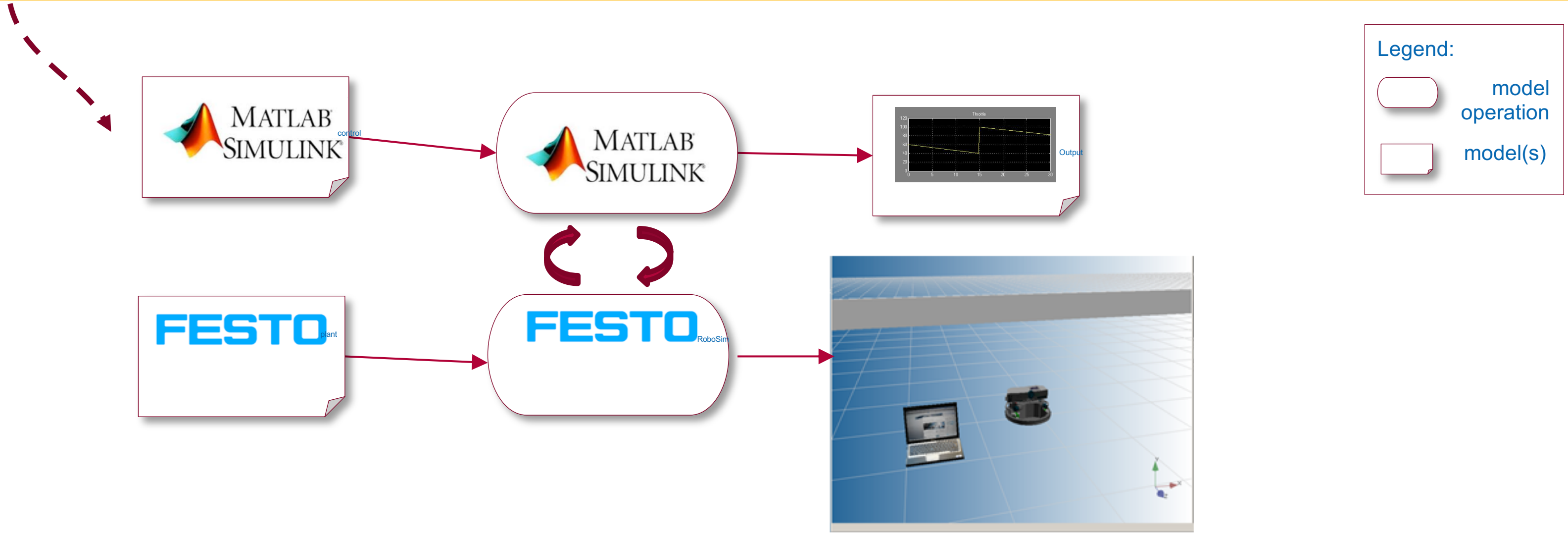
- Layer: Abstract Control Algorithm
- Domains: Control/Software (+ Physics)
- Multi-Paradigm: Yes, if control is discrete and input continuous
- Cyber-Physical system: Yes, as control is cyber and input is (conceptually) from the physical world
- Integration: Decomposition and composition-based

Model in the Loop (MiL)



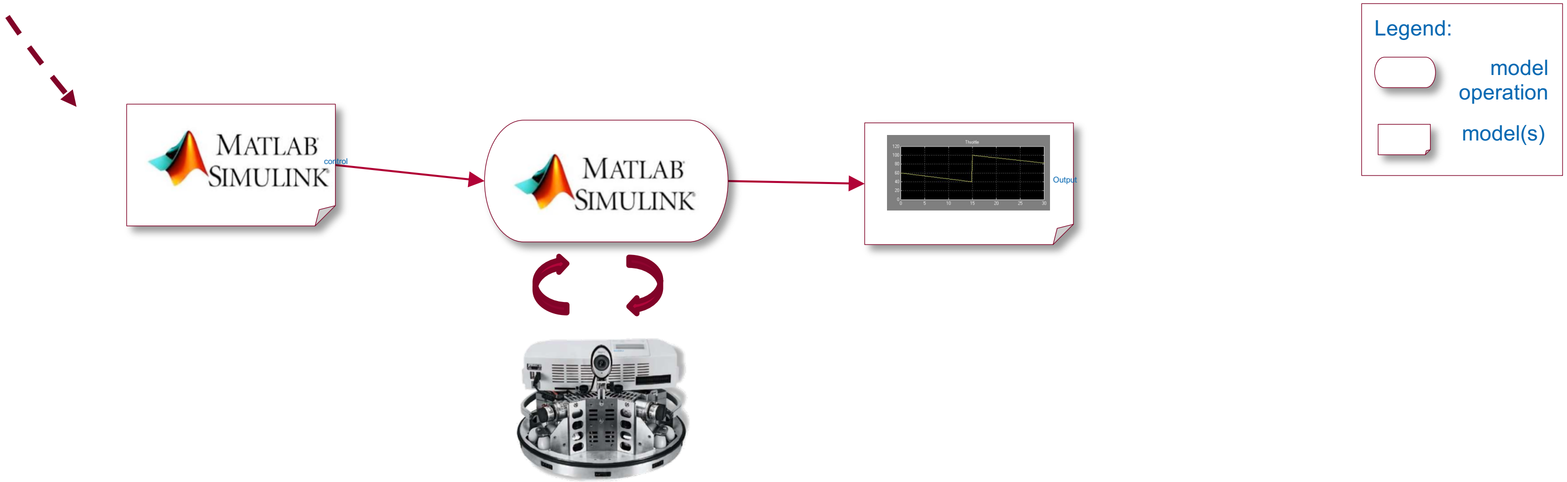
- Layer: Abstract Control Algorithm + Idealized Plant
- Domain: Control/Software + Physics
- Multi-Paradigm: Yes, if control is discrete
- Cyber-Physical system: Yes, as control is cyber world and plant is from the physical world
- Integration: Decomposition & Composition compostion-based

Rapid Prototyping (RP) vs. Robot Simulator



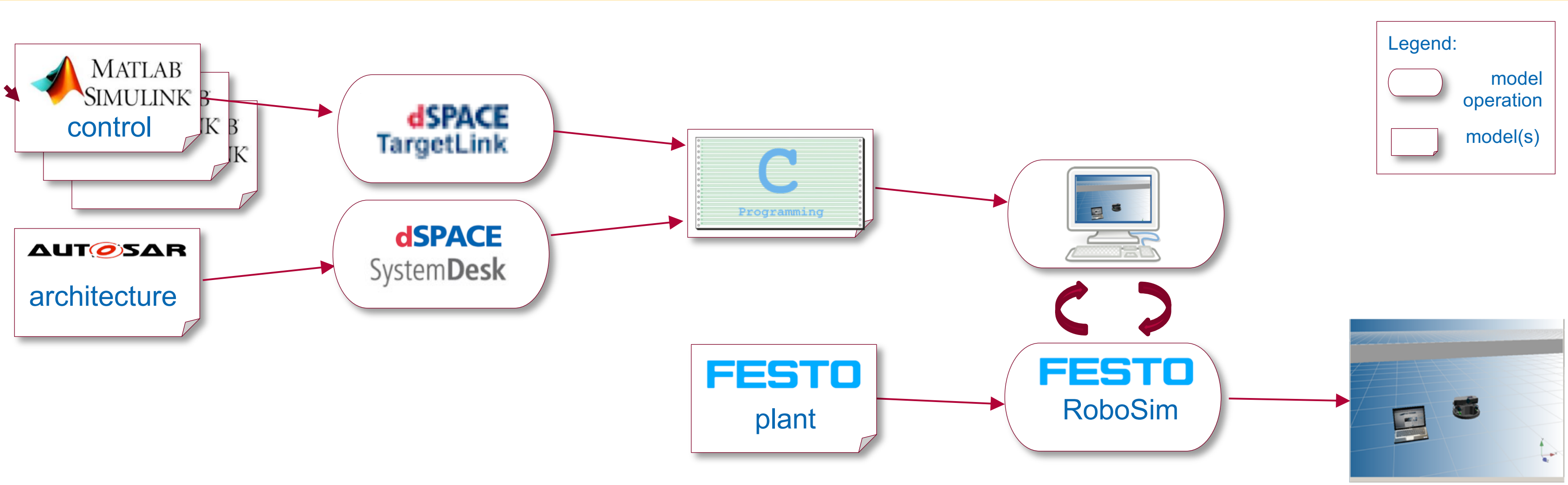
- Layer: Abstract Control Algorithm + Realistic Plant
- Domain: Control/Software + Physics
- Multi-Paradigm: Yes, if control is discrete
- Cyber-Physical system: Yes, as control is cyber world and plant is from the physical world
- Integration: Consistency via co-simulation (tool-based)

Rapid Prototyping (RP) vs. Robot



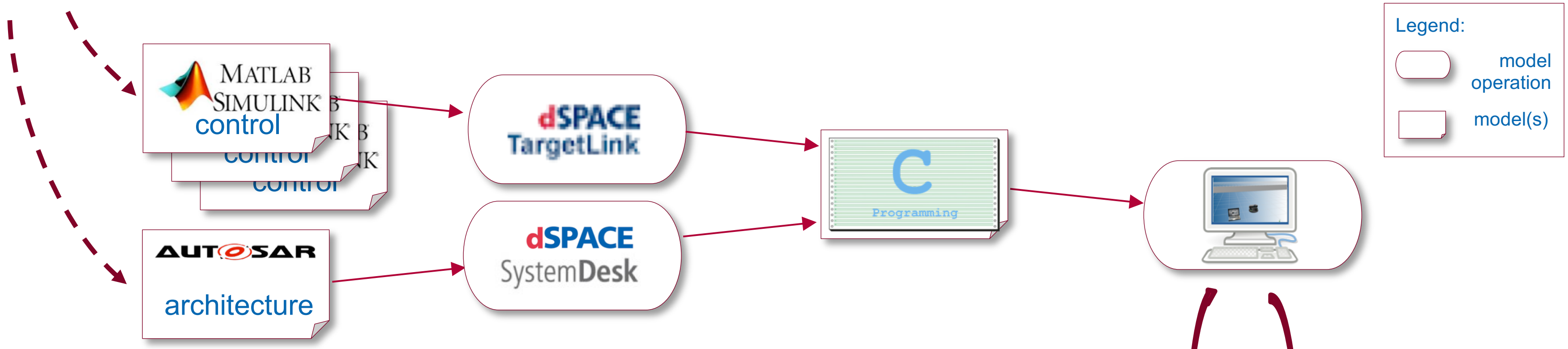
- Layer: Abstract Control Algorithm + Real Plant
- Domain: Control/Software + Real Physics
- Multi-Paradigm: Yes, if control is discrete
- Cyber-Physical system: Yes, as control is cyber world and plant is from the physical world
- Integration: Consistency via rapid protoyping (tool-based)

Software in the Loop (SiL) vs. Desktop + Sim

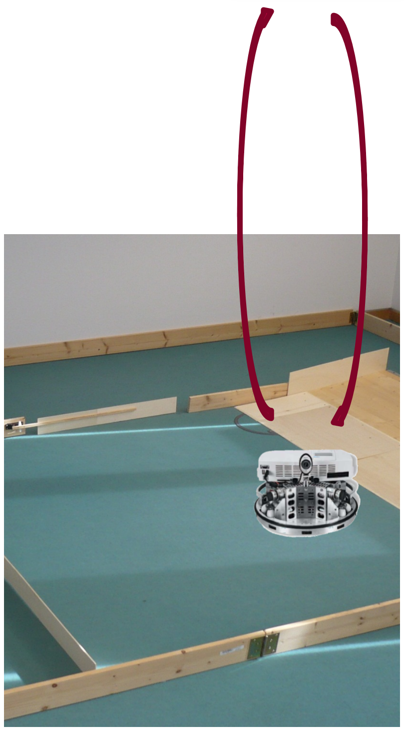


- Layer: Control Software + Architecture + Realistic Plant
- Domain: Control/Software + Scheduling + Realistic Physics
- Multi-Paradigm: 1) Yes, if control is discrete 2) Combine architecture and control
- Cyber-Physical system: Yes, as control is cyber world and plant is from the physical world (control and architecture are both cyber)
- Integration for 1): Consistency via co-simulation (tool-based)
- Integration for 2): Decomposition and synthesis composition-based

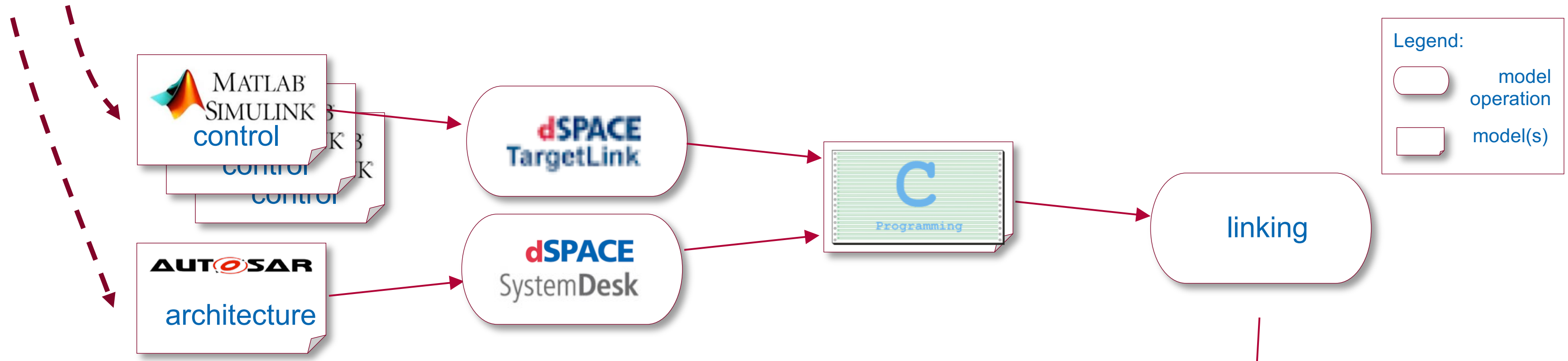
Software in the Loop (SiL) vs. Desktop + Robot



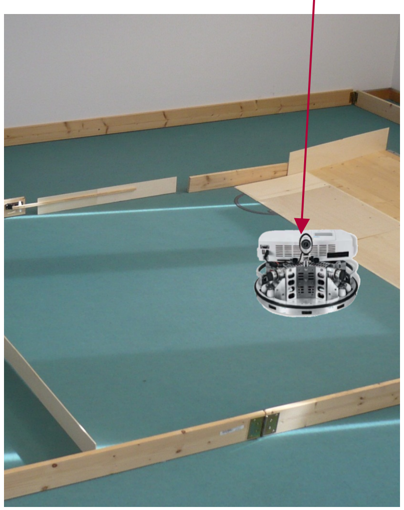
- Layer: Control Software + Architecture + idealized Hardware + Real Plant
- Domain: Control/Software +Architecture + Scheduling + WLAN + Real Physics
- Multi-Paradigm: 1) Yes, if control is discrete 2) Combine architecture and control
- Cyber-Physical system: Yes, as control is cyber world and plant is from the physical world (control and architecture are both cyber)
- Integration for 1): Consistency via rapid-protoyping (tool-based) via WLAN
- Integration for 2): Decomposition and synthesis composition-based

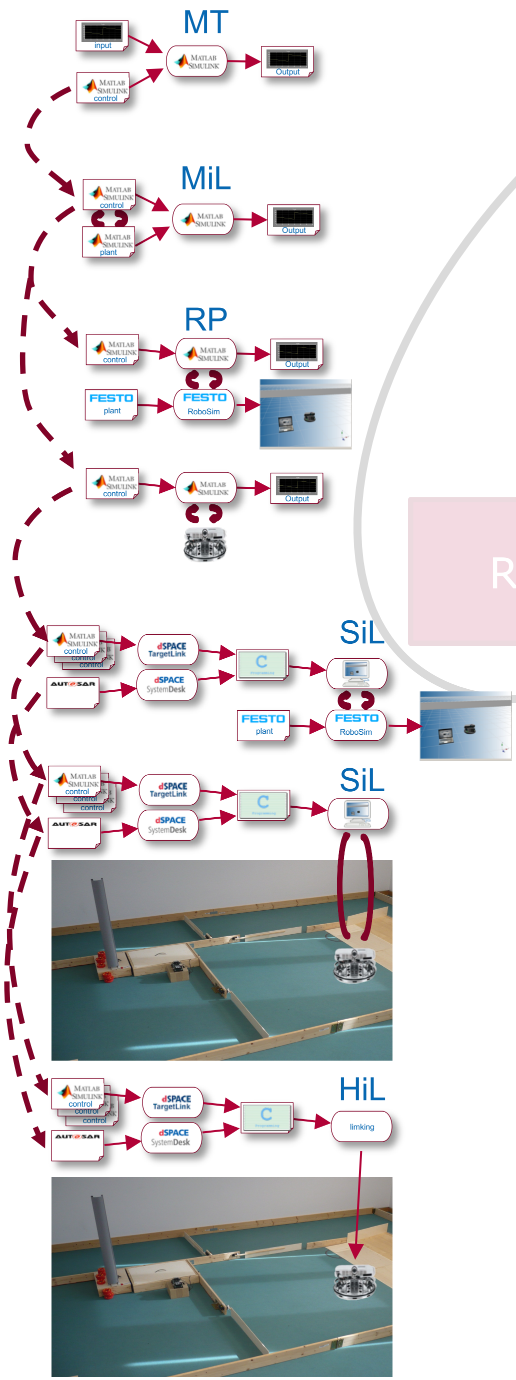


Hardware in the Loop (HiL)



- Layer: Control Software + Architecture + Real Hardware + Real Plant
- Domain: Control/Software +Architecture + Scheduling + Real Physics
- Multi-Paradigm: 1) Yes, if control is discrete 2) Combine architecture and control
- Cyber-Physical system: Yes, as control is cyber world and plant is from the physical world (control and architecture are both cyber)
- Integration for 1): Consistency via execution (tool-based)
- Integration for 2): Decomposition and synthesis composition-based



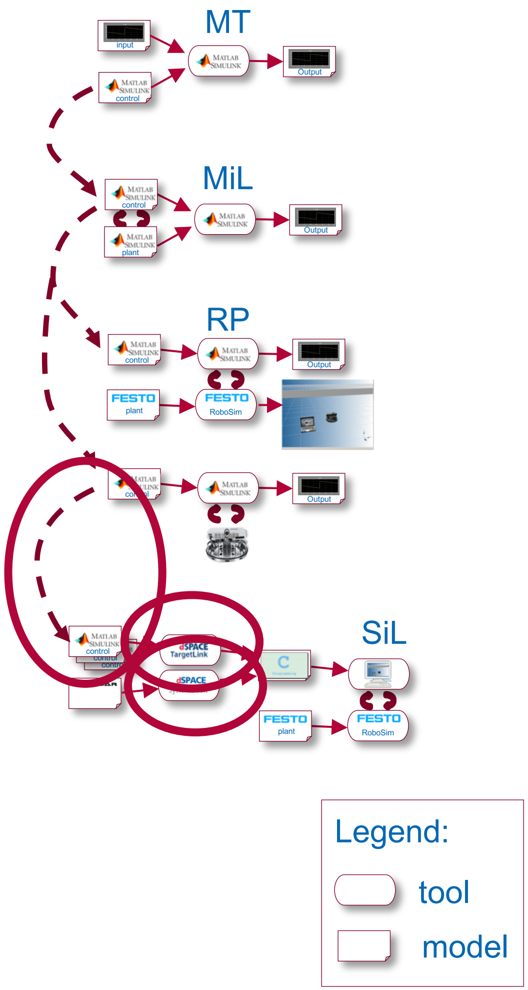


Legend:
- tool
- model

MT/MiL

- Vertical **refinement** of functional models (consistency manually)
- Horizontal **integration** of functional and plant models
- Horizontal **integration** of multiple functional models, an architecture model, and a plant model
- Vertical **refinement** of functional models (to realize functions while meeting resource constraints)

Vertical Enrichment & Transformation



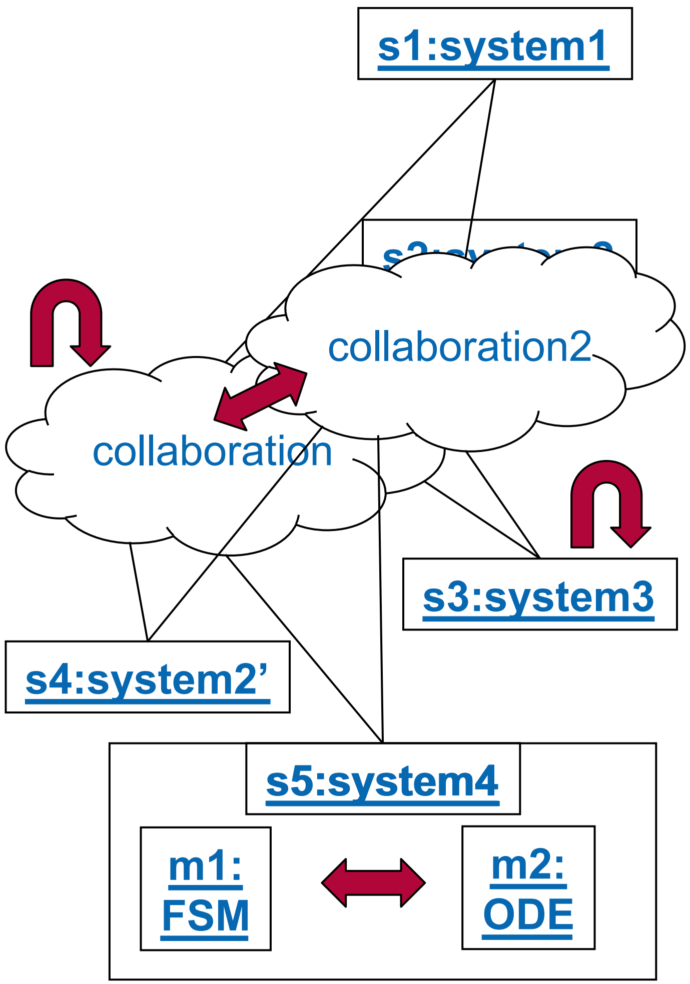
- Vertical **enrichment** of functional models and architecture
- Floating-Point 2 Fix-Point to reduce resource demands models (consistency manually)
- Fix-Point data-flow model 2 C-code models (consistency automatically)
- Autosar 2 C-code models (consistency automatically)

Different paradigms

Outline

1. Foundations
2. Cyber-Physical Systems
3. HPI CPSLab & Integration
4. **Future Needs for Integration**
5. Conclusion & Outlook

4. Future Needs for Integration



- **Operational** and **managerial independence**
  - operated independent from each other without global coordination
  - no centralized management decisions (possibly confliction decisions)
- **Dynamic architecture** and **openness**
  - must be able to dynamically adapt/absorb structural deviations
  - subsystems may join or leave over time in a not pre-planned manner
- **Advanced adaptation**
- **Resilience**
- **Cross-Domain Integration**
- **Integrate Models of Computation**

Bridging Paradigms & Formalism as Backbone

Tool-based integration of the models		
Requires an implicit notion of composition combining the formalisms of the models	Composition-based integration of the models	
Requires an implicit notion of formalism bridging the formalisms of the models	Requires an implicit notion of formalism bridging the formalisms of the models	Formalism-based integration of the models (formalism covers the formalisms of the models)

Overview over the Needs for Formalisms

Needs:

- **Operational** and **managerial independence**
- **Dynamic architecture** and **openness**
- **Scale** for local systems or networked resp. large-scale systems of systems
- **Integration** of the physical, cyber, (and social) dimension
- **Incremental adaptation** at the system and system of system level
- Independent **evolution** of the systems and joint **evolution** the system of system
- **Resilience** of the system of system

Model Characteristics:

- **Compositionality**
- **Dynamic structures**
- **Abstraction**
- **Hybrid behavior**
- **Non-deterministic**
- **Reflection for models**
- **Incremental extensions**
- **Probabilistic**

Coverage of the Needs for Formalisms

HPI Hasso Plattner Institut

Needs:

- **Operational** and **managerial independence**
- **Dynamic architecture** and **openness**
- **Scale** for local systems or networked resp. large-scale systems of systems
- **Integration** of the physical, cyber, (and social) dimension
- **Incremental adaptation** at the system and system of system level
- Independent **evolution** of the systems and joint **evolution** the system of system
- **Resilience** of the system of system

Model Characteristics:

- **Compositionality**
- **Dynamic structures**
- **Abstraction**
- **Hybrid behavior**
- **Non-deterministic**
- **Reflection for models**
- **Incremental extensions**
- **Probabilistic**

Our Work:

- **SMARTSOS (**employing Timed and Hybrid GTS [Giese+2015]**)**
- **Timed GTS ([**Becker&Giese2008]**)**
- **Hybrid GTS ([**Becker&Giese2012]**)**
- **Probabilistic GTS ([**Krause&Giese2012]**)**

Probabilistic timed GTS ([Maximova2018]**)**

?

BUT: We would need as foundation formalisms that supports all required characteristics at once!

Outline

1. Foundations
2. Cyber-Physical Systems
3. HPI CPSLab & Integration
4. Future Needs for Integration
5. **Conclusion & Outlook**

5. Conclusion & Outlook

HPI Hasso Plattner Institut

- **Multiple models** and their **integration** is the heart of the matter developing complex systems
- In case of **cyber-physical systems** it holds:
  - models employ **different paradigms** specific for their **layer** and/or **domain**
  - **Integration** of the models is of **paramount importance** during the development
- **Current challenges:**
  - Build cost-effectively the required **formalisms** / **compositions** / **tools** to integrate the models
  - Support analysis also for **emergent properties**

Conclusion & Outlook

- Future **cyber-physical systems** have many **additional needs** (compositionality, dynamic structures, reflection, ...) we have to address **at once** (via formalism, composition, or tool).
- **Future challenges:**
  - Setup the **foundation** for the required **formalisms** / **compositions** / **tools** to integrate the models covering the additional needs
  - Support analysis for **emergent properties** covering also the additional needs
  - Support integration **at runtime**

Bibliography (1/3)

[Broman+2012] David Broman, Edward A. Lee, Stavros Tripakis and Martin Torngren. Viewpoints, Formalisms, Languages, and Tools for Cyber-physical Systems. In Proceedings of the 6th International Workshop on Multi-Paradigm Modeling, Pages 49--54, ACM, New York, NY, USA, 2012.

[Brooks+2008] Christopher Brooks, Chihhong Cheng, Thomas Huining Feng, Edward A. Lee and Reinhard von Hanxleden. Model Engineering using Multimodeling. In 1st International Workshop on Model Co-Evolution and Consistency Management (MCCM '08), September 2008.

[Broy+2012] Manfred Broy, MaríaVictoria Cengarle and Eva Geisberger. Cyber-Physical Systems: Imminent Challenges. In Radu Calinescu and David Garlan editors, Large-Scale Complex IT Systems. Development, Operation and Management, Vol. 7539:1-28 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2012.

[Becker+2006] Basil Becker, Dirk Beyer, Holger Giese, Florian Klein and Daniela Schilling. Symbolic Invariant Verification for Systems with Dynamic Structural Adaptation. In Proc. of the 28th International Conference on Software Engineering (ICSE), Shanghai, China, ACM Press, 2006.

[Becker&Giese2008] Basil Becker and Holger Giese. On Safe Service-Oriented Real-Time Coordination for Autonomous Vehicles. In In Proc. of 11th International Symposium on Object/component/service-oriented Real-time distributed Computing (ISORC), Pages 203--210, IEEE Computer Society Press, 5-7 May 2008.

[Becker&Giese2012] Basil Becker and Holger Giese. Cyber-Physical Systems with Dynamic Structure: Towards Modeling and Verification of Inductive Invariants. Technical report, 64, Hasso Plattner Institute at the University of Potsdam, Germany, 2012.

[Burmester+2008] Sven Burmester, Holger Giese, Eckehard Münch, Oliver Oberschelp, Florian Klein and Peter Scheideler. Tool Support for the Design of Self-Optimizing Mechatronic Multi-Agent Systems. In International Journal on Software Tools for Technology Transfer (STTT), Vol. 10(3):207-222, Springer Verlag, June 2008.

[Giese+2010] Holger Giese, Stefan Neumann and Stephan Hildebrandt. Model Synchronization at Work: Keeping SysML and AUTOSAR Models Consistent. In Gregor Engels, Claus Lewerentz, Wilhelm Schäfer, Andy Schürr and B. Westfechtel editors, Graph Transformations and Model Driven Engineering - Essays Dedicated to Manfred Nagl on the Occasion of his 65th Birthday, Vol. 5765:555-579 of

Bibliography (2/3)

[Giese+2011] Holger Giese, Stefan Henkler and Martin Hirsch. A multi-paradigm approach supporting the modular execution of reconfigurable hybrid systems. In Transactions of the Society for Modeling and Simulation International, SIMULATION, Vol. 87(9):775-808, 2011.

[Giese+2015] Holger Giese, Thomas Vogel and Sebastian Wätzoldt. Towards Smart Systems of Systems. In Mehdi Dastani and Marjan Sirjani editors, Proceedings of the 6th International Conference on Fundamentals of Software Engineering (FSEN '15), Vol. 9392:1--29 of Lecture Notes in Computer Science (LNCS), Springer, 2015.

[Giese&Schäfer2013] Holger Giese and Wilhelm Schäfer. Model-Driven Development of Safe Self-Optimizing Mechatronic Systems with MechatronicUML. In Javier Camara, Rogério de Lemos, Carlo Ghezzi and AntÃ³nia Lopes editors, Assurances for Self-Adaptive Systems, Vol. 7740:152-186 of Lecture Notes in Computer Science (LNCS), Springer, January 2013.

[Ghezzi2012| Carlo Ghezzi. Evolution, Adaptation, and the Quest for Incrementality. In Radu Calinescu and David Garlan editors, Large-Scale Complex IT Systems. Development, Operation and Management, Vol. 7539:369-379 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2012.

[Krause&Giese2012] Christian Krause and Holger Giese. Probabilistic Graph Transformation Systems. In Proceedings of Intern. Conf. on Graph Transformation (ICGT' 12), Vol. 7562:311-325 of Lecture Notes in Computer Science, Springer-Verlag, 2012.

[Maier1998] Mark W. Maier. Architecting principles for systems-of-systems. In Systems Engineering, Vol. 1(4):267--284, John Wiley & Sons, Inc., 1998.

[Maximova2018] Maria Maximova, Holger Giese and Christian Krause. Probabilistic Timed Graph Transformation Systems. In Journal of Logical and Algebraic Methods in Programming, Vol. 101:110 - 131, 2018.

[Northrop+2006] Northrop, Linda, et al. Ultra-Large-Scale Systems: The Software Challenge of the Future. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2006.

Bibliography (3/3)

[Pereira+2013]	Eloi Pereira, Christoph M. Kirsch, Raja Sengupta and Jo~ao Borges de Sousa. Bigactors - A Model for Structure-aware Computation. In ACM/IEEE 4th International Conference on Cyber-Physical Systems, Pages 199--208, ACM/IEEE, Philadelphia, PA, USA, 2013.
[Sztipanovits2011]	Janos Sztipanovits with Ted Bapty, Gabor Karsai and Sandeep Neema. MODEL-INTEGRATION AND CYBER PHYSICAL SYSTEMS: A SEMANTICS PERSPECTIVE. FM 2011, Limerick, Ireland. 22 June 2011
[Sztipanovits+2012]	Janos Sztipanovits, Xenofon Koutsoukos, Gabor Karsai, Nicholas Kottenstette, Panos Antsaklis, Vineet Gupta, B. Goodwine, J. Baras and Shige Wang. Toward a Science of Cyber-Physical System Integration. In Proceedings of the IEEE, Vol. 100(1):29-44, January 2012.
[Valerdi+2008]	Ricardo Valerdi, Elliot Axelband, Thomas Baehren, Barry Boehm, Dave Dorenbos, Scott Jackson, Azad Madni, Gerald Nadler, Paul Robitaille and Stan Settles. A research agenda for systems of systems architecting. In International Journal of System of Systems Engineering, Vol. 1(1-2):171--188, 2008.
[Vogel+2009]	Thomas Vogel, Stefan Neumann, Stephan Hildebrandt, Holger Giese and Basil Becker: Model-Driven Architectural Monitoring and Adaptation for Autonomic Systems. In: Proc. of the 6th International Conference on Autonomic Computing and Communications (ICAC'09), Barcelona, Spain, ACM (15-19 June 2009)
[Vogel+2010]	Thomas Vogel and Stefan Neumann and Stephan Hildebrandt and Holger Giese and Basil Becker. Incremental Model Synchronization for Efficient Run-Time Monitoring. In Sudipto Ghosh, ed., Models in Software Engineering, Workshops and Symposia at MODELS 2009, Denver, CO, USA, October 4-9, 2009, Reports and Revised Selected Papers, vol. 6002 of Lecture Notes in Computer Science (LNCS), pages 124-139. Springer-Verlag, 4 2010.
[Vogel&Giese2012]	Thomas Vogel and Holger Giese. A Language for Feedback Loops in Self-Adaptive Systems: Executable Runtime Megamodels. In Proceedings of the 7th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS 2012), pages 129-138, 6 2012. IEEE Computer Society.