Hasso Plattner Institut

IT Systems Engineering | Universität Potsdam

Formal Models and Analysis for Self-Adaptive Cyber-Physical Systems

International Conference on Formal Aspects of Component Software, Besançon, France, 19th October 2016.

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

Outline

1. Needs & Self-Adaptive CPS
2. Available Options
3. Challenges for Formal Models
4. Challenges for Formal Analysis
5. Conclusions & Outlook

Outline

Needs & Self-Adaptive CPS

- Cyber-Physical Systems
- System of Systems
- Ultra-Large-Scale Systems
- …

2. Available Options

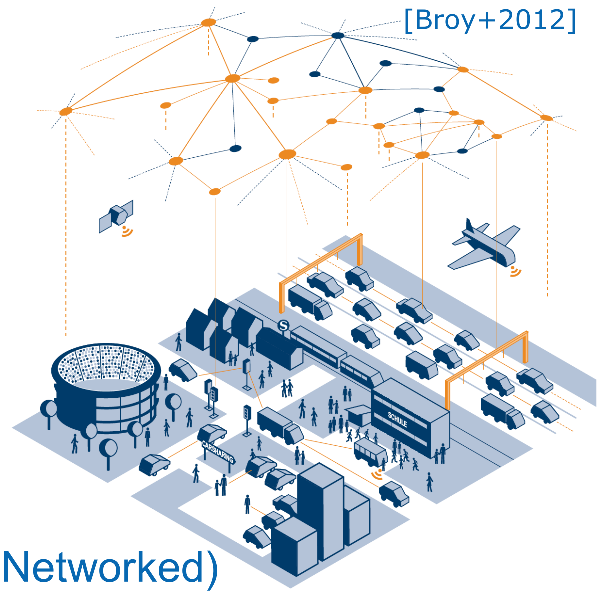
3. Challenges for Formal Models

4. Challenges for Formal Analysis

5. Conclusions & Outlook

The Future: You name it …

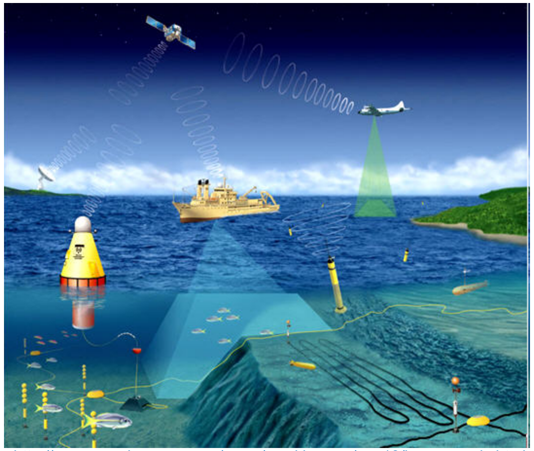
HPI Hasso Plattner Institut



[Broy+2012]

(Networked) Cyber-Physical Systems

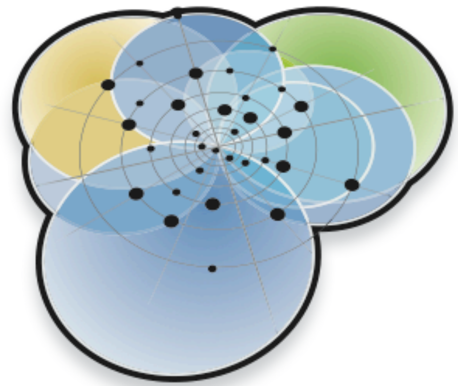
Internet of Things



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

System of Systems

[Northrop+2006]

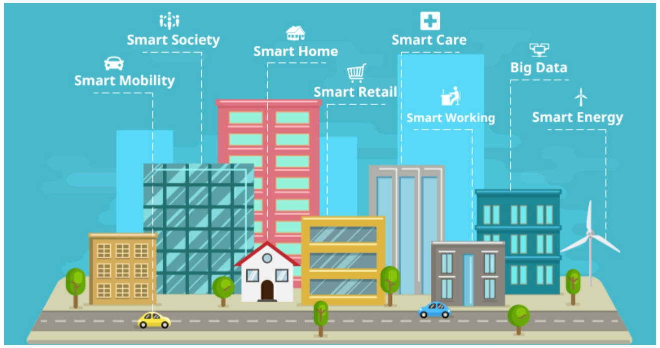


Ultra-Large-Scale Systems

Smart Factory - E.g. Industry 4.0

Smart Logistic

Micro Grids



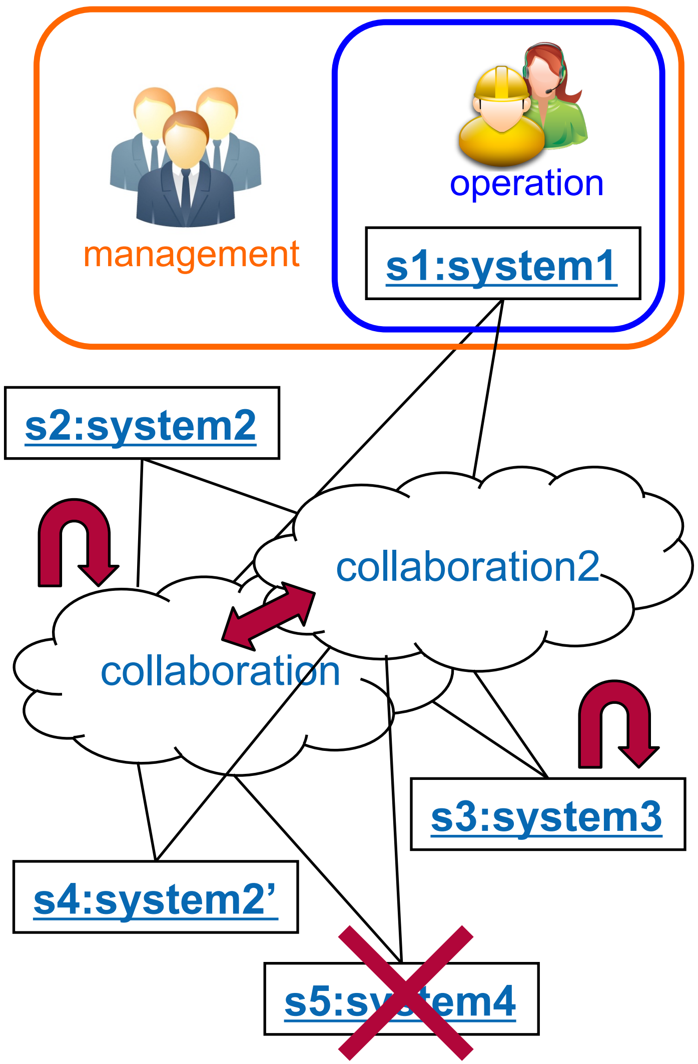
Smart City

E-Health

Smart Home

Ambient Assisted Living

Resulting Needs



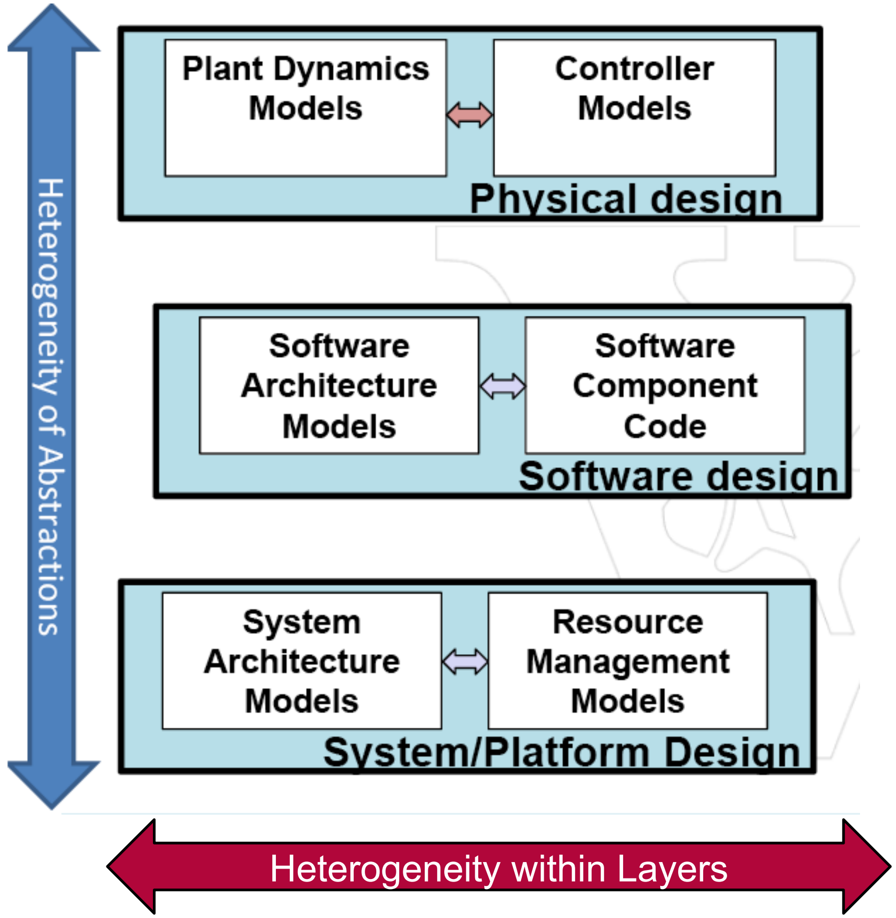
- **Operational** and **managerial independence**
  - operated independent from each other without global coordination
  - no centralized management decisions (possibly confliction decisions)
- **Dynamic architecture** and **openness**
  - must be able to dynamically adapt/absorb structural deviations
  - subsystems may join or leave over time in a not pre-planned manner of
- **Scale** for local systems or networked resp. large-scale systems of systems
- **Integration** of the physical, cyber, (and social) dimension
- **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

Need: Integration

[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



Need: 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: 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]

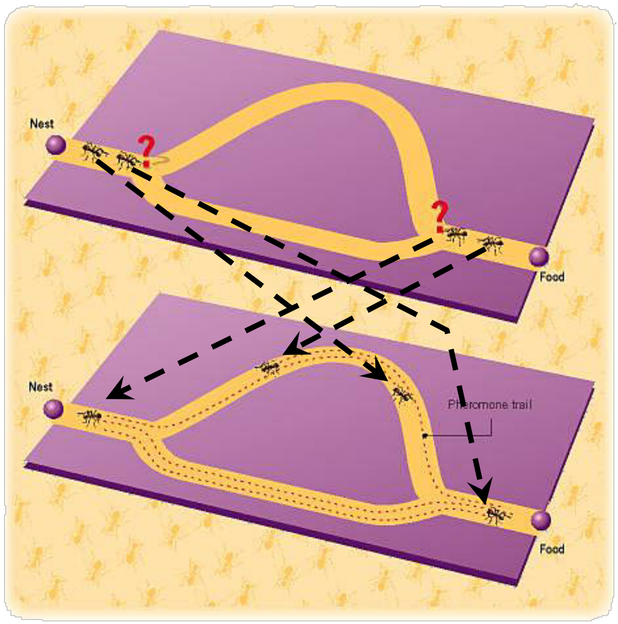
"Resilience is the capability of a system with specific characteristics before, during and after a disruption to absorb the disruption, recover to an acceptable level of performance, and sustain that level for an acceptable period of time." Resilient Systems Working Group, INCOSE

Required coverage of resilience:

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

Let’s have a look at Nature …

HPI Hasso Plattner Institut



Ant colonies operate as a **superorganism** that combines information processing of many ants and their interaction with the environment at the physical level (using stigmergy as coordination mechanism).

Example:

- Asymmetric binary bridge experiment

Observations:

- Initially both options will be taken with the same probability.
- The concentration of the pheromones will increase faster on the shorter path.
- The higher concentration of pheromones on the shorter path will make it more likely that an ant choses this shorter one.
- Positive feedback will amplify this effect and thus finally the longer path will only be used seldom.

➜ Can our problems be solved by borrow ideas from nature?

Let's have a **second** look at Nature ...

Another Example:

- "Ant Mill"

Observations:

- Such a behavior would be not acceptable for an engineered system even if they are confronted with **unexpected circumstances (rare events)**.
- If even "Nature" come up with designed solutions that fail (even evolution selected for ages), how could we envision to be more successful?
- But there is also a solution in nature:

  reflection/**adaptation** on itself (**self-awareness**)

Need for Self-Adaptive Cyber-Physical Systems

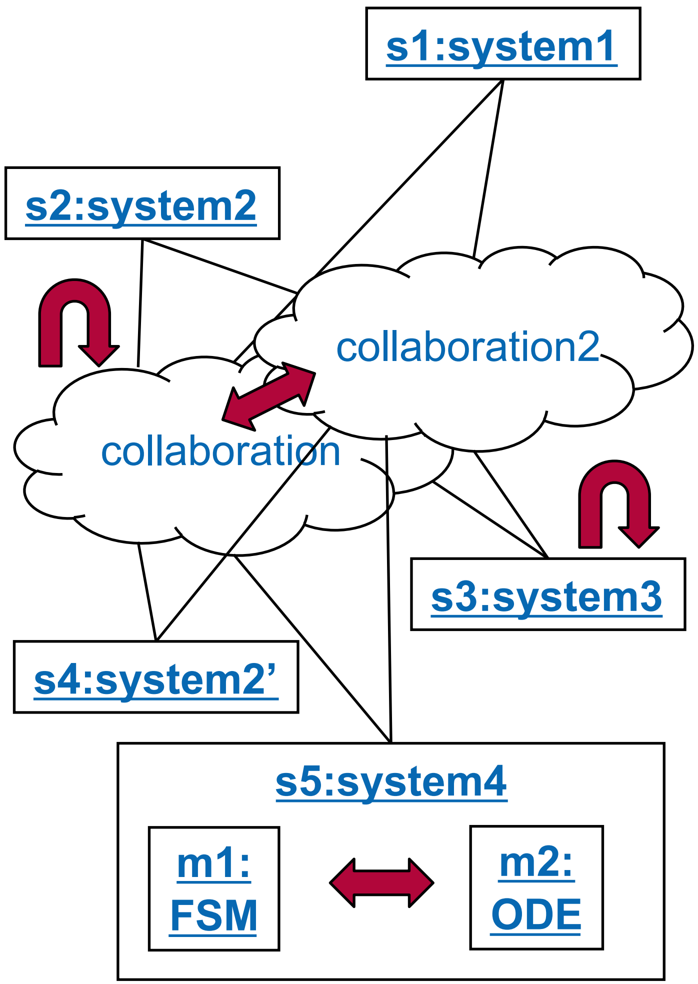
➔ Often CPS **requires** the capability of **self-awareness** to be able to handle problems due to unexpected circumstances

- Models must be able to evolve (**runtime models**)
- Systems must **reflect** on itself (**self-aware** of goals)
- Systems must **adapt**/self-adapt/learn

➔ We need **Self-Adaptive Cyber-Physical Systems**

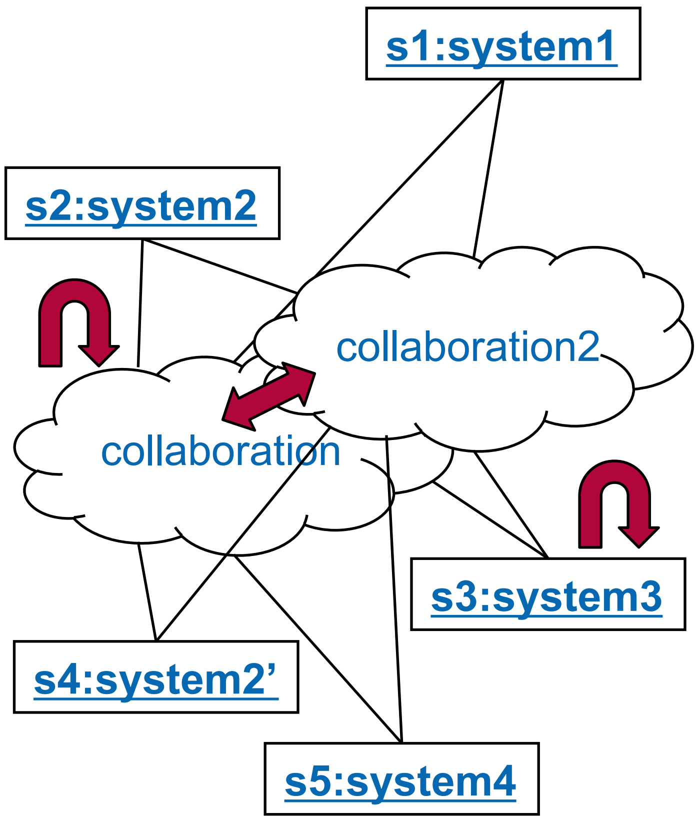
Outline

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: Self-Adaptive & Self-Organization

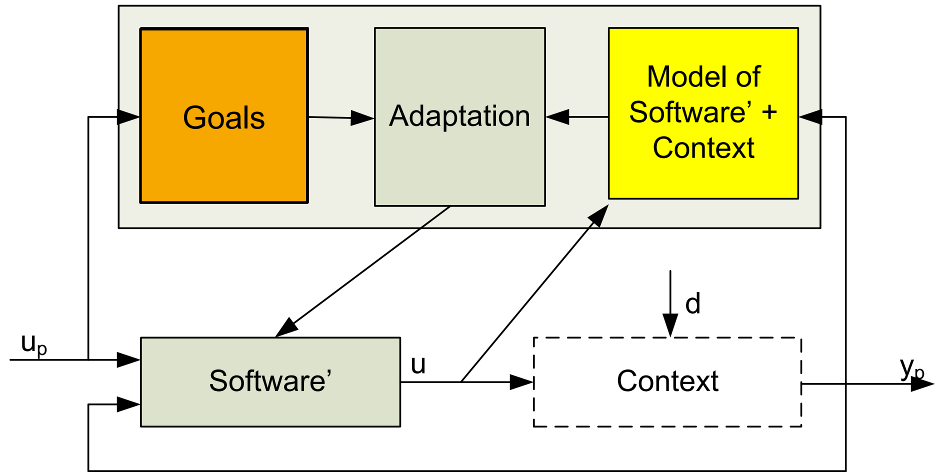


- **Self-Adaptive Systems:**
  - Make systems self-aware, context-aware, and requirements-aware using some form of **reflection**
  - Enable systems to adjust their structure/behavior accordingly
- **Self-Organization:**
  - The capability of a group of systems to organize their structure/behavior without 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: Runtime Models

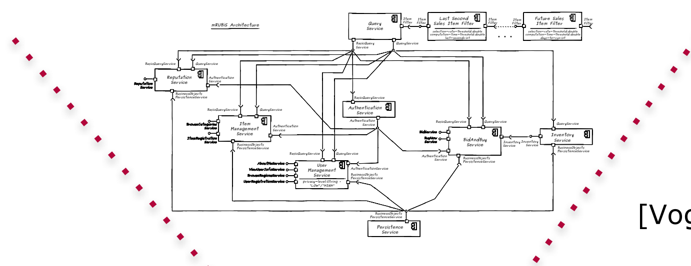
Runtime models:

- A **causal relation** between the software and/or context and the runtime model
- Self-Adaptation can operate at a higher level of abstraction

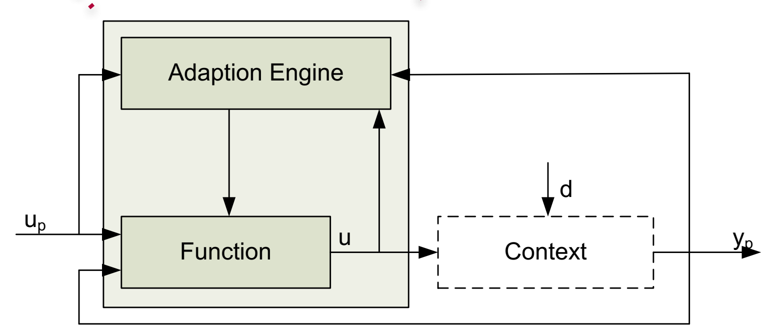


Observation:

- Generic **runtime models** can capture many possible changes
- Adaptation adjust the **Software'** according to the **Goals**

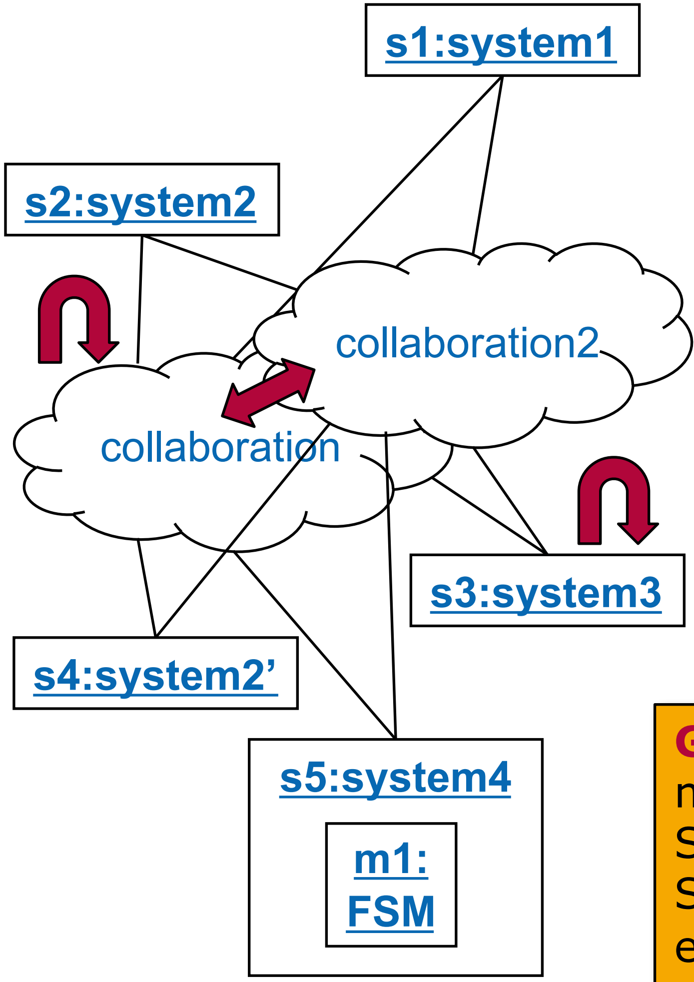


[Vogel&Giese2012]



Outline

Some Observations Concerning the Options



[Giese+2015]

- **Service-Oriented Architecture** can be described by a **graph** of links between the systems that evolve
- **Self-Adaptive** and **Self-Organization** can be described by a **graph** of links between the components resp. systems that evolve/reconfigure and in case of **reflection** most models can be described by such a **graph** as well
- **Runtime Models** can be described by a dynamic **graph** of models and links between them

Graph transformation systems encoding models and their linking would allow to combine Service-Oriented Architecture, Self-Adaptive / Self-Organization, and Runtime Models with evolving structures and **could be** the basis for a **solid foundation…**

Railcab System: Example Overview

A system of **autonomous shuttles** that operate on demand and in a decentralized manner using a **wireless network**.

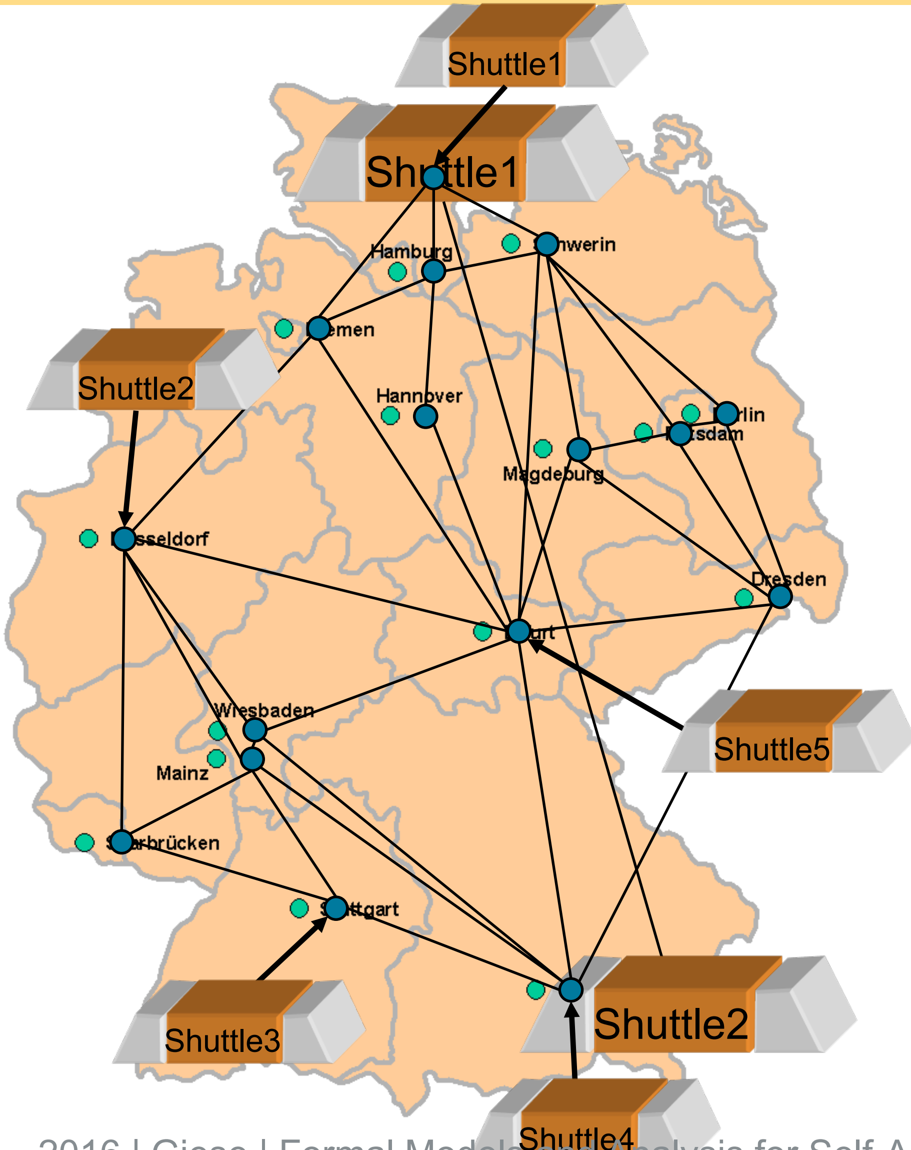
Self-Adaptive CPS:

- Hard real-time
- Safety-critical
- Self-Optimization

Needs:

- Optimized maneuvers, operation, and resource utilization (e.g., convoy)

Related Observation Concerning the Example



Modeling Problems:

- Shuttles move on a topology of tracks
- Arbitrary large topologies

Solution:

- State = Graph
- Reconfiguration rules = graph transformation rules
- Safety properties = forbidden graphs
- ⇨ Formal Verification possible

Very strong reduction: not all properties are represented

- Dynamic convoy structures and movement of the shuttles on the topology of tracks
- Real-Time movement of the shuttles on the topology of tracks
- Real-Time protocols for convoy coordination
- Continuous driving behavior
- Random communication errors
- …

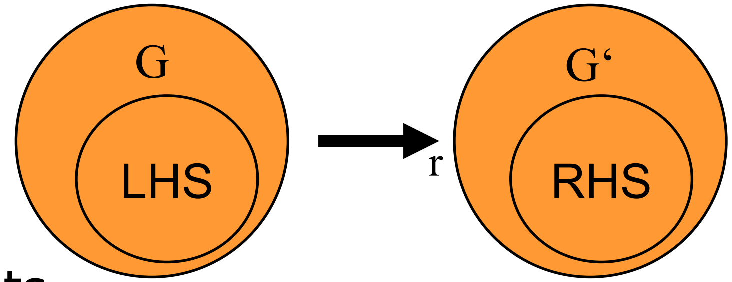
} non-functional

Graph Transformation System: Definition

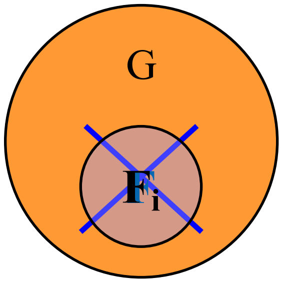
A **graph transformation system** (we omit here NACs) consists of

- a type graph describing all possible model configurations,
- a set of rules R with LHS and RHS, and
- a function $prio\colon R \rightarrow Int$ which assigns priorities to all rules.

We also use a set of **forbidden graph patterns** F for unsafe situations.

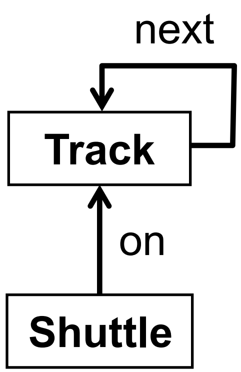


- A rule r of R is **enabled** if an occurrence of its LHS in a graph G exists.
- A rule r of R is **applied** on graph G by replacing an occurrence of its LHS in G by the RHS (DPO).
- A forbidden graph pattern $\mathbf{F_i}$ in F is **respected** by a graph G if it is not contained.

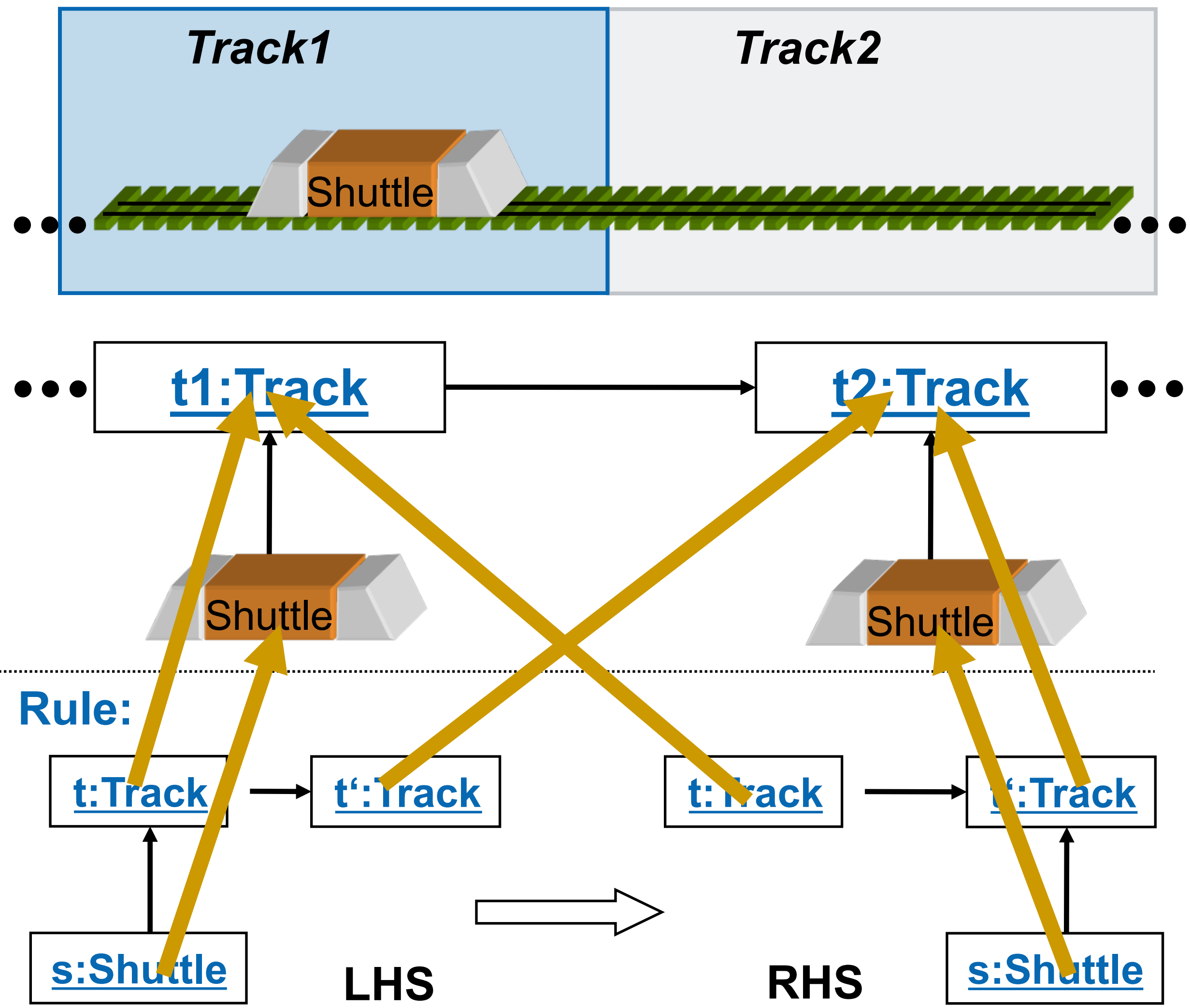


Graph Transformation Systems: Naïve Example

- Map the tracks
- Map the shuttles

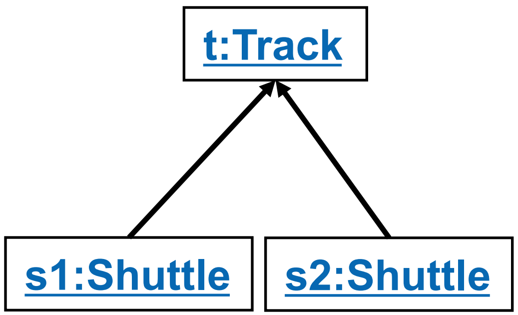


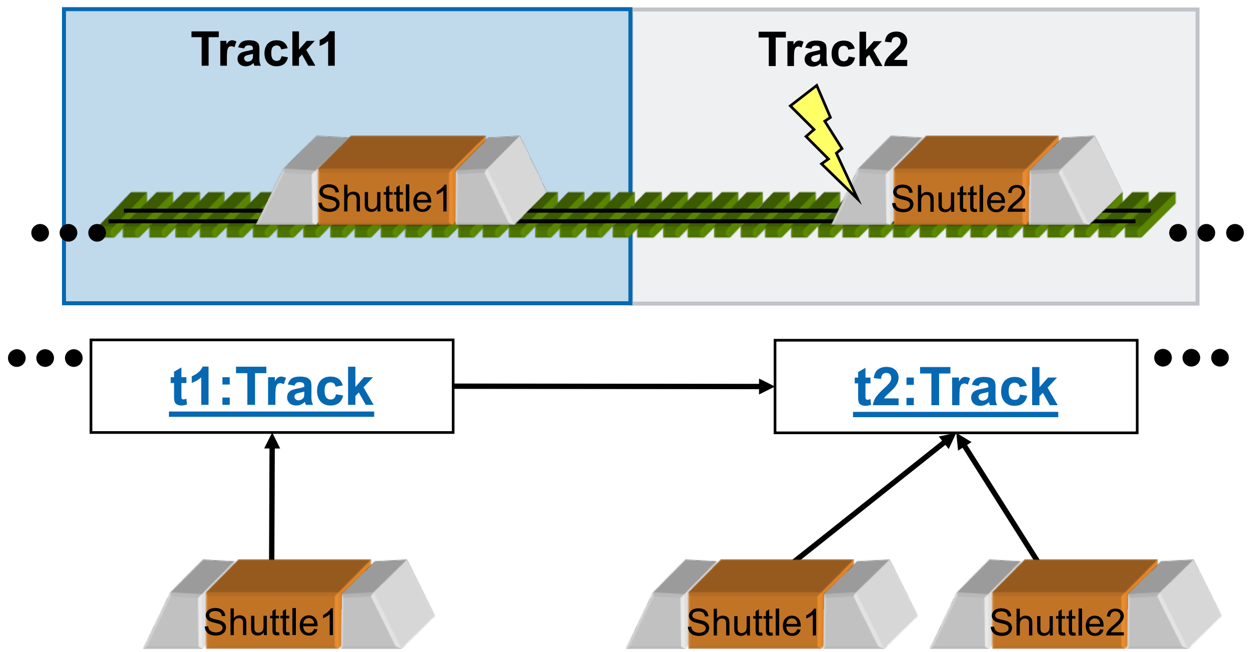
- Map the movement to rules (movement equals dynamic structural adaptation on the abstract level)



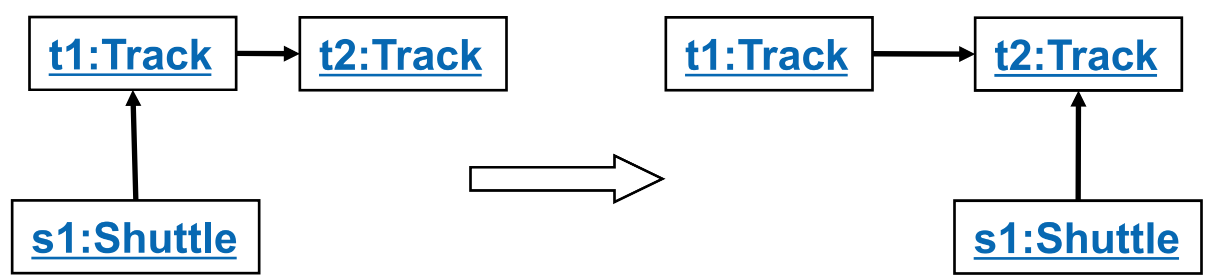
Graph Transformation Systems: Naïve Example

Forbidden Graph





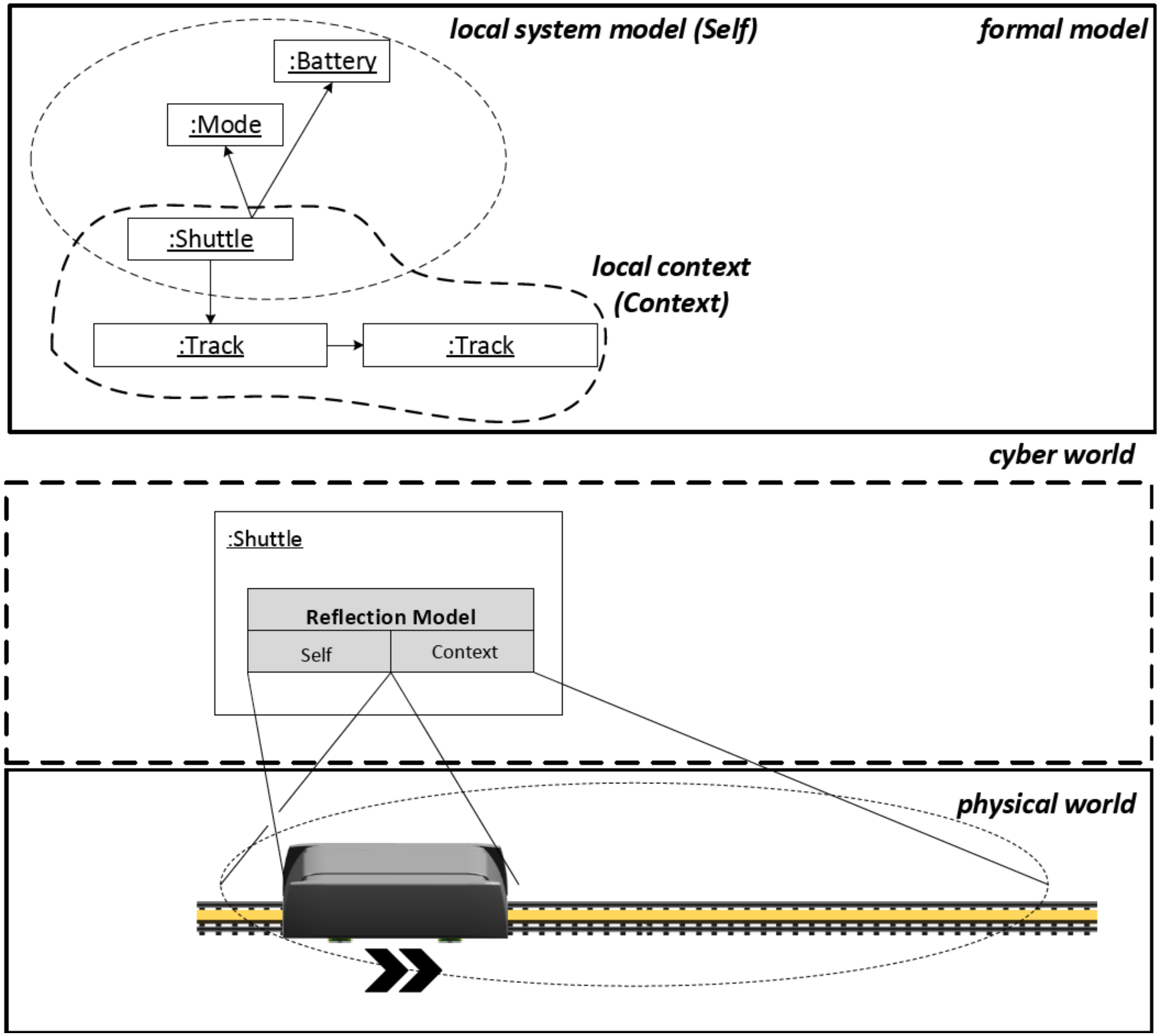
Rule:



SMARTSOS: Main Idea

Hasso Plattner Institut

[Giese+2015]



Use

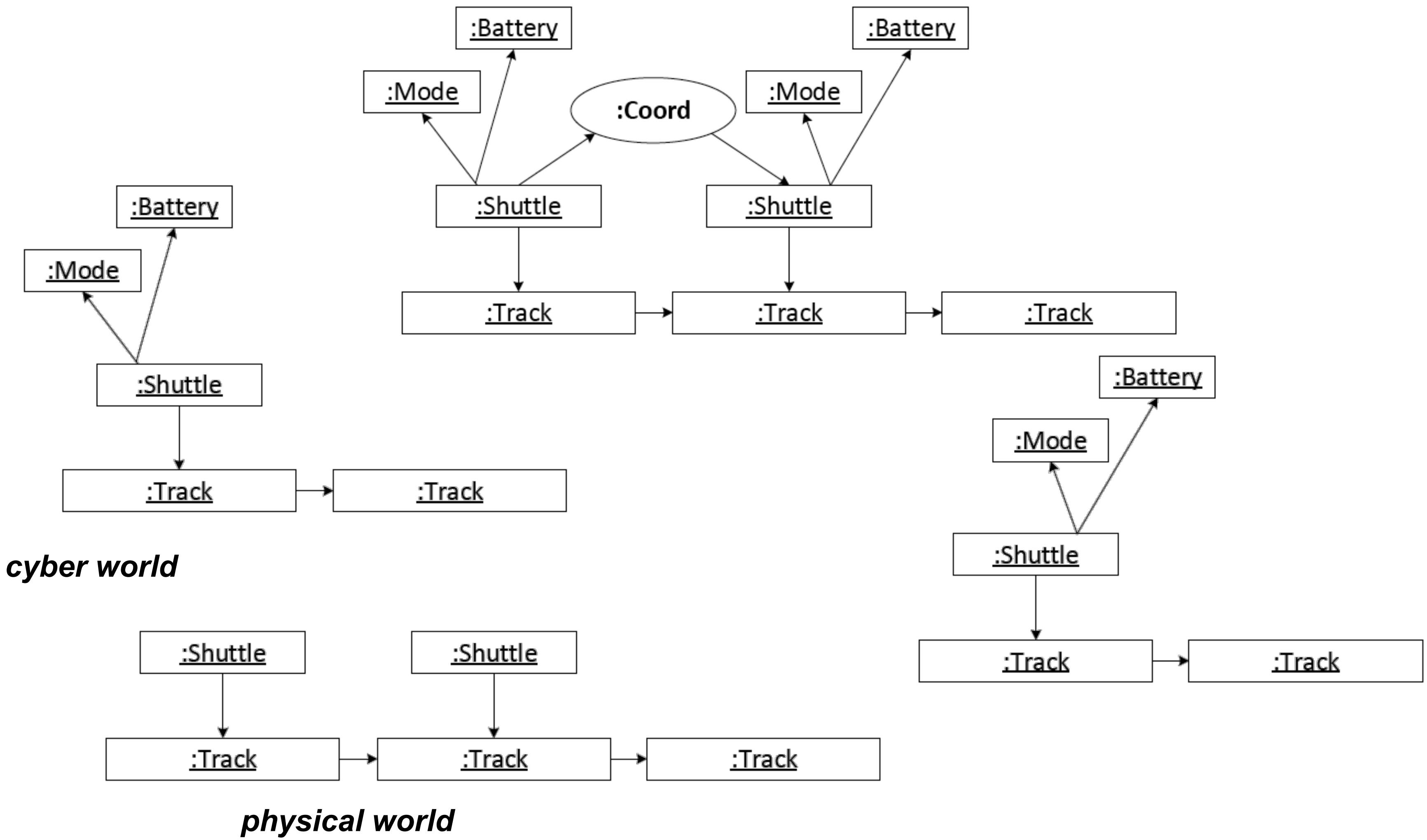
- a **graph** of links between the systems, components, and internal represented data as well as
- **graph transfor-mations** to capture possible changes

to model

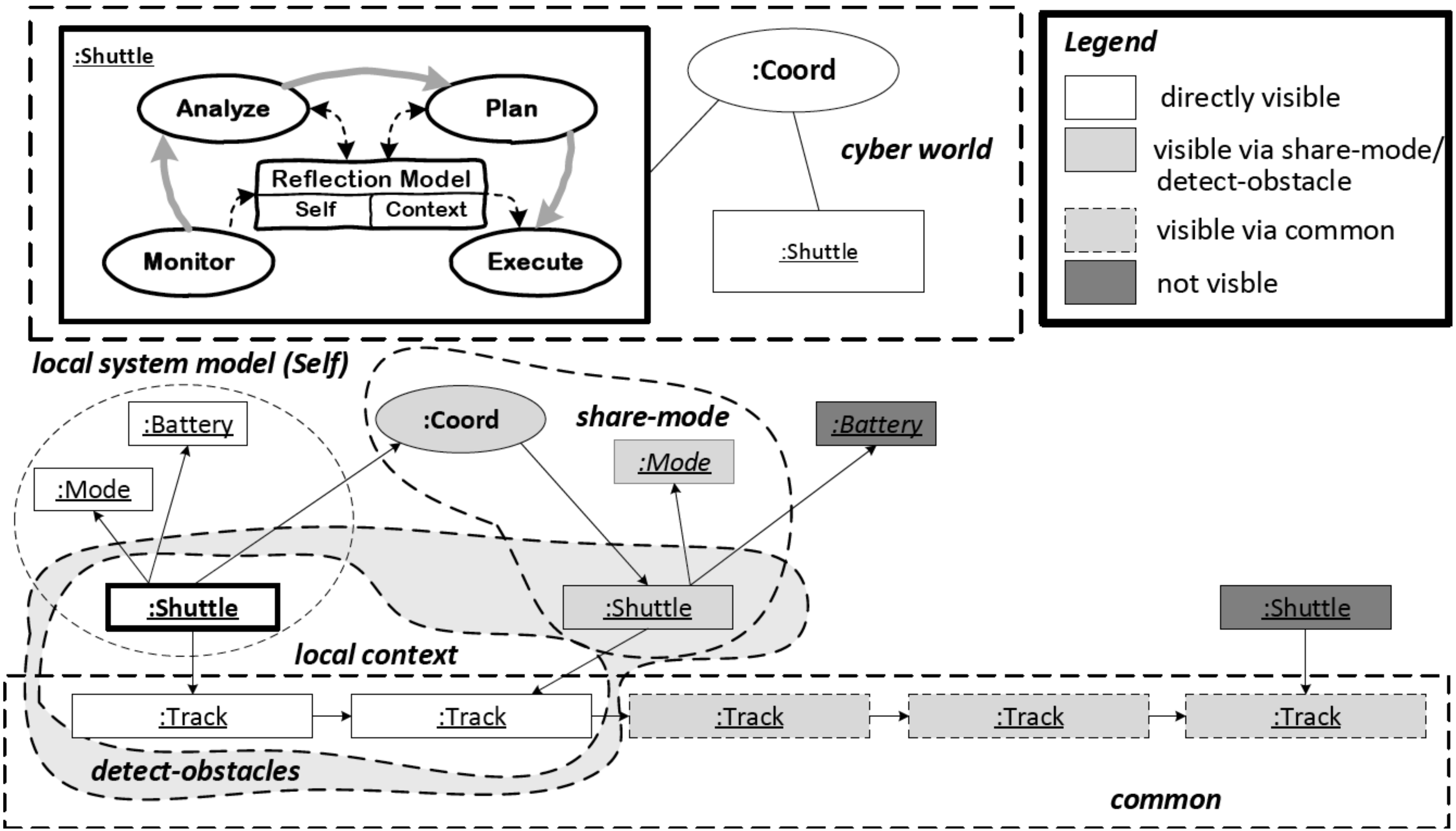
- **Service-Oriented Architecture**,
- **Self-Adaptive** and **Self-Organization**, and
- **Runtime Models**

Consistency of Cyber & Physical World

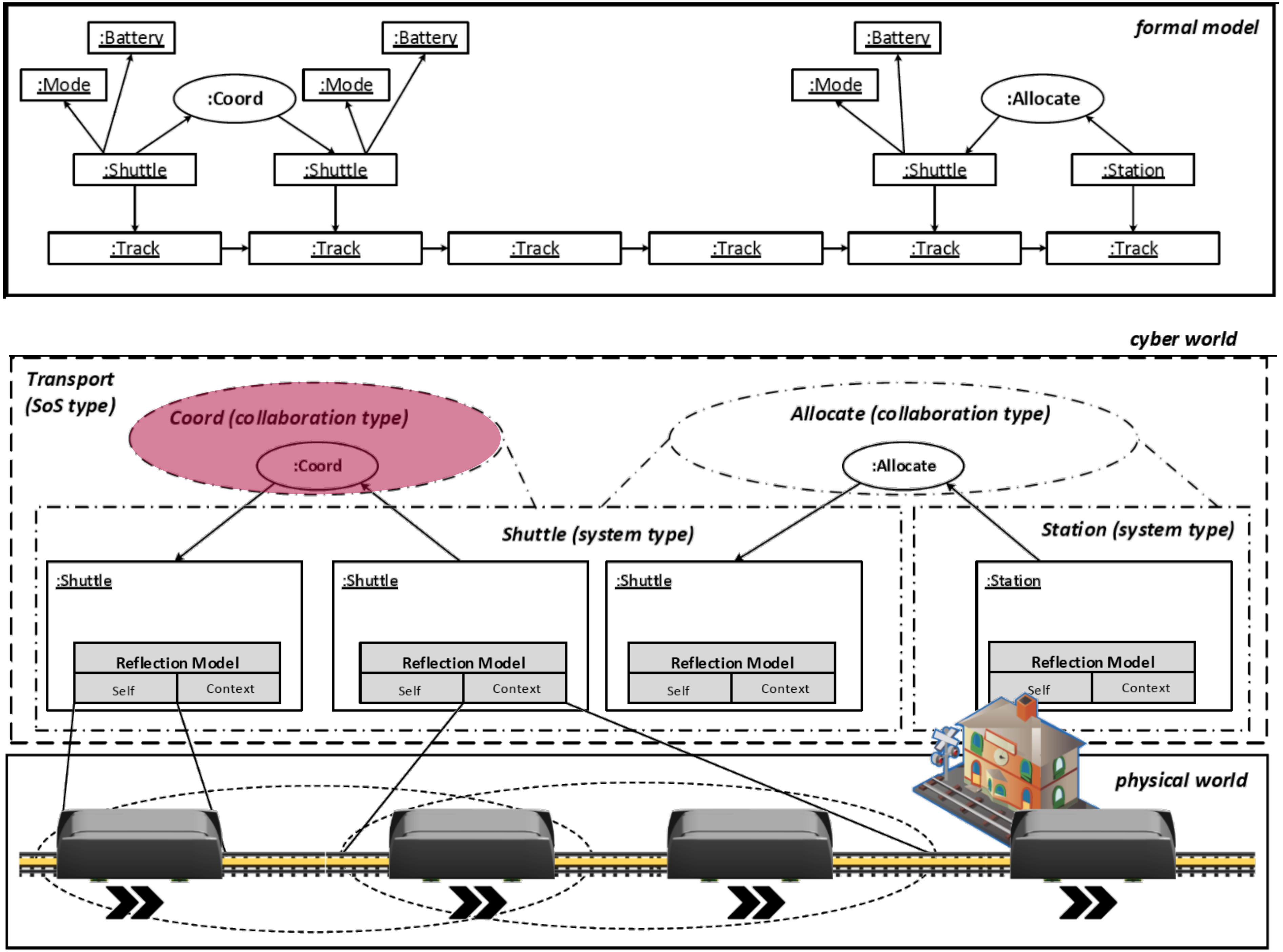
[Giese+2015]



Sharing Runtime Models & Visibility

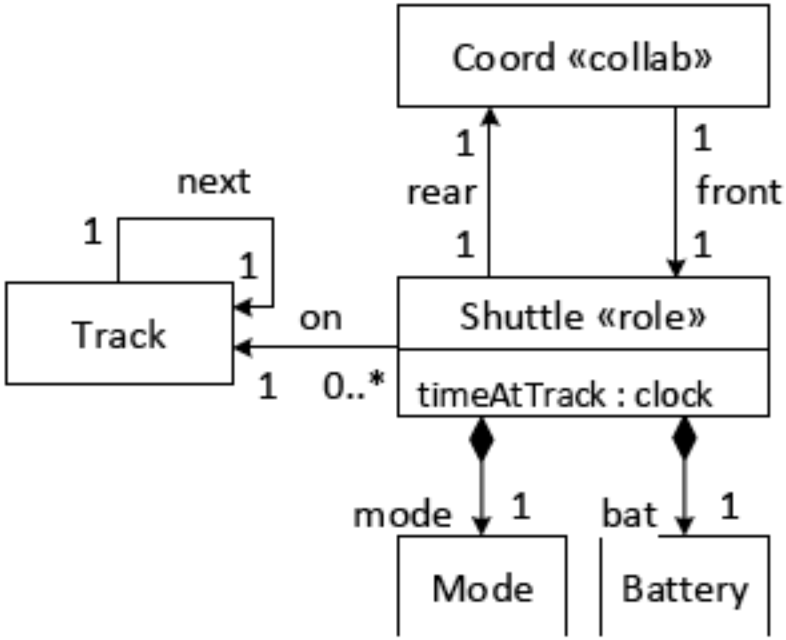


[Giese+2015]


formal model
:Battery
:Battery
:Battery
:Mode
:Coord
:Mode
:Mode
:Allocate
:Shuttle
:Shuttle
:Shuttle
:Station
:Track
:Track
:Track
:Track
:Track
:Track
cyber world
Transport (SoS type)
Coord (collaboration type)
:Coord
Allocate (collaboration type)
:Allocate
Shuttle (system type)
Station (system type)
:Shuttle
:Shuttle
:Shuttle
:Station
Reflection Model
Self
Context
Reflection Model
Self
Context
Reflection Model
Self
Context
Reflection Model
Self
Context
physical world

SMARTSOS: Collaboration Types

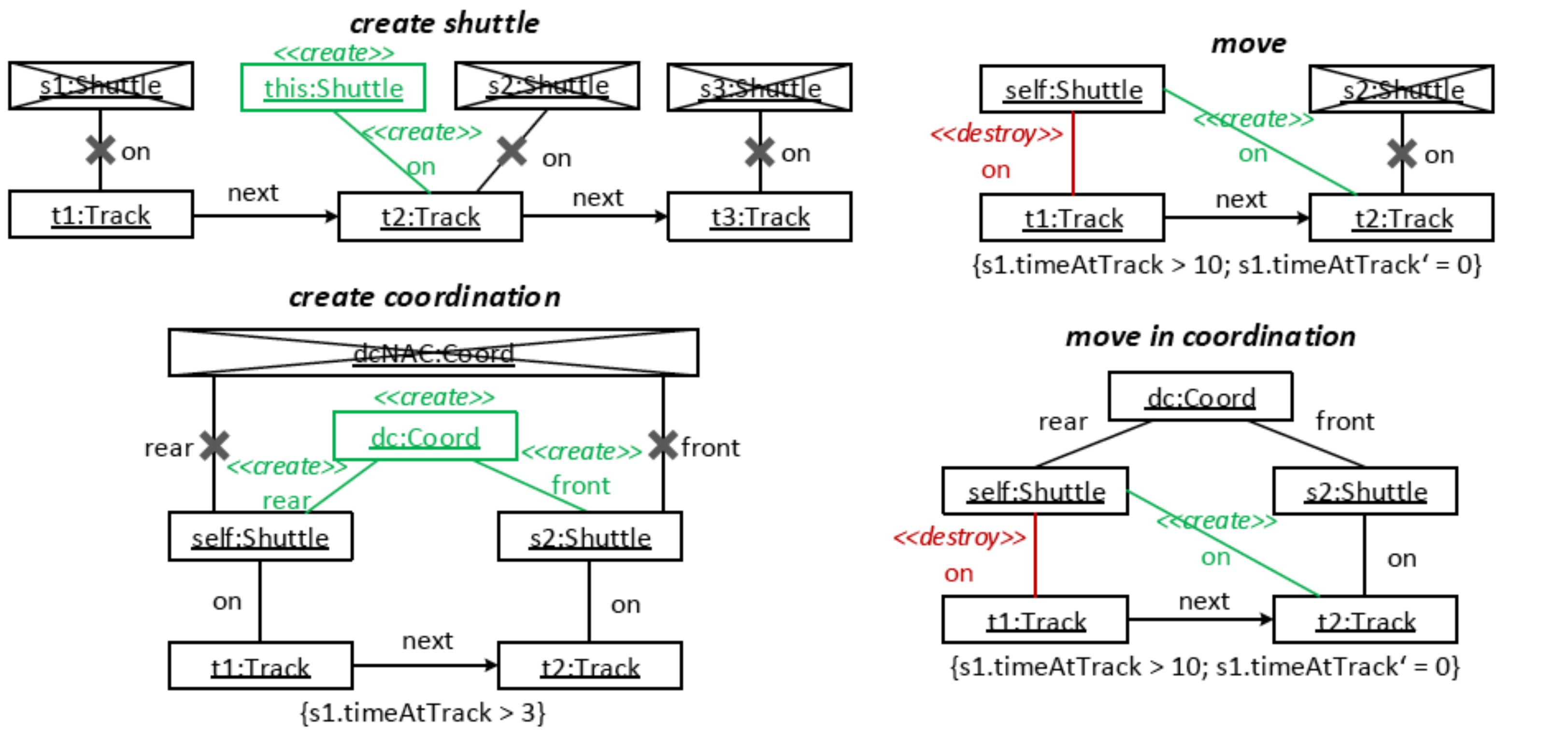
Definition 2 (see [54]). *A* collaboration type $\mathsf{Col}_i = (\mathsf{col}_i, (ro_i^1, \ldots, ro_i^{n_i}), CD_i, R_i, \Phi_i)$ *consists of a collaboration type node* col_i*, a number of role types* ro_i^j*, an UML class diagram* CD_i*, a function* $R_i : \{\mathsf{col}_i, ro_i^1, \ldots, ro_i^{n_i}\} \mapsto 2^{\mathcal{R}}$ *assigning rules to role types, and a guaranteed property* Φ_i*.*



[Giese+2015]

SMARTSOS: Collaboration Types

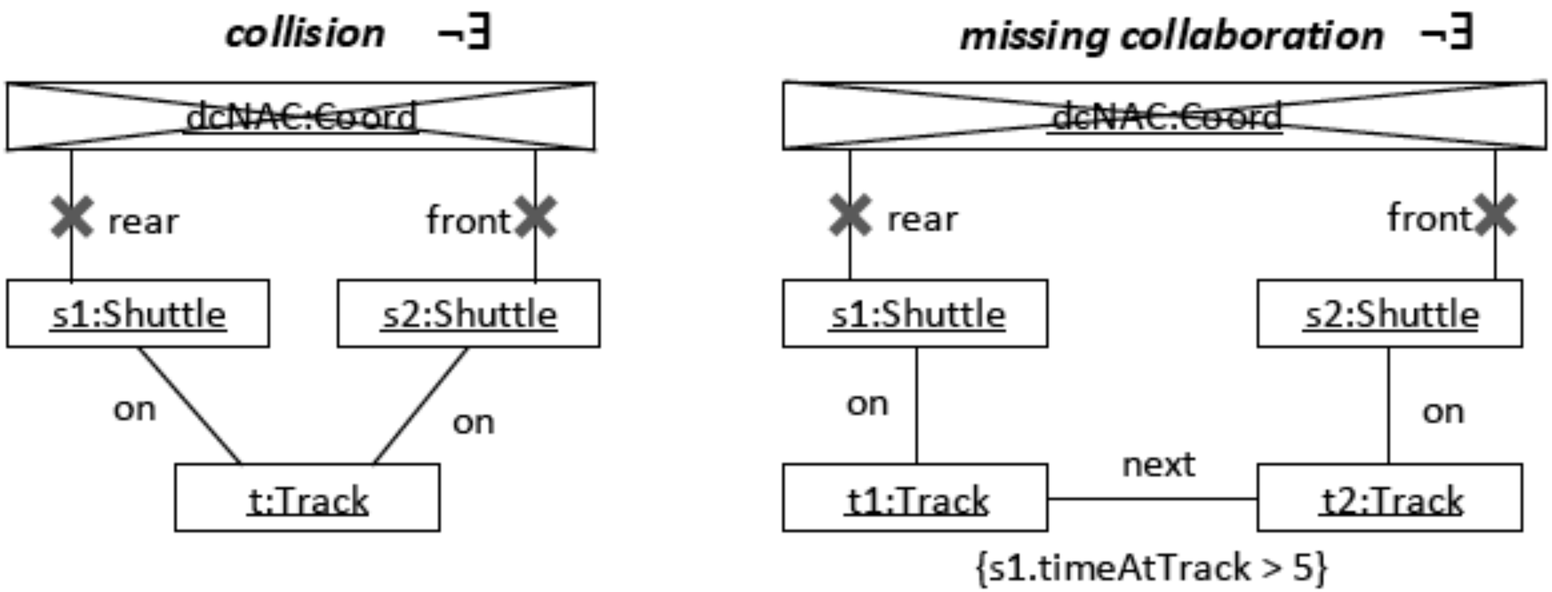
Definition 2 **(see [54]).** *A collaboration type* $\mathsf{Col}_i = (\mathsf{col}_i, (ro_i^1, \ldots, ro_i^{n_i}), CD_i, R_i, \Phi_i)$ *consists of a collaboration type node* col_i*, a number of role types* ro_i^j*, an UML class diagram* CD_i*, a function* $R_i : \{\mathsf{col}_i, ro_i^1, \ldots, ro_i^{n_i}\} \mapsto 2^{\mathcal{R}}$ *assigning rules to role types, and a guaranteed property* Φ_i*.*



[Giese+2015]

SMARTSOS: Collaboration Types

Definition 2 (see [54]). *A collaboration type* $\mathsf{Col}_i = (\mathsf{col}_i, (ro_i^1, \ldots, ro_i^{n_i}), CD_i, R_i, \Phi_i)$ *consists of a collaboration type node* col_i*, a number of role types* ro_i^j*, an UML class diagram* CD_i*, a function* $R_i : \{\mathsf{col}_i, ro_i^1, \ldots, ro_i^{n_i}\} \mapsto 2^{\mathcal{R}}$ *assigning rules to role types, and a guaranteed property* Φ_i*.*



[Giese+2015]

SMARTSOS: Collaboration Types

Hasso Plattner Institut

Definition 2 **(see [54]).** *A collaboration type* $\mathsf{Col}_i = (\mathsf{col}_i, (ro_i^1, \ldots, ro_i^{n_i}), CD_i, R_i, \Phi_i)$ *consists of a collaboration type node* col_i*, a number of role types* ro_i^j*, an UML class diagram* CD_i*, a function* $R_i : \{\mathsf{col}_i, ro_i^1, \ldots, ro_i^{n_i}\} \mapsto 2^{\mathcal{R}}$ *assigning rules to role types, and a guaranteed property* Φ_i*.*

- The roles of the collaborations capture the permitted behavior:
  - Underspecification permits local decisions/self-adaptation. E.g.,
    - Non-determinism provide options for decisions
    - Time intervals allow to optimize timing via self-adaptation
- **Self-Organization** based on runtime models become possible:
  - Required properties must emerge from local rules
  - Context and runtime models can be employed as well (stigmergy, context-aware rules, …)

→ We support **SoS-Level Self-Organization**, **SoS-Level Structural Dynamics**, and **Runtime Knowledge Exchange**

[Giese+2015]

SMARTSOS: System Types

Definition 3 **(see [54]).** *A system type* $\mathsf{Sys}_i = (\mathsf{sys}_i, (ro_i^1, \ldots, ro_i^{m_i}), CD_i, R_i, I_i, \Psi_i)$ *consists of a system type node* sys_i*, a number of role types* ro_i^j*, a class diagram* CD_i*, a function* $R_i : \{\mathsf{sys}_i, ro_i^1, \ldots, ro_i^{m_i}\} \mapsto 2^{\mathcal{R}}$ *assigning rules to role types, a set of initial rules* $I_i \subseteq R_i(\mathsf{sys}_i)$*, and a safety property* Ψ_i*.*

- The system behavior has to respect the roles (of the collaborations):
  - All rules with side effects have to refine permitted behavior
  - All rules can access the elements visible via collaborations
- **Self-Adaptation** based on runtime models become possible:
  - Self: runtime model of the system itself
  - Local context: local context of the system
  - Shared context: runtime models of other systems

➔ We have enabled **Self-Adaptation** for the systems

[Giese+2015]

Requirements for Formal Models

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
- **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**

Requirements for Formal Models

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
- **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**

My Work:

- **SMARTSOS**
- **Timed GTS** ([Becker&Giese2008])
- **Hybrid GTS** ([Becker&Giese2012])
- **Probabilistic GTS** ([Krause&Giese2012])

BUT: We in fact would need a formal model that supports all required characteristics at once for **Self-Adaptive Cyber-Physical Systems**!

Outline

1. Needs & Self-Adaptive CPS
2. Available Options
3. Challenges for Formal Models
4. **Challenges for Formal Analysis**
5. Conclusions & Outlook

Requirements for Formal Analysis

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
- **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**

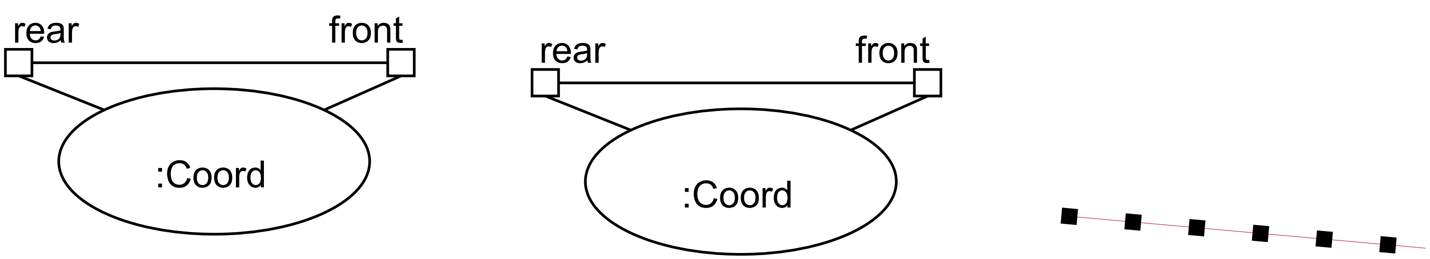
Analysis Required:

- **Complex state properties**
- **Complex sequence properties**
- **Even ensemble properties (like stability)**
- **Probabilistic sequence properties**

SMARTSOS: Correct Collaborations

[Giese+2015]

Definition 8 (see [54]). *A collaboration type* $\mathsf{Col}_i = (\mathsf{col}_i, (ro_i^1, \ldots, ro_i^{n_i}), CD_i, R_i, \Phi_i)$ *is* correct *if for all initial configurations* $G_I \in \mathcal{G}_\emptyset(CD_i)$ *holds that for* $R_i(\mathsf{Col}_i) = R_i(ro_i^1) \cup \cdots \cup R_i(ro_i^n) \cup R_i(\mathsf{col}_i))$ *the overall behavior of the collaboration the reachable collaboration configurations are correct:* $G_I, R_i(\mathsf{Col}_i) \models \Phi_i$.



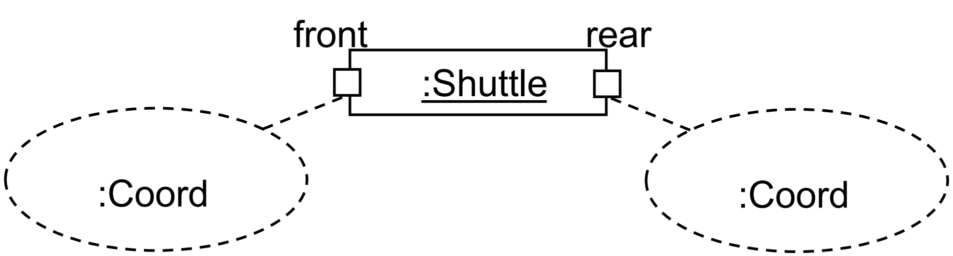
[Giese+2015]

SMARTSOS: Correct Systems

Definition 9 (see [54]). *A system type* $\mathsf{Sys}_i = (\mathsf{sys}_i, (ro_i^1, \ldots, ro_i^{m_i}), CD_i, I_i, \Psi_i)$ *is correct if for all initial configurations* $G_I \in \mathcal{G}_\emptyset(CD_i)$ *holds that the reachable configurations are correct* $G_I, R_i(\mathsf{sys}_i) \cup COMP(\mathsf{Sys}_i) \cup I_i \models \Psi_i \quad (1)$ *and the system behavior* $R_i(\mathsf{sys}_i)$ *refines the orthogonally combined role behavior and creation behavior*

$$R_i(\mathsf{sys}_i) \sqsubseteq R_i(ro_i^1) \cup \cdots \cup R_i(ro_i^{m_i}) \quad \cup \quad I_i. \qquad (2)$$

To add the collaboration behavior to the system behavior for each role without the role itself, we employ here $COMP(\mathsf{Sys}_i) = \bigcup_{1 \leq l \leq m_i} COMP(\mathsf{Sys}_i, ro_i^l)$ *with* $COMP(\mathsf{Sys}_i, ro_i^l) = R_j(\mathsf{Col}_j)$ *which is covered by* $R_i(\mathsf{sys}_i)$ *to derive a related closed behavior.*



[Giese+2015]

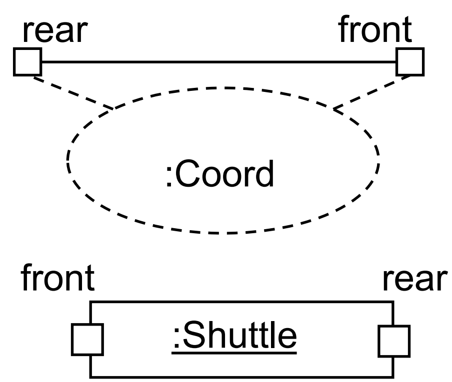
SMARTSOS: Scalable Correctness SoS

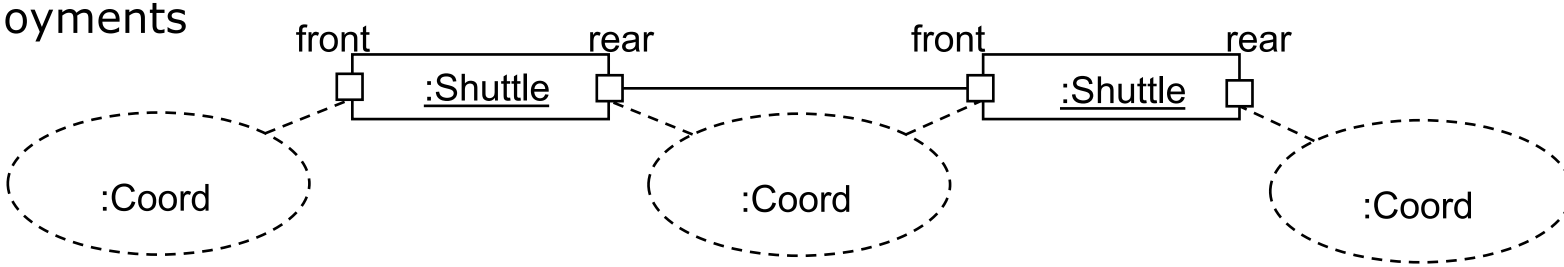
Hasso Plattner Institut

Theorem 1 ([54]). *A system of systems* $\mathsf{sos} = (\mathsf{SoS}, G_\emptyset)$ *with system of system type* $\mathsf{SoS} = ((\mathsf{Col}_1, \ldots, \mathsf{Col}_n), (\mathsf{Sys}_1, \ldots, \mathsf{Sys}_m))$ *is* correct *if (1) the system of system type* SoS *is* type conform, *(2) all collaboration types* $\mathsf{Col}_1, \ldots, \mathsf{Col}_n$ *are* correct, *and (3) all system types* $\mathsf{Sys}_1, \ldots, \mathsf{Sys}_m$ *are* correct.

Decompose verification:

- Verification guarantees properties for the collaborations (no collision)
- Verification guarantees conformance for systems (ports refine roles)
- **Compositional result:** Properties hold for all collaborations in correctly composed system deployments





➔ We have a first element for the **Resilience** of the SoS

[Giese+2015]

SMARTSOS: Correctness of a Collaboration

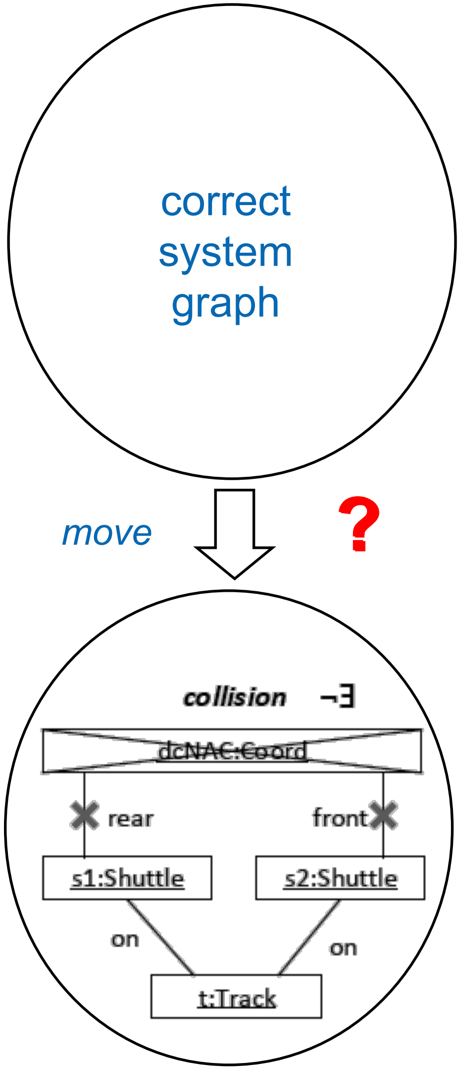
[Becker+2006, Becker&Giese2008]

Verification Problem:

- Infinite many initial states or reachable state are possible
- State and sequence properties would be of interest

Checking Options:

- Model Checking (mapping to GROOVE; only debugging)
  - Limited to small configurations and finite models
  - Extension for continuous time have been developed
- Invariant Checker for state properties (our development)
  - Analyze that changes can not lead from safe to unsafe situations (**inductive invariants**)
  - Supports infinite many start configurations specified only by their structural properties
  - Supports infinite state models
  - Extension of time and discrete variables exist
  - Incremental check for changed rules
  - Extension of hybrid behavior



Requirements for Formal Analysis

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
- **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**

SMARTSOS

State-of-the-Art & our Work:

- **Checking Inductive Invariants for GTS** ([Becker+2006]), **Timed GTS** ([Becker&Giese2008]), **and Hybrid GTS** ([Becker&Giese2012])

  Only state properties!

- **Model Checking Timed and Hybrid Systems**

  Only sequence properties for finite state systems with rather bad scalability!

- **Model Checking Probabilistic GTS** ([Krause&Giese2012])

  Only very restricted probabilistic sequence properties for finite state systems with bad scalability!

BUT: We have to assure resilience for complex sequence properties (even ensemble properties) of **hybrid probabilistic infinite state systems**.

Outline

Conclusions

Hasso Plattner Institut

Often CPS **requires** the capability of **self-awareness** to be able to handle problems due to unexpected circumstances

- Models must be able to evolve (**runtime models**)
- Systems must **reflect** on itself (**self-aware** of goals)
- Systems must **adapt**/self-adapt/learn

➔ existing formal models and analysis approaches for CPS are no longer applicable as they do not cover reflection/adaptation (design, verification, ...)

Graph transformation systems encoding models and their linking allow to combine Service-Oriented Architecture, Self-Adaptive / Self-Organization, and Runtime Models with evolving structures and are a suitable basis for a **solid foundation** for **Self-Adaptive CPS.**

- Collaborations support **SoS-Level Self-Organization**, **SoS-Level Structural Dynamics**, and **Runtime Knowledge Exchange**
- Runtime models and via collaborations shared runtime models enabled **Self-Adaptation** of the systems
- Compositional Verification is a first element for the **Resilience** of the SoS

[Giese+2015]

Outlook

Limitations:

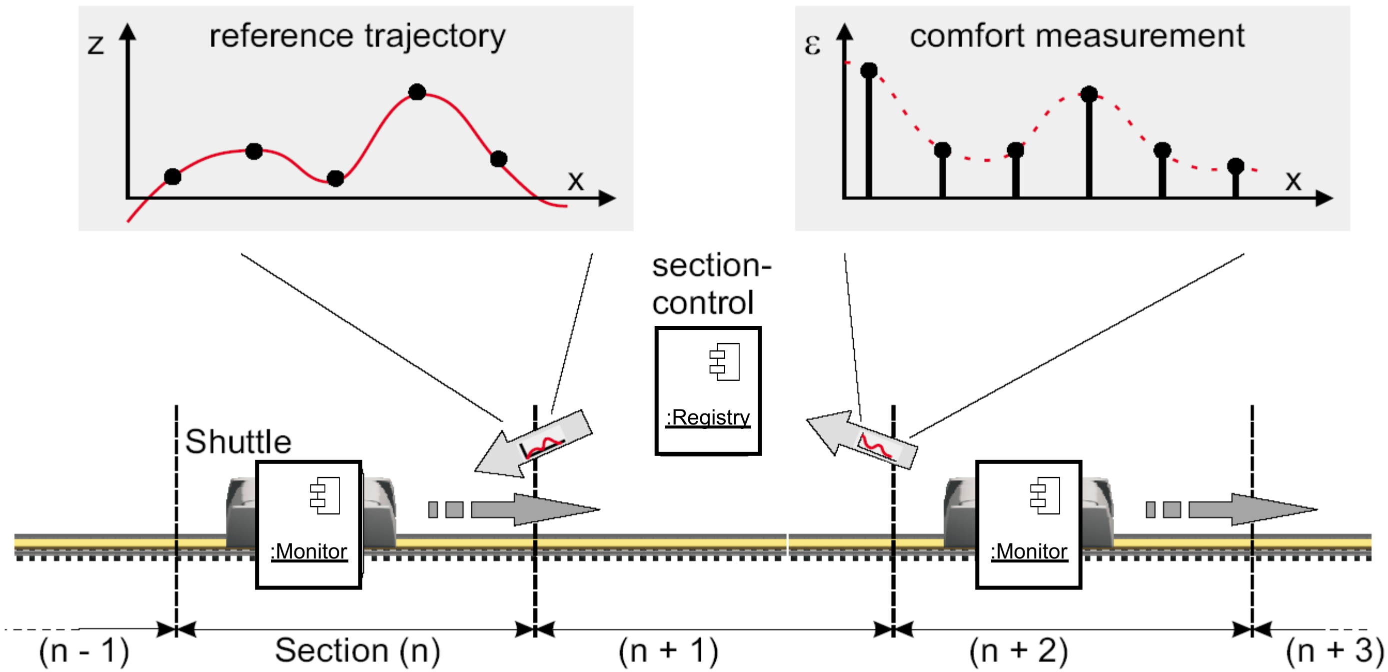
- The suggested model is a rather strong **idealization**:
  - If wrong, likely also related less idealized design will fail as well
  - More accurate explicit runtime models can be used (but then verification will get much harder)
    - the systems may copy (with some measurement errors) their context to an explicit runtime model to capture delays etc.
    - the systems may hand over copies of their runtime models to other systems such that the visible shared context is exchanged explicitly
- The formal model requires that a strong **separation** into collaborations is possible to support the compositional analysis
- Any approach based on formal models and analysis relies on the **validity/trustworthiness** of the employed models
  - Development-time models may become invalid over time
  - Run-time models may become invalid

[Giese+2015]

Is the Runtime Model valid/trustworthy? (2/2)

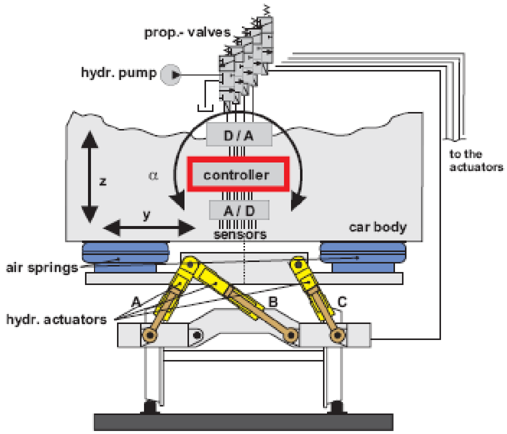
HPI Hasso Plattner Institut

[Burmester+2008]



- **Server (Registry of the section control; not global!):**
  - Offers track profile (distributed learning of a runtime model of the track)
- **Client (Monitor of the shuttle):**
  - Applies track profile (local learning of a runtime model of the shuttle and planning an adaptation in form of an optimal trajectory)
  - Must handle cases where the service is available or not

Is the Runtime Model valid/trustworthy? (2/2)



Suspension/tilt module

- air springs (filter for higher frequencies)
- active suspension system (lower frequencies)

We consider three different control strategies:

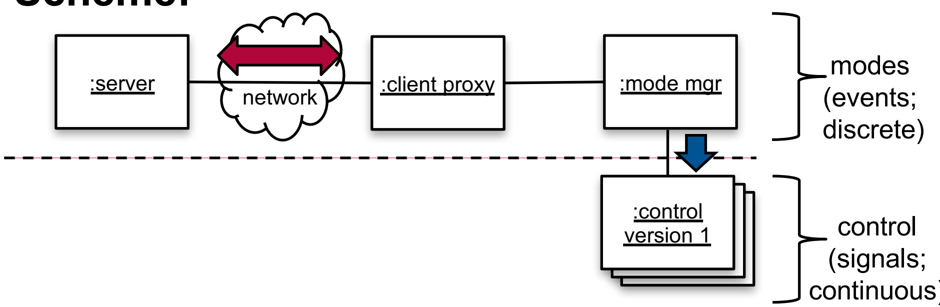
(1) **robust controller:** track as reference point; damping the relative movement
⇨ only achieves moderate damping.

(2) **absolute controller** uses a virtual skyhook in order to ensure the absolute acceleration of the shuttle body is minimized
⇨ comfort usually maximized; problematic on inclines

(3) **reference controller**: Instead of virtual skyhook, the real track is used as reference
⇨ highest comfort; requires data about the track

[Burmester+2008]

Scheme:



Client proxy:

- Find local responsible registry
- register at the local registry (requestInfo)
- Receive data from the registry (sendInfo)
- Manage cases where the data is available or not (outside the proxy)
- Send data to the registry (experience)
- **PLUS:** detect invalid runtime model!

PROBLEM: There is no guarantee that the runtime models are not invalid due to fact that they always rely on potentially erroneous or outdated measurements ➜ **detection + backup strategy necessary**

Bibliography

HPI 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.

[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+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.

[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.

Bibliography

[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.

[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.