Hasso Plattner Institut

IT Systems Engineering | Universität Potsdam

Model-Driven Engineering for Cyber-Physical Systems

Kolloquiumsvortrag, Department Elektrotechnik und Informatik, Universität Siegen, 8th May 2015.

UNIVERSITÄT SIEGEN

Fakultät IV: Department Elektrotechnik und Informatik

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

Outline

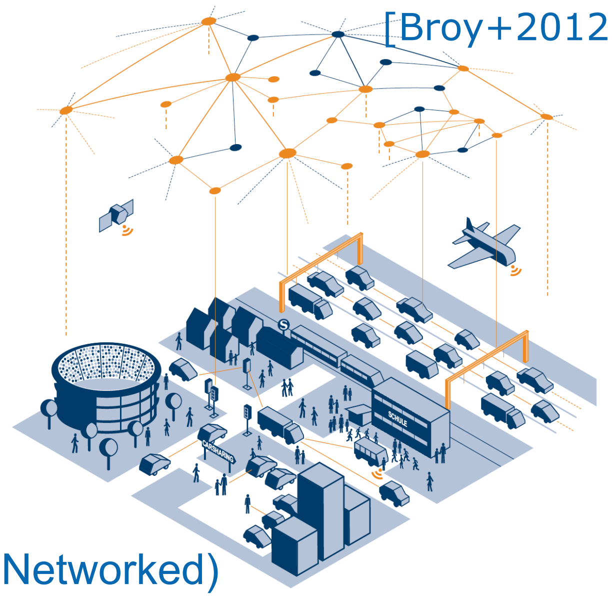
1. **Challenges Ahead**
2. **Available Options**
3. **Example: Mechatronic UML**
4. **Conclusions & Outlook**

Outline

1. **Challenges Ahead**
   - Cyber-Physical Systems
   - System of Systems
   - Ultra-Large-Scale Systems
2. **Available Options**
3. **Example: Mechatronic UML**
4. **Conclusions & Outlook**

Envisioned Challenges for Future Embedded Systems

Hasso Plattner Institut

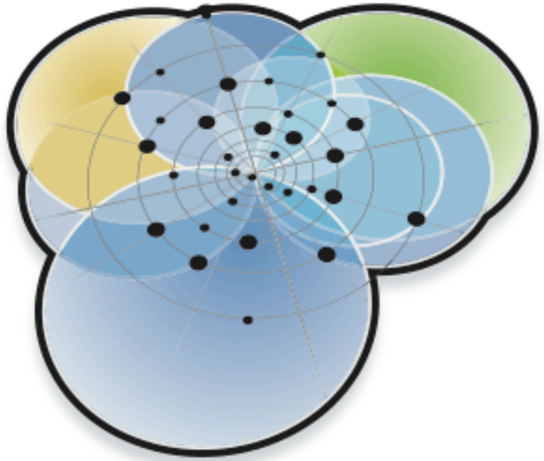


[Broy+2012]

(Networked)
Cyber-Pyhsical Systems

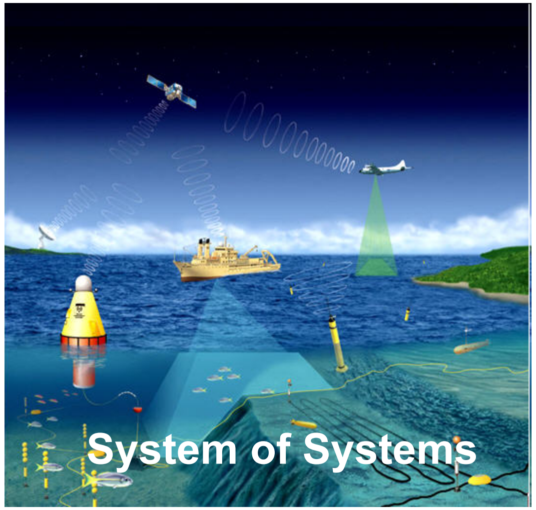
Internet of Things

Smart City



[Northrop+2006]

Ultra-Large-Scale Systems



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

Smart Factory -
E.g. Industry 4.0

Smart Logistic

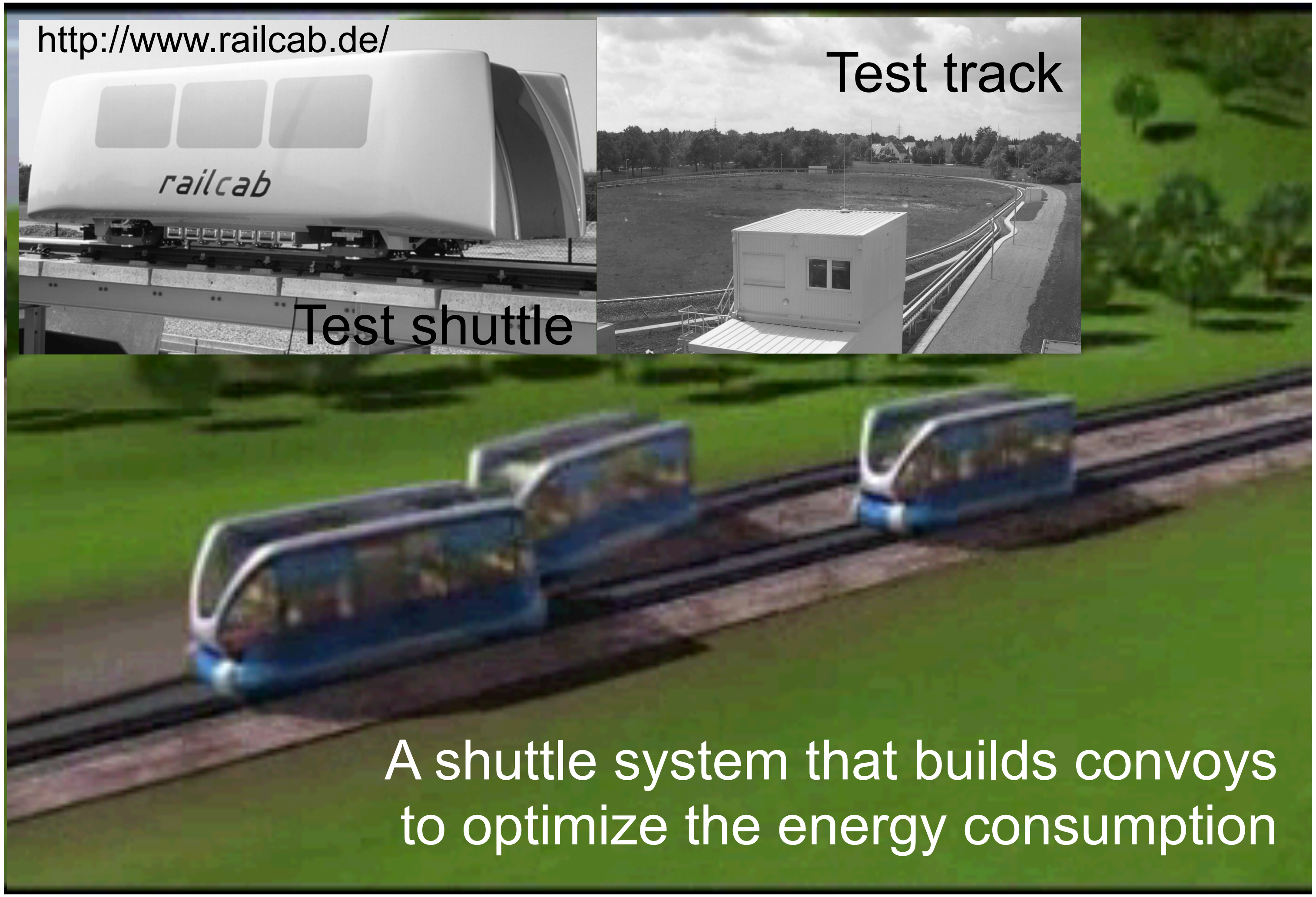
Micro Grids

Smart Home

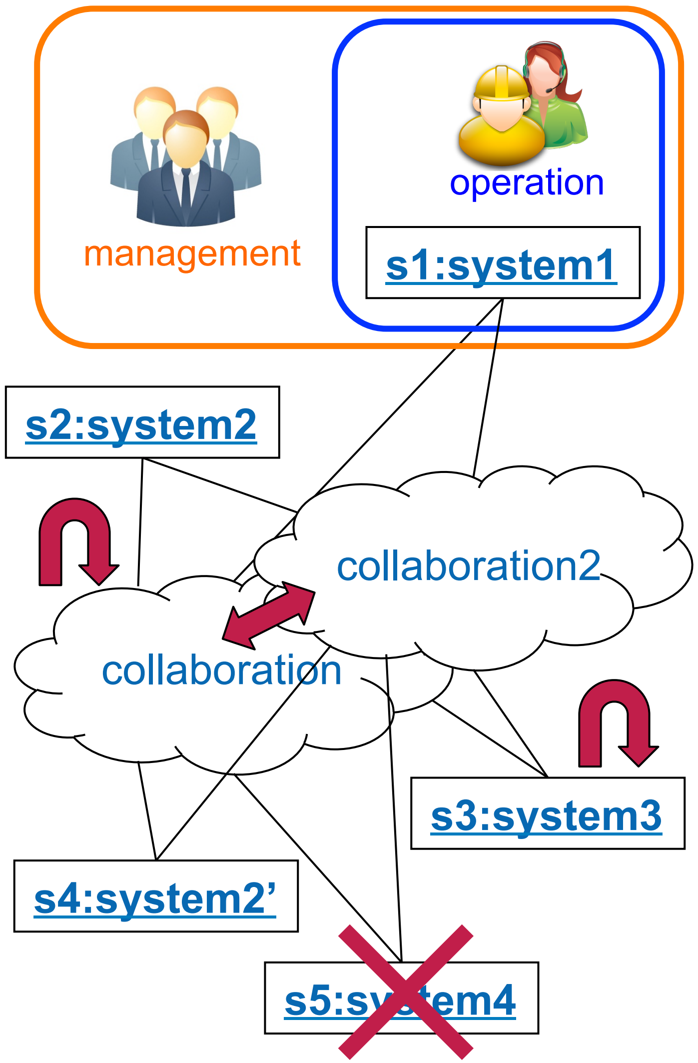
E-Health

Ambient
Assisted Living

RailCab Example: A Short Video …



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**
- **Integration**
- **Resilience**

Challenge: Operational and Managerial Independence

"A system-of-systems is an assemblage of components which individually may be regarded as systems, and which possesses two additional properties:

- **Operational Independence** of the Components: If the system-of-systems is disassembled into its component systems the component systems must be able to usefully operate independently. That is, the components fulfill customer-operator purposes on their own.
- **Managerial Independence** of the Components: The component systems not only can operate independently, they do operate independently. The component systems are separately acquired and integrated but maintain a continuing operational existence independent of the system-of-systems."

[Maier1998]

Challenge: Dynamic Architecture and Openness

"The sheer scale of ULS systems will change everything. ULS systems will necessarily be **decentralized** in a variety of ways, developed and used by a wide variety of stakeholders with conflicting needs, **evolving continuously**, and constructed from heterogeneous parts." [Northrop+2006]

"The vision of Cyber-Physical System (CPS) is that of open, ubiquitous systems of coordinated computing and physical elements which interactively adapt to their context, are capable of learning, dynamically and automatically reconfigure themselves and **cooperate with other CPS** (resulting in a compound CPS), possess an adequate man-machine interface, and fulfill stringent safety, security and private data protection regulations." [Broy+2012]

Required capabilities:

- must be able to dynamically adapt/absorb structural deviations
- systems may join/leave over time in a not pre-planned manner

Challenge: Advanced Adaptation

"**Adaptation** is needed to compensate for changes in the mission requirements […] and operating environments […]"

[Northrop+2006]

"The vision of Cyber-Physical System (CPS) is that of open, ubiquitous systems of coordinated computing and physical elements which interactively **adapt to their context, are capable of learning, dynamically and automatically reconfigure themselves** and cooperate with other CPS (resulting in a compound CPS), possess an adequate man-machine interface, and fulfill stringent safety, security and private data protection regulations." [Broy+2012]

Required kind of adaptation:

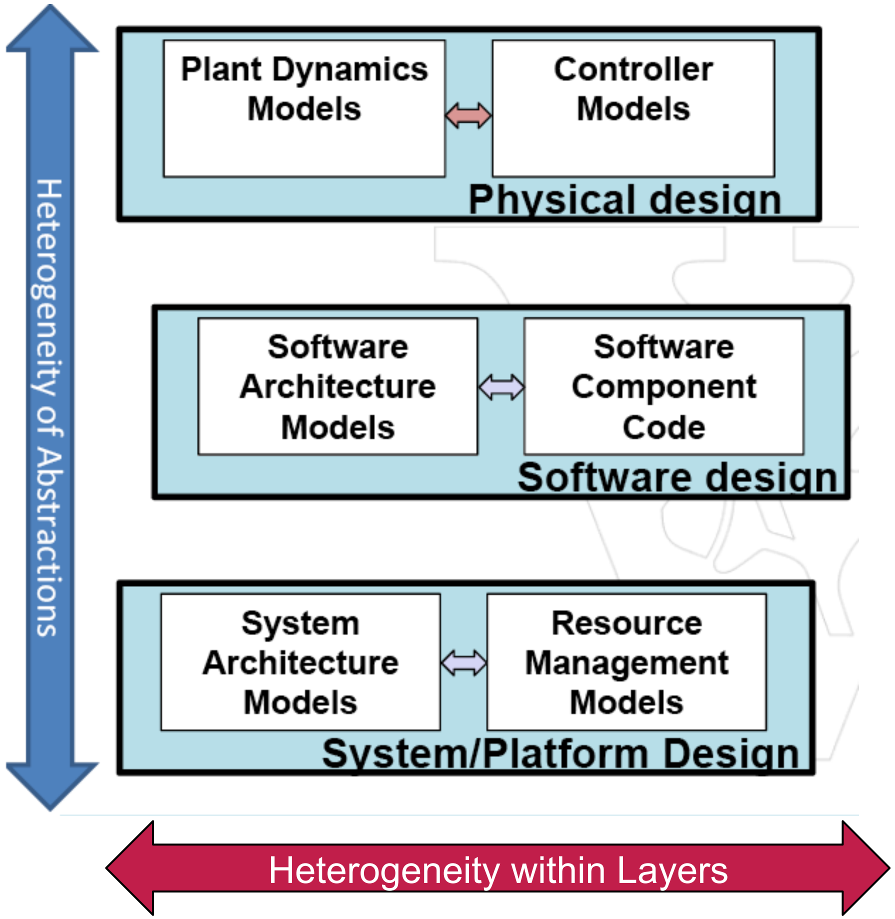
- System level adaptation
- System-of-systems level adaptation

Challenge: Integration (1/2)

[Sztipanovits2011]

Model Integration?

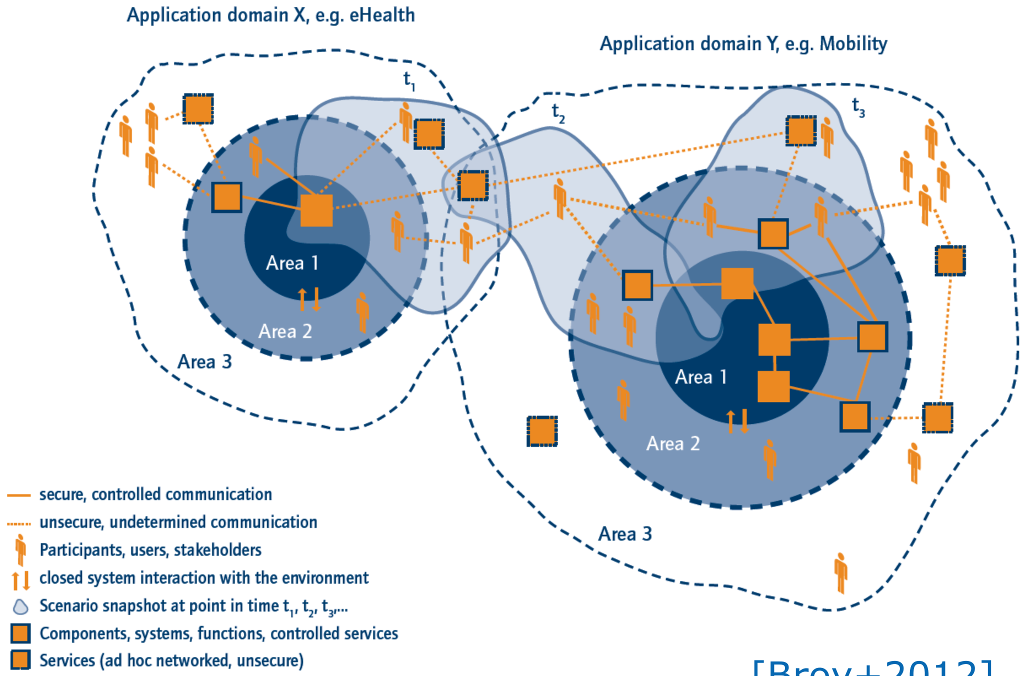
- Problem to integrate models within one layer as different models of computation are employed
- Leaky abstractions are caused by lack of composability across system layers. Consequences:
  - intractable interactions
  - unpredictable system level behavior
  - full-system verification does not scale



Challenge: Integration (2/2)

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.

Challenge: Resilience

"The vision of Cyber-Physical System (CPS) is that of open, ubiquitous systems […] which […] and **fulfill stringent safety, security and private data protection regulations**." [Broy+2012]

"Resilience[:] This area is the attribute of a system, in this case a SoS that makes it less likely to experience failure and more likely to recover from a major disruption." [Valerdi+2008]

"Resilience is the capability of a system with specific characteristics before, during and after a disruption to absorb the disruption, recover to an acceptable level of performance, and sustain that level for an acceptable period of time." Resilient Systems Working Group, INCOSE

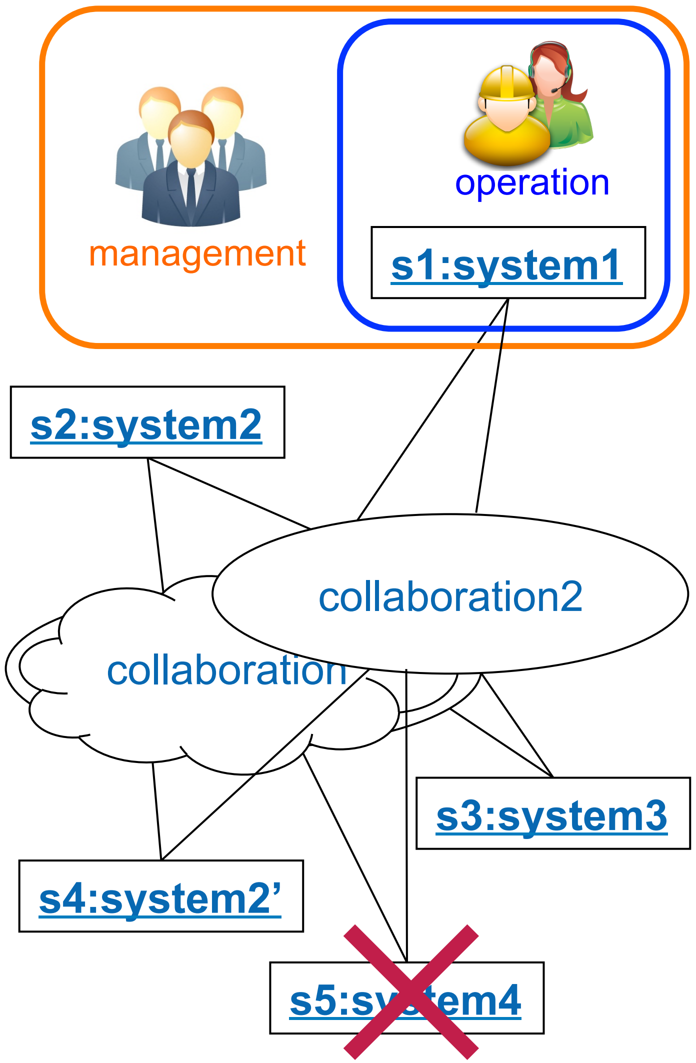
Required coverage of resilience:

- Physical and control elements (via layers of idealization)
- Software elements (via layers of abstraction)
- Horizontal and vertical composition of layers

Outline

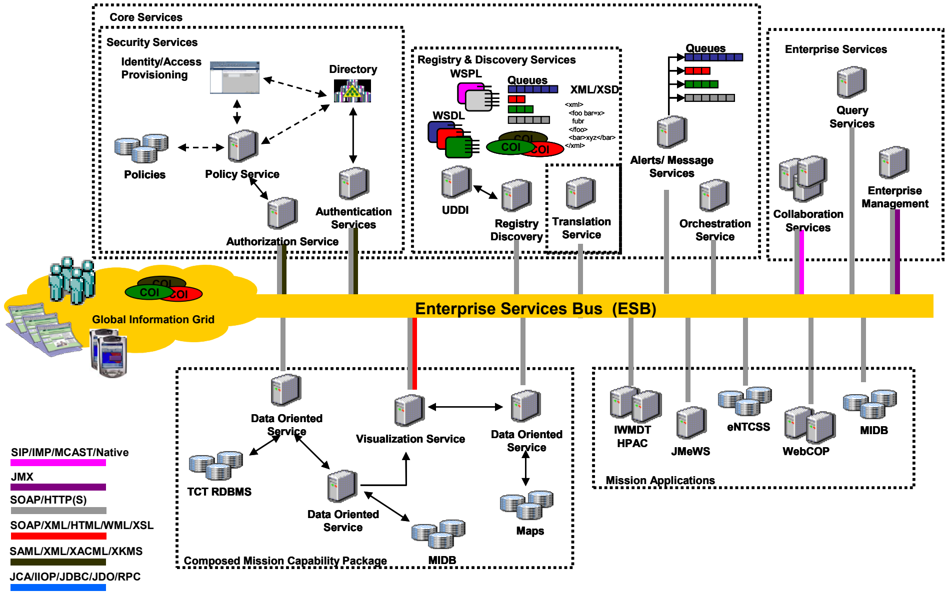
1. **Challenges Ahead**
2. **Available Options**
   - Service-Oriented Architecture
   - Self-Adaptive & Self-Organization
   - Multi-Paradigm Modeling
3. **Example: Mechatronic UML**
4. **Conclusions & Outlook**

Option: Service-Oriented Architecture



- **Service-Oriented Architecture:**
  - Dedicated services are offered by systems via defined **service contracts** can be offered, looked up, and bound at run-time
  - Interoperability is provided by a **service bus**
- **Service oriented architecture Modeling Language (SoaML)**
  - a UML profile for modeling
  - Support **collaborations** as first class elements (service contracts)
  - Links collaborations with **component-based models**

Option: Service-Oriented Architecture

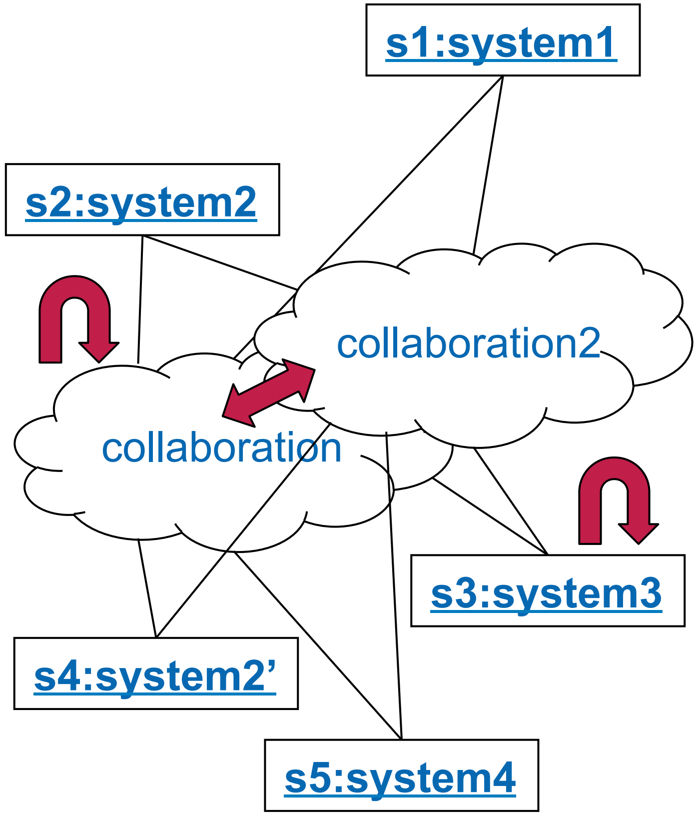


Challenges \ Approaches	SOA
Operational and managerial independence	✓
Dynamic architecture and openness	✓
Advanced adaptation	(✓)
Integration	(✓)
Resilience	✗

Observations:

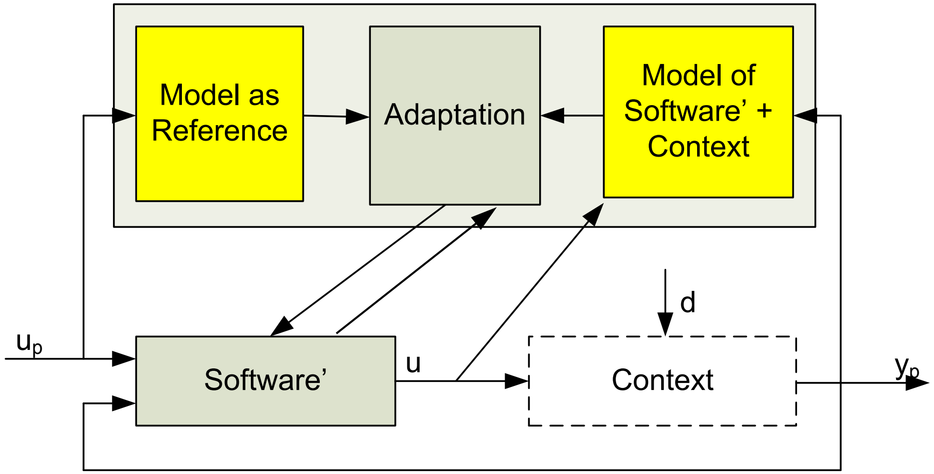
- Service contracts permit to realize **operational** and **managerial independence**
- Offering, look up, and bin service and runtime supports **dynamic architectures** and **openness** (but not modeled)
- Under-specification in the service contracts preserves degrees of freedom for **adaptation** of the components (but not at the level of the collaboration)
- Service contracts can make cross-domain **integration** possible (but also required mapping concepts are not supported)
- No specific support for **resilience**

Option: Self-Adaptive & Self-Organization



- **Self-Adaptive Systems:**
  - Make systems self-aware, context-aware, and requirements-aware using some form of **reflection**
  - Enable systems to adjust their structure/behavior accordingly
- **Self-Organization:**
  - The capability of a group of systems to organize their structure/behavior without a central control (emergent behavior)
- **Engineering perspective:**
  - a spectrum from centralized top-down self-adaptation to decentralized bottom-up self-organization with many intermediate forms (e.g. partial hierarchies) exists

Option: Self-Adaptive & Self-Organization

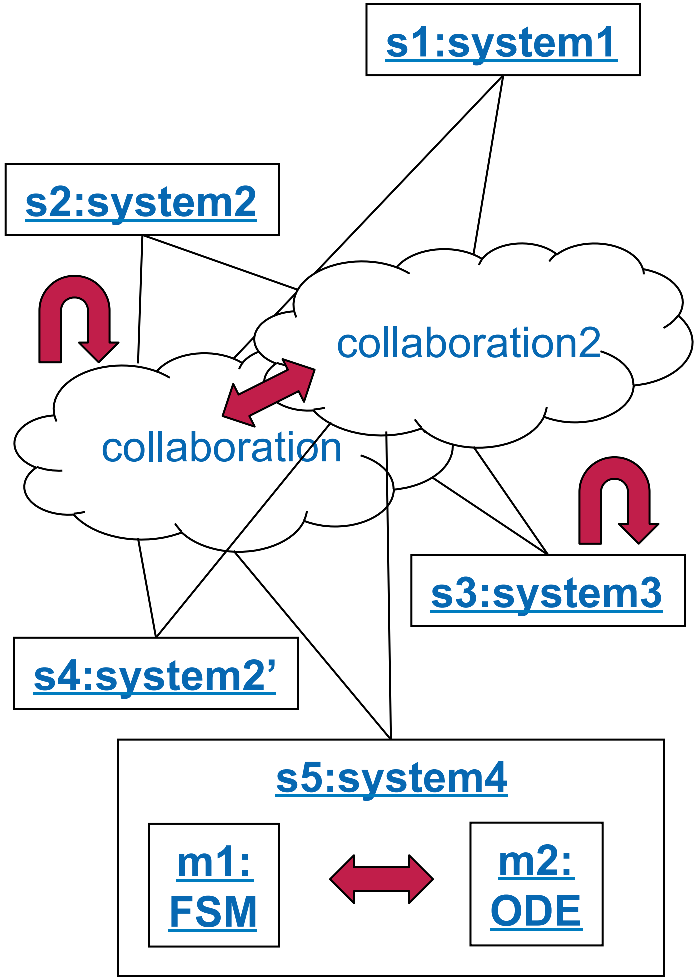


Challenges \ Approaches	Self-Adaptive / Self-Org
Operational and managerial independence	(✓)
Dynamic architecture and openness	(✓)
Advanced adaptation	✓
Integration	(✓)
Resilience	?

Observations:

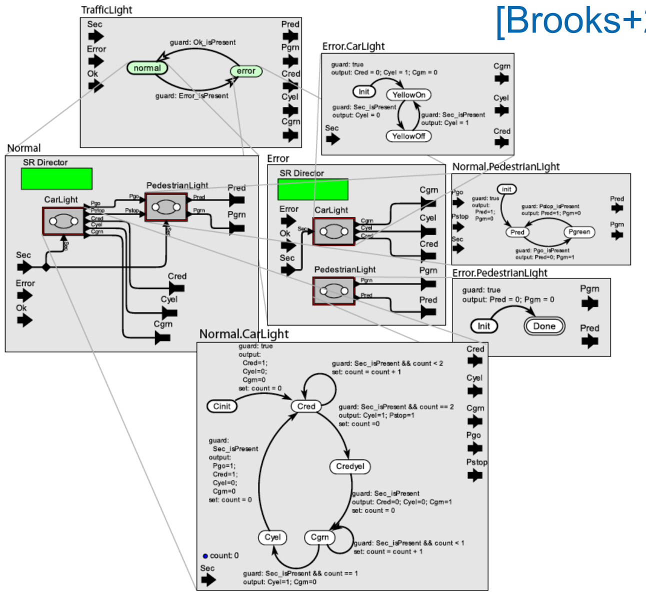
- Can **co-exist** with managerial and operational independence as well as dynamic architecture and openness, but both make the problem considerable harder
- Self-adaptive systems enable advanced adaptation at the **system-level** while self-organization cover the **system-of-systems-level**
- Cross-domain **integration** is possible (but there is no support for adaptation across the domains)
- While both self-adaptive behavior as well as self-organization **can contribute to resilience**, it also makes the problem considerable harder

Option: Multi-Paradigm Modeling



- **Multi-Paradigm Modeling:**
  - Enable to use different domain-specific models with different models of computation for different modeling aspects
  - Can be employed at the system-level to combine all necessary models for a system
  - Can be employed at the system-of-systems-level to combine all necessary models for a system-of-systems
  - Requires that for employed model combinations a suitable semantic integration is known (and supported by the tools)

Option: Multi-Paradigm Modeling



[Brooks+2008]

Challenges \ Approaches	Multi-Paradigm Modeling
Operational and managerial independence	(✓)
Dynamic architecture and openness	✗
Advanced adaptation	(✓)
Integration	✓
Resilience	✗

Observations:

- Can **co-exist** with managerial and operational independence as well advanced adaptation, but both make multi-paradigm modeling considerable harder
- The multi-paradigm modeling approaches assume a fixed hierarchical structure and therefore do no fit to dynamic architectures and openness (exceptions: [Giese+2011] for a specific case and [Pereira+2013] for a MoC)
- integration is well supported for the models and also across domains
- Leaky abstractions caused by lack of composability across system layers make it hard to achieve **resilience** (exceptions: [Sztipanovits+2012] for stability and [Giese&Schäfer2013] for safety)

Overview Concerning the Options

Challenges \ Approaches	SOA	Self-Adaptive / Self-Org	Multi-Paradigm Modeling
Operational and managerial independence	✓	(✓)	(✓)
Dynamic architecture and openness	✓	(✓)	✗
Advanced adaptation	(✓)	✓	(✓)
Integration	(✓)	(✓)	✓
Resilience	✗	?	✗

- Besides Resilience all challenges can be covered by one of the available options

Are we Ready to for the Envisioned Future Cyber-Physical Systems? **We need a combination!**

Outline

1. Challenges Ahead

2. Available Options

3. Example: Mechatronic UML

- Micro Architecture
- Macro Architecture

4. Conclusions & Outlook

Example: MechatronicUML

At the level of code it seems impossible to build trustworthy advanced Cyber-Physical Systems:

Modeling separately

- the integration of intelligent behavior,
- the integration with control theory,

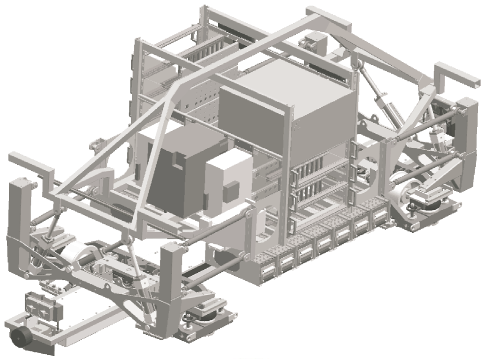
} Micro Architecture

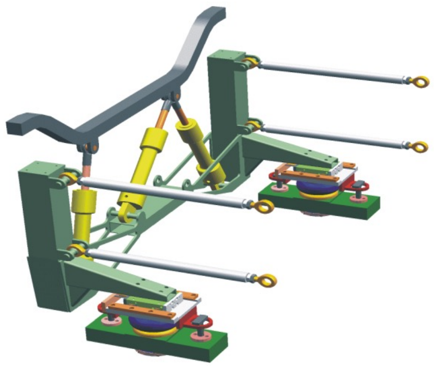
- the real-time coordination, and
- the reconfiguration at the level of agents.

} Macro Architecture

- Analyze the models in a compositional manner
- Synthesize the code

Application Example: Railcab System

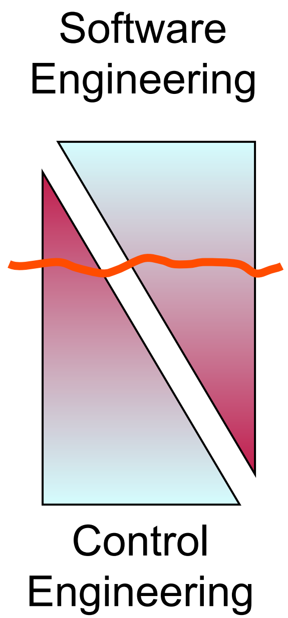




Domains:

- Logistic
- Real-time coordination
- Local control
- Electronics
- Mechanics

(Local control, Electronics, Mechanics: Classical Engineering (Mechatronics))



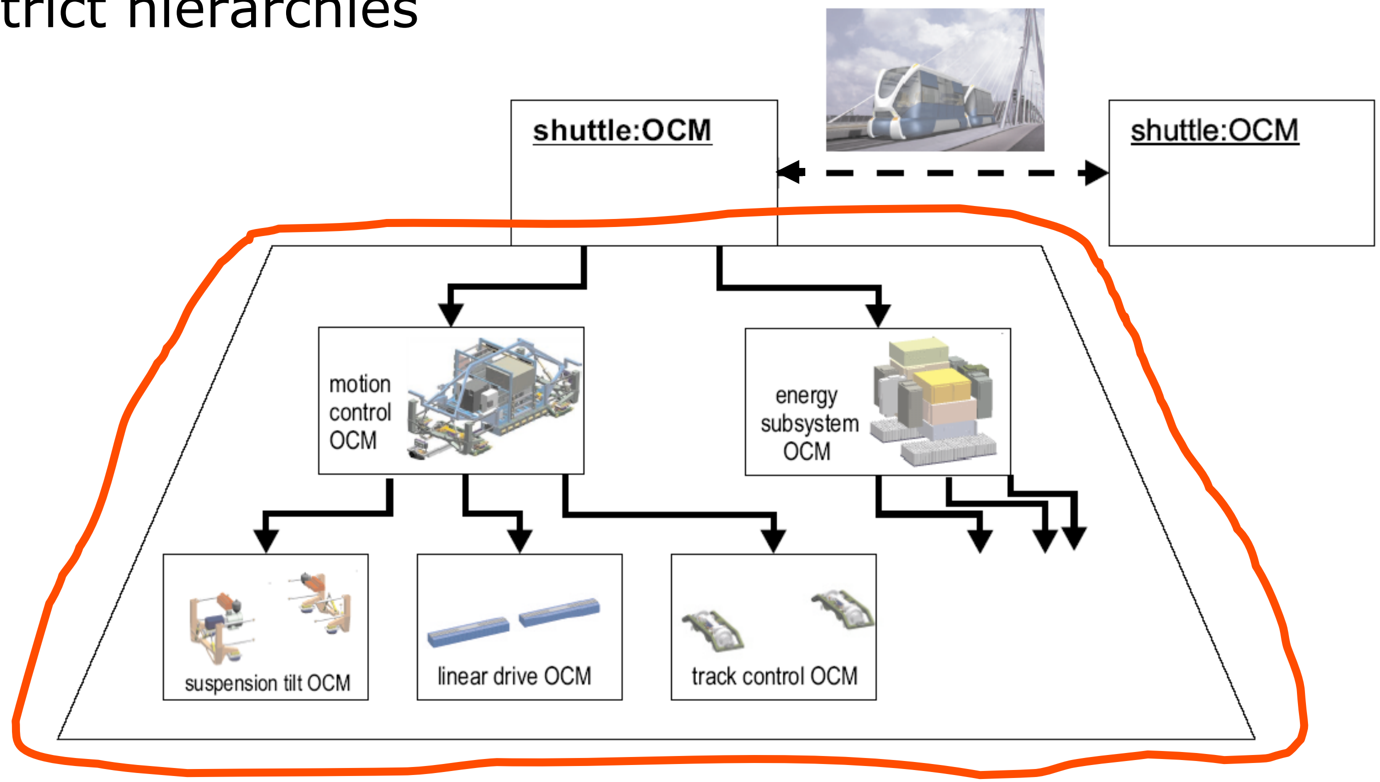
⇨ Integration of the different worlds

⇨ Self-optimization at multiple levels

⇨ Self-adaptation/self-coordination via software

Micro Architecture

- **Autonomous subsystems (shuttles)**
- Within: strict hierarchies



Micro Architecture

Operator-Controller Module [ICINCO04]

- **Cognitive operator ("intelligence")**

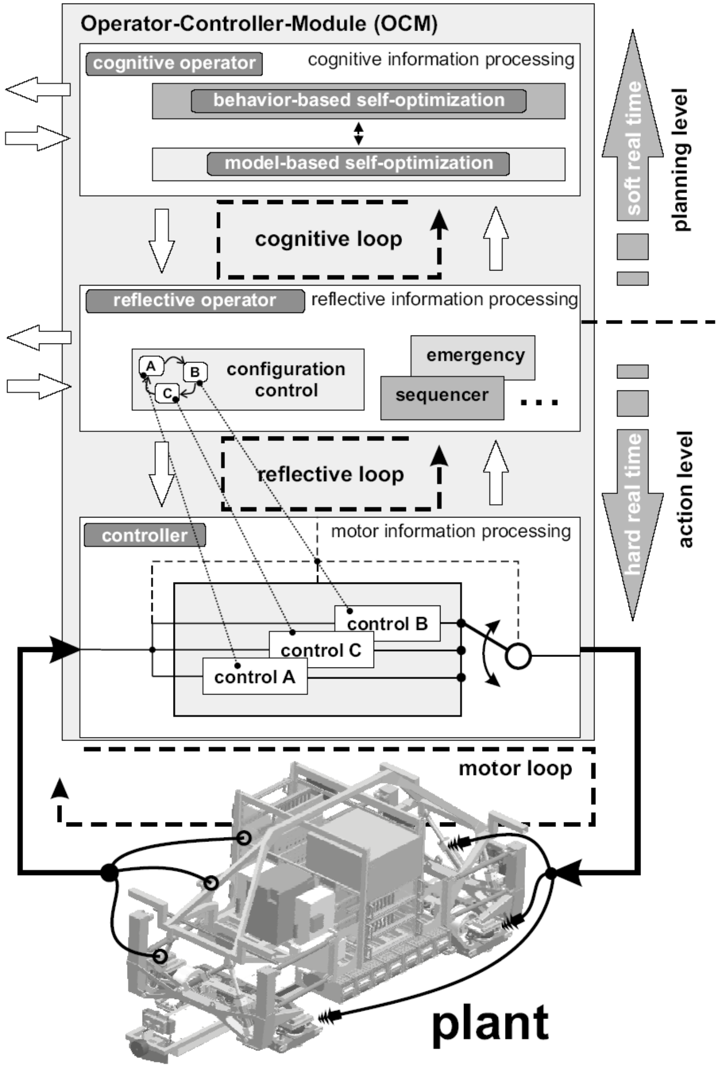
  decoupled from the hard real-time processing

- **Reflective operator**

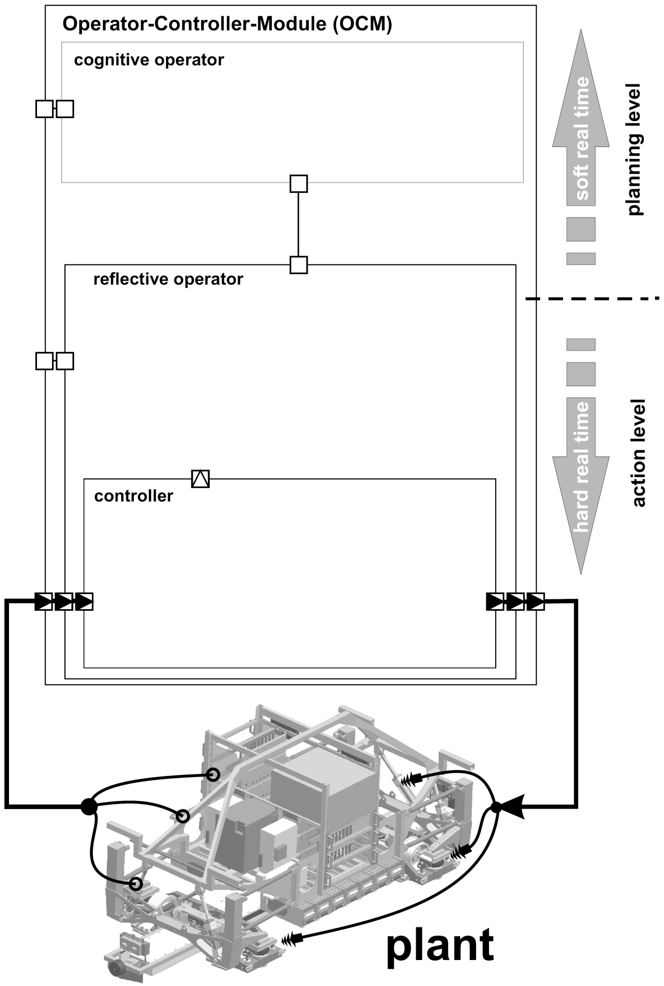
  Real-time coordination and reconfiguration

- **Controller**

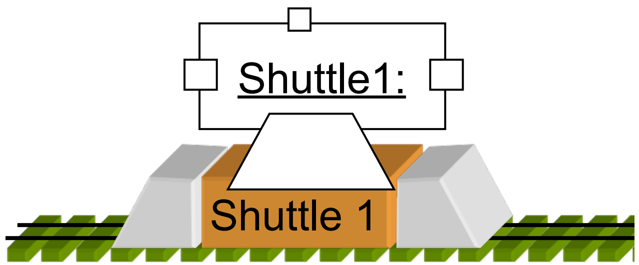
  Control via sensors and actuators in hard real-time



MECHATRONIC UML: Components

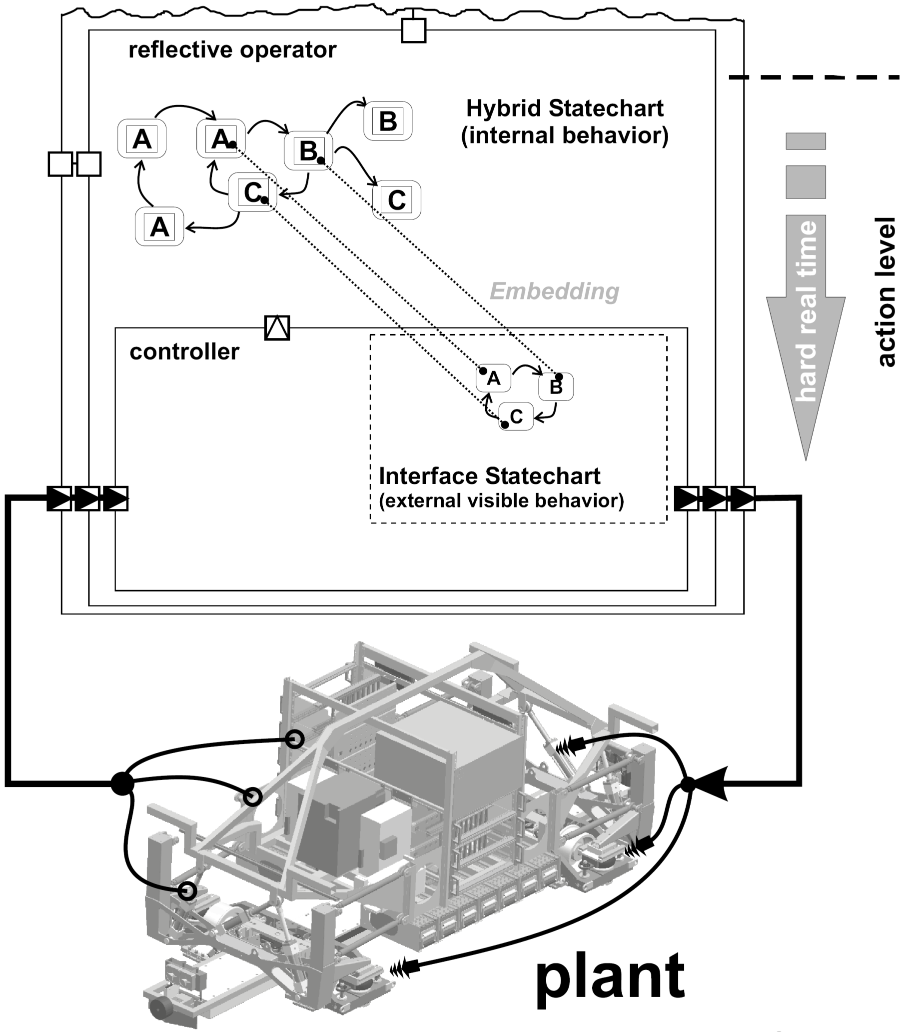


- Model the structure of the Software with hybrid UML components with
- Hybrid behavior
  - Regular ports (discrete)
  - Continuous ports
  - Hybrid ports
- Reconfiguration
  - Permanent ports
  - potential ports



Integration Reflective Operator & Controller

- Hybrid components
  - UML components (Fujaba)
  - Block diagrams (CAMeL)
- Hybride Statecharts can embed subordinated hybrid components
  - Controller or
  - The reflective operator of subordinated OCMs
- Interface statecharts enable **modular reconfiguration** across the boundaries of hybrid components
- Automatic **check** for correct embedding

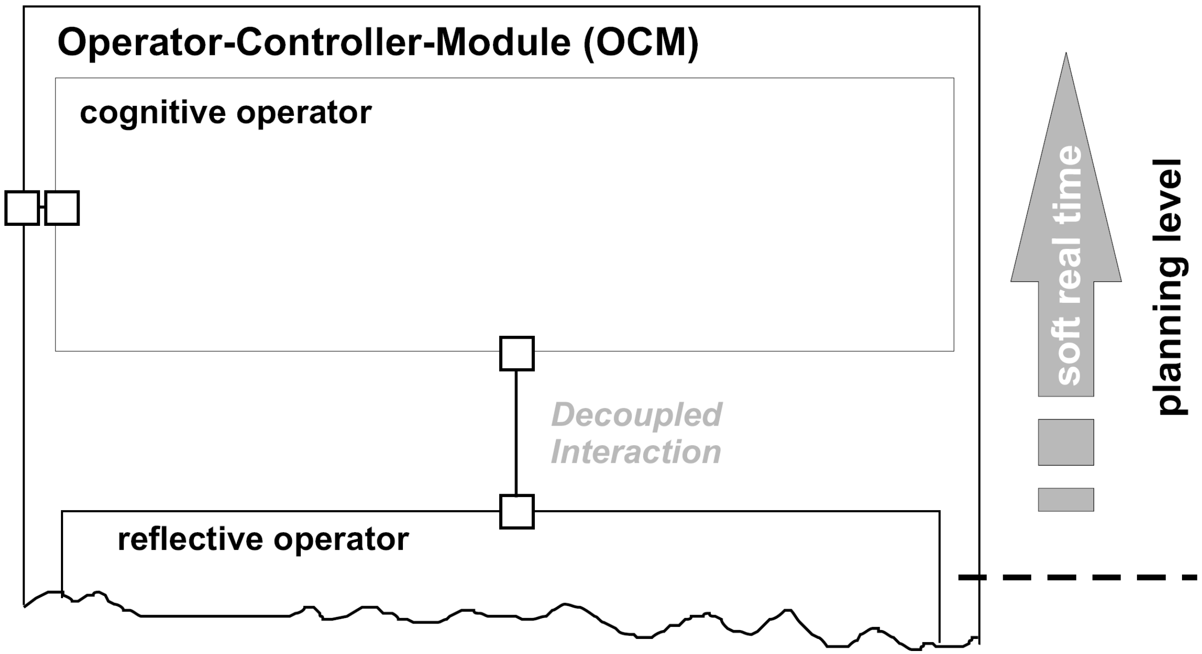


[FSE04]

Integration Cognitive & Reflective Operator

The cognitive operator is decoupled from the rest:

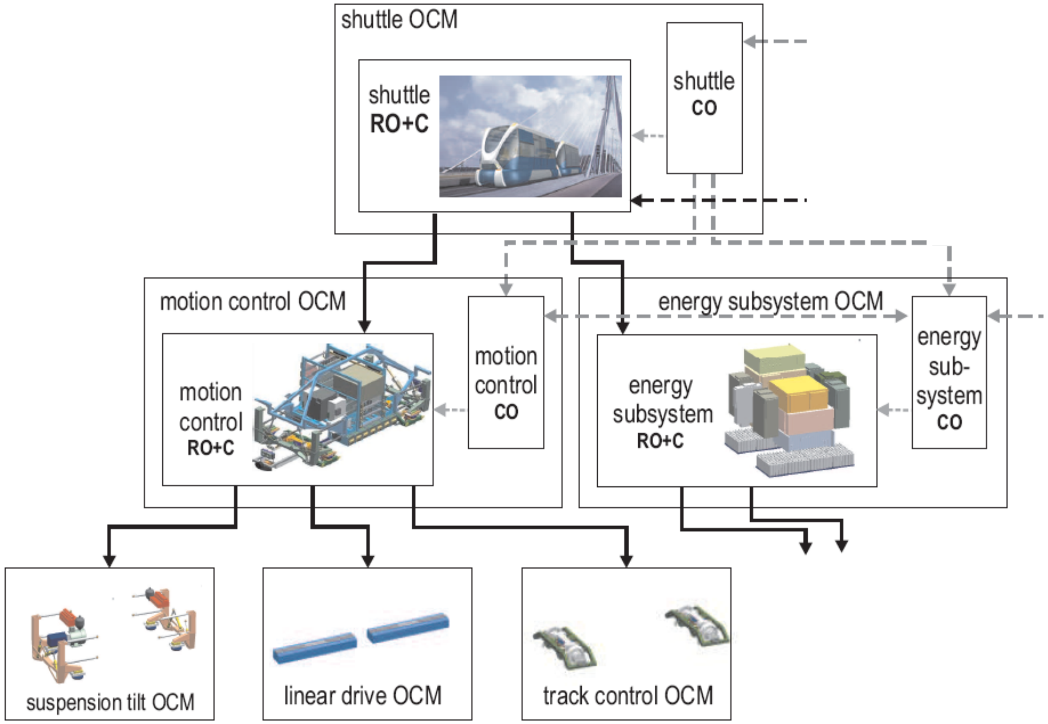
- We **check** that the reflective operator realizes a "***Filter***" which excludes unsafe reactions.
- The cognitive operator can "**guide**" the reflective operator as long as the commands given are considered to be safe and occur in time.



Strict Hierarchies

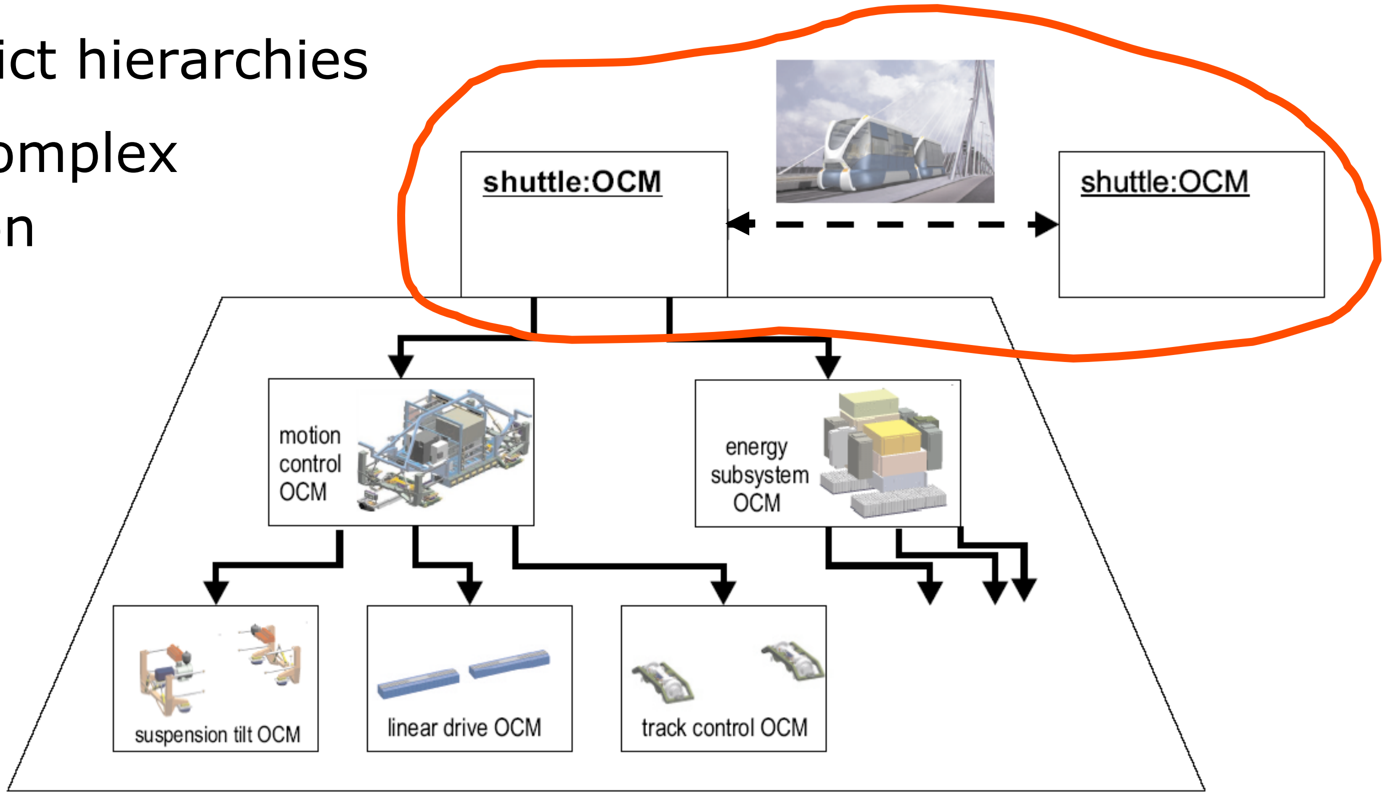
Concepts [FSE04]:

- Hybrid components: UML components or block diagrams
- Hybride Statecharts embed hybrid components (controller or the **reflective operator** of subordinated OCMs)
- Interface statecharts enable **modular reconfiguration** across the boundaries of hybrid components



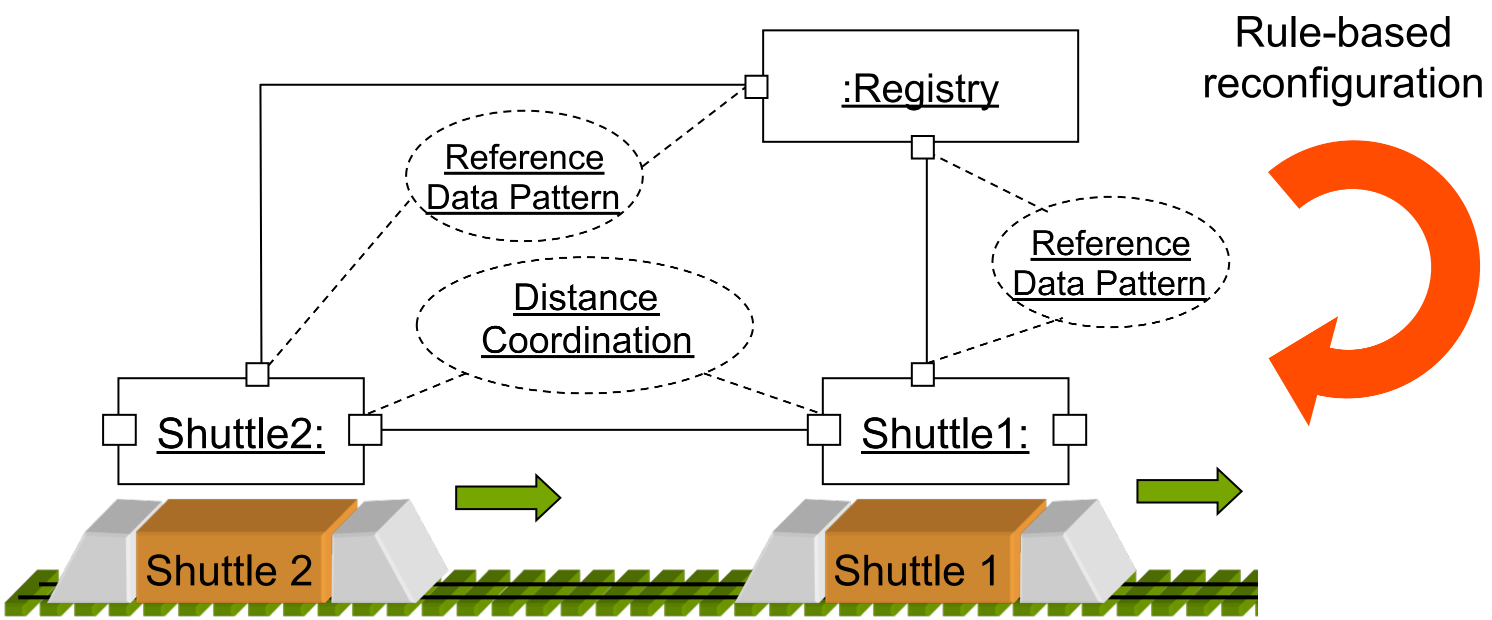
Macro Architecture

- **Autonomous subsystems (shuttles)**
- Within: strict hierarchies
- Outside: complex coordination

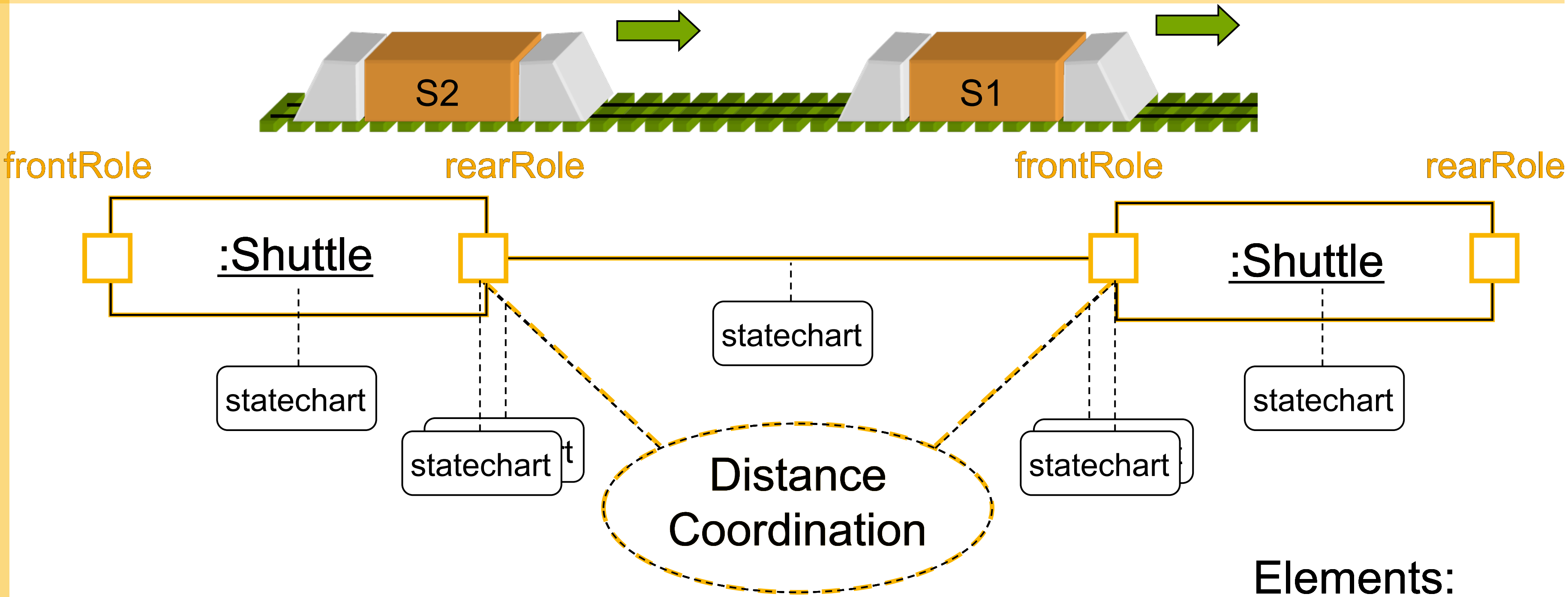


Complex Coordination

- **Real-time coordination via pattern [ESEC/FSE03]**
  - Real-time protocol state machines for each role
  - Real-time state machines for each connector
- **Rule-based reconfiguration (self-coordination) [ICSE06]**
  - Rules for instantiation and deletion of patterns



Real-Time Coordination via Patterns



Pattern (Distance Coordination):

- Model: Statecharts for roles and connector
- Specification: required OCL RT properties

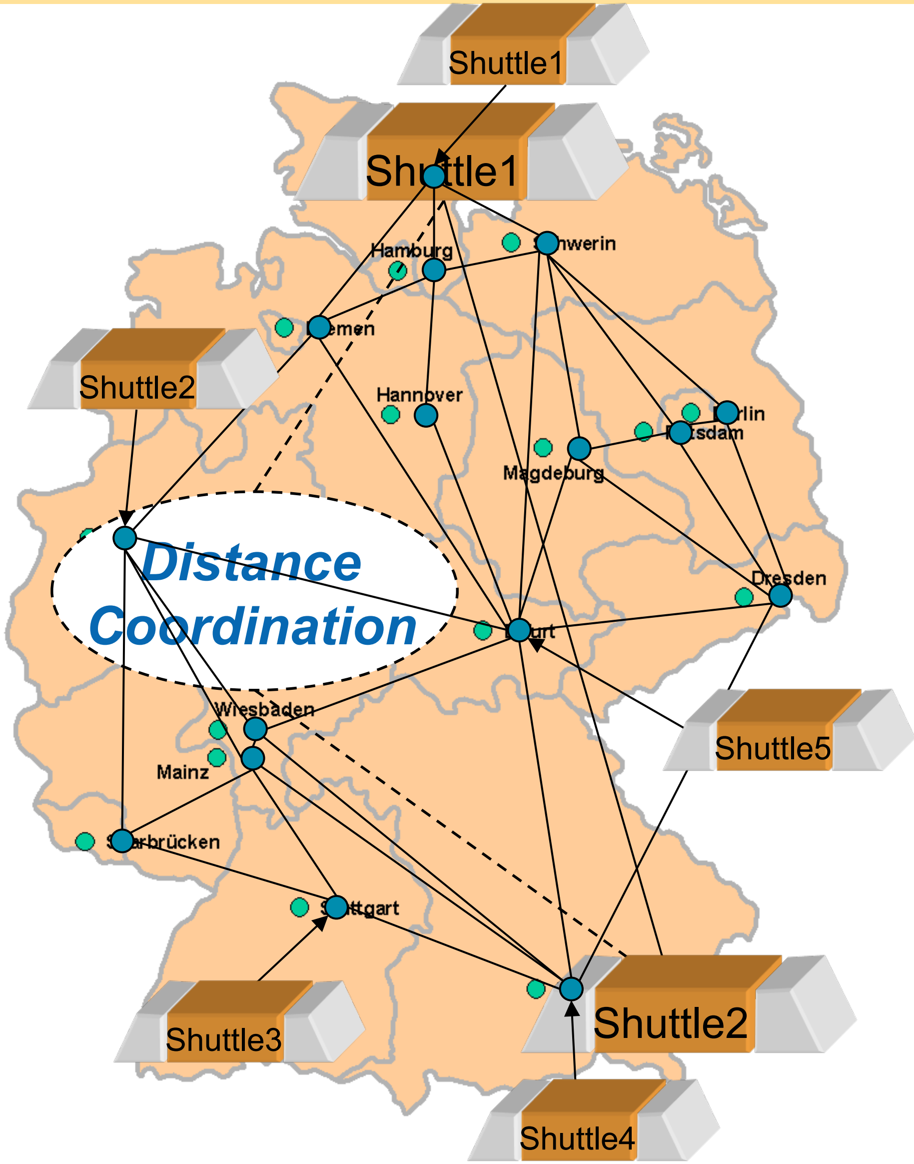
Components (Shuttles):

- Model: Statecharts for ports (refined roles) and synchronization
- Specification: local OCL constraints

Elements:

- Components
- Ports
- Connectors
- Patterns
- Roles

Rule-Based Reconfiguration (1/2)



Problem:

- Shuttles move and create resp. delete Distance Coordination patterns
- Arbitrary large topologies with moving shuttles

Solution:

- State = Graph
- Reconfiguration rules = graph transformation rules
- Safety properties = forbidden graphs

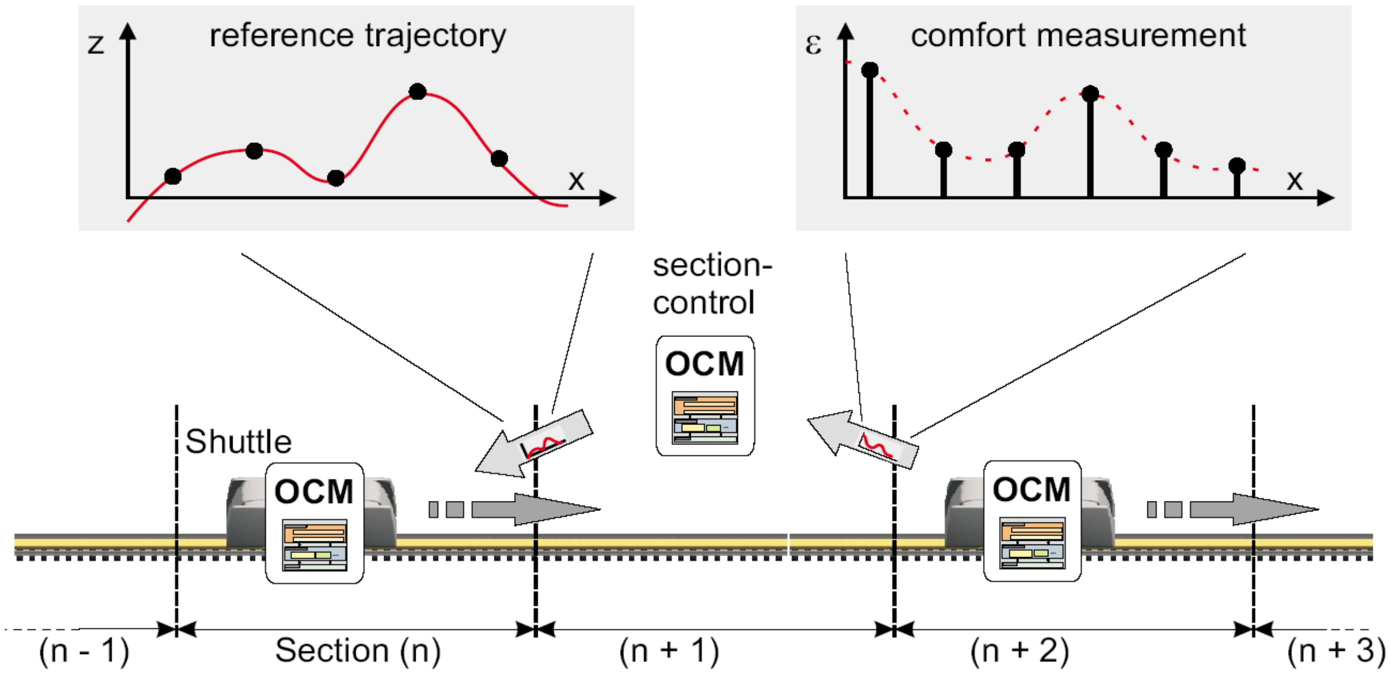
⇨ Formal Verification possible

Application Example: Self-Coordination

- **Cognitive Operators:** do self-optimization
  - Maneuver planning
  - Convoy planning
  - Shuttle planning
- **Reflective Operator:** switch to guarantee safety
  - Realize maneuvers planned by the cognitive operator(s)
  - Recognize timeouts and enforced related safety maneuvers
  - Detect problems of controllers and enforced related safety maneuvers

Application Example: Self-Optimization

[STTT2008]



- **Cognitive Operators:** do distributed self-optimization
  - Distributed learning of a model of the track (environment)
  - Local learning of a model of the shuttle (system hardware)
  - Planning an adaptation in form of an optimal trajectory
- **Reflective Operator:** switch to robust local control if necessary

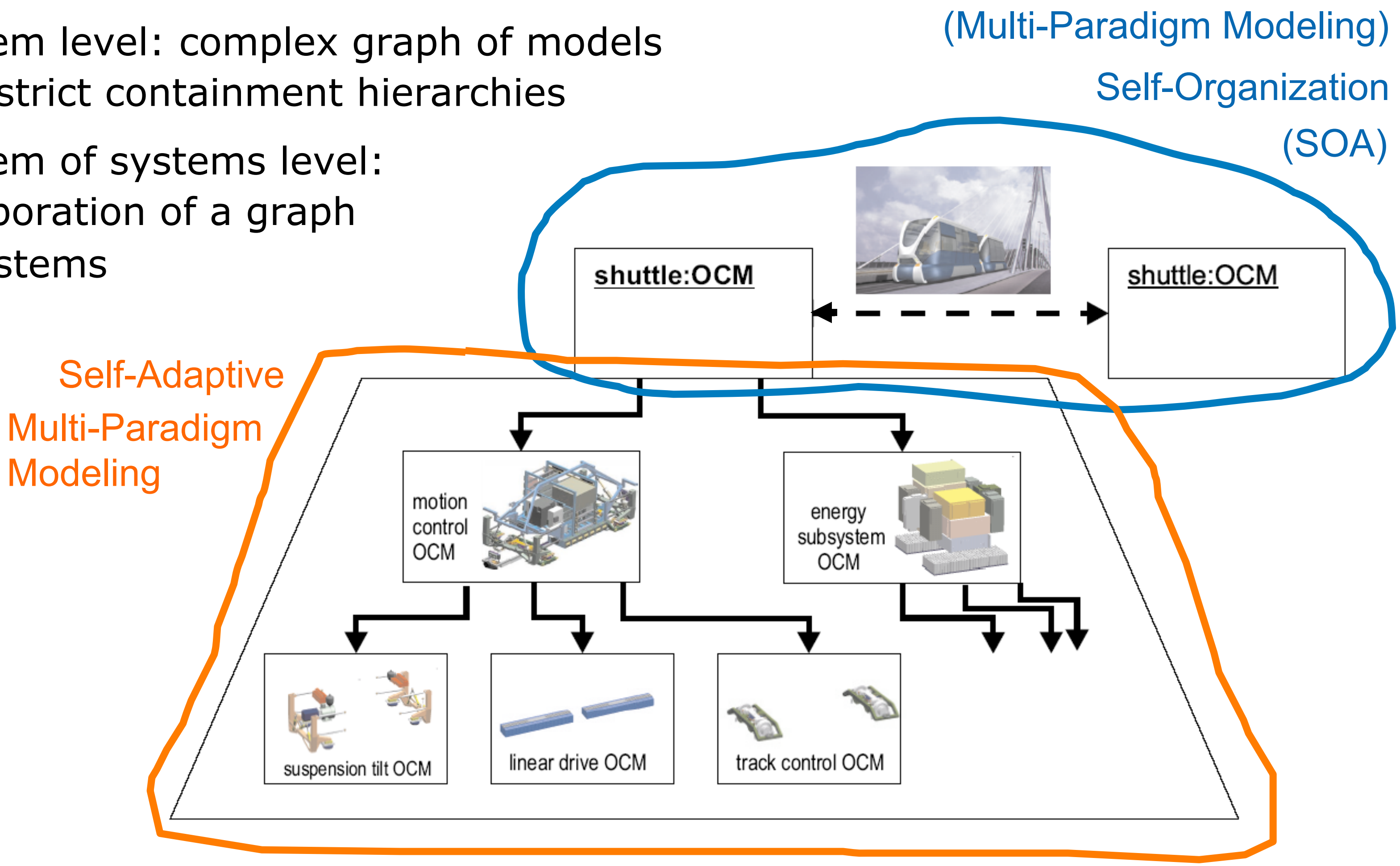
Outline

1. **Challenges Ahead**
2. **Available Options**
3. **Example: Mechatronic UML**
4. **Conclusions & Outlook**

mUML Example: Challenges & Options

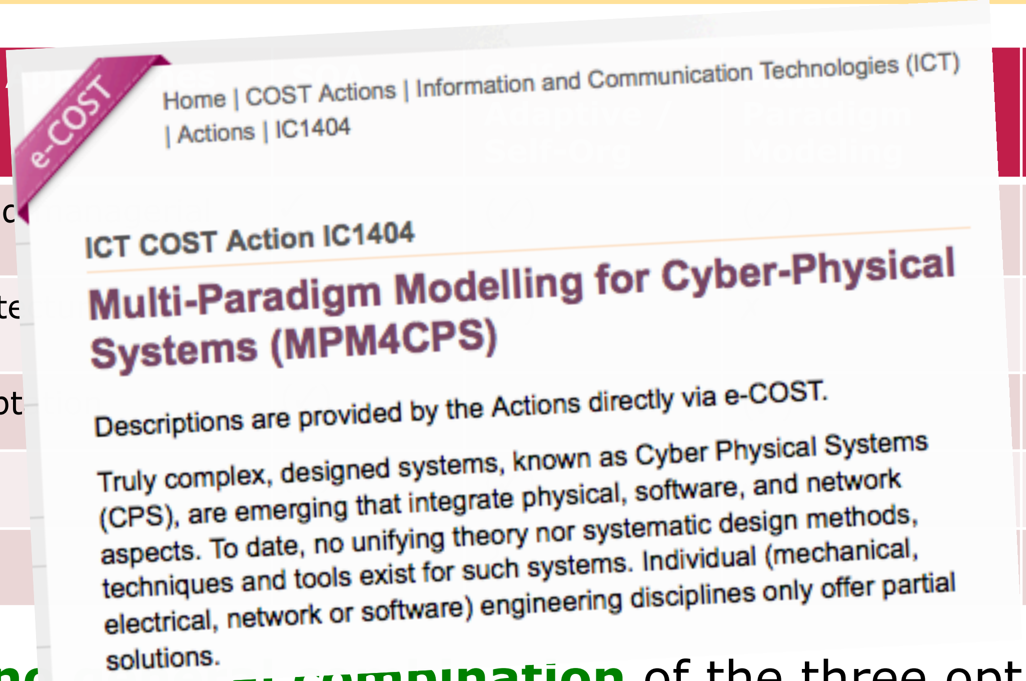
MechatronicUML:

- System level: complex graph of models with strict containment hierarchies
- System of systems level: collaboration of a graph of systems



Some Conclusions Concerning the Options

Challenges \	Mechatronic UML
Operational and independence	(✓)
Dynamic archite openness	✓
Advanced adapt	✓
Integration	✓
Resilience	✓


e-COST

Home | COST Actions | Information and Communication Technologies (ICT) | Actions | IC1404

ICT COST Action IC1404

Multi-Paradigm Modelling for Cyber-Physical Systems (MPM4CPS)

Descriptions are provided by the Actions directly via e-COST.

Truly complex, designed systems, known as Cyber Physical Systems (CPS), are emerging that integrate physical, software, and network aspects. To date, no unifying theory nor systematic design methods, techniques and tools exist for such systems. Individual (mechanical, electrical, network or software) engineering disciplines only offer partial solutions.

- However, no general combination of the three options exists

For the **MECHATRONICUML** approach we had to develop tool support
- that integrates existing tools in a particular way [Burmester+2008],
- that allows simulating the highly dynamic models [Giese+2011], and
- that ensure the safety at the system-of-systems level [Giese&Schäfer2013].

Problem:
- The high effort is required for each specific domain (limited coverage!)

Bibliography

Hasso Plattner Institut

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