



# Process-based Collaboration in Global Software Engineering

Harald Klein:

*Siemens AG, Corporate Technology*

*Software & Engineering Processes, 91051 Erlangen*

Andreas Rausch, Edward Fischer:

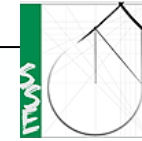
*Technische Universität Clausthal Zellerfeld, Chair for Software Systems*

*Engineering*

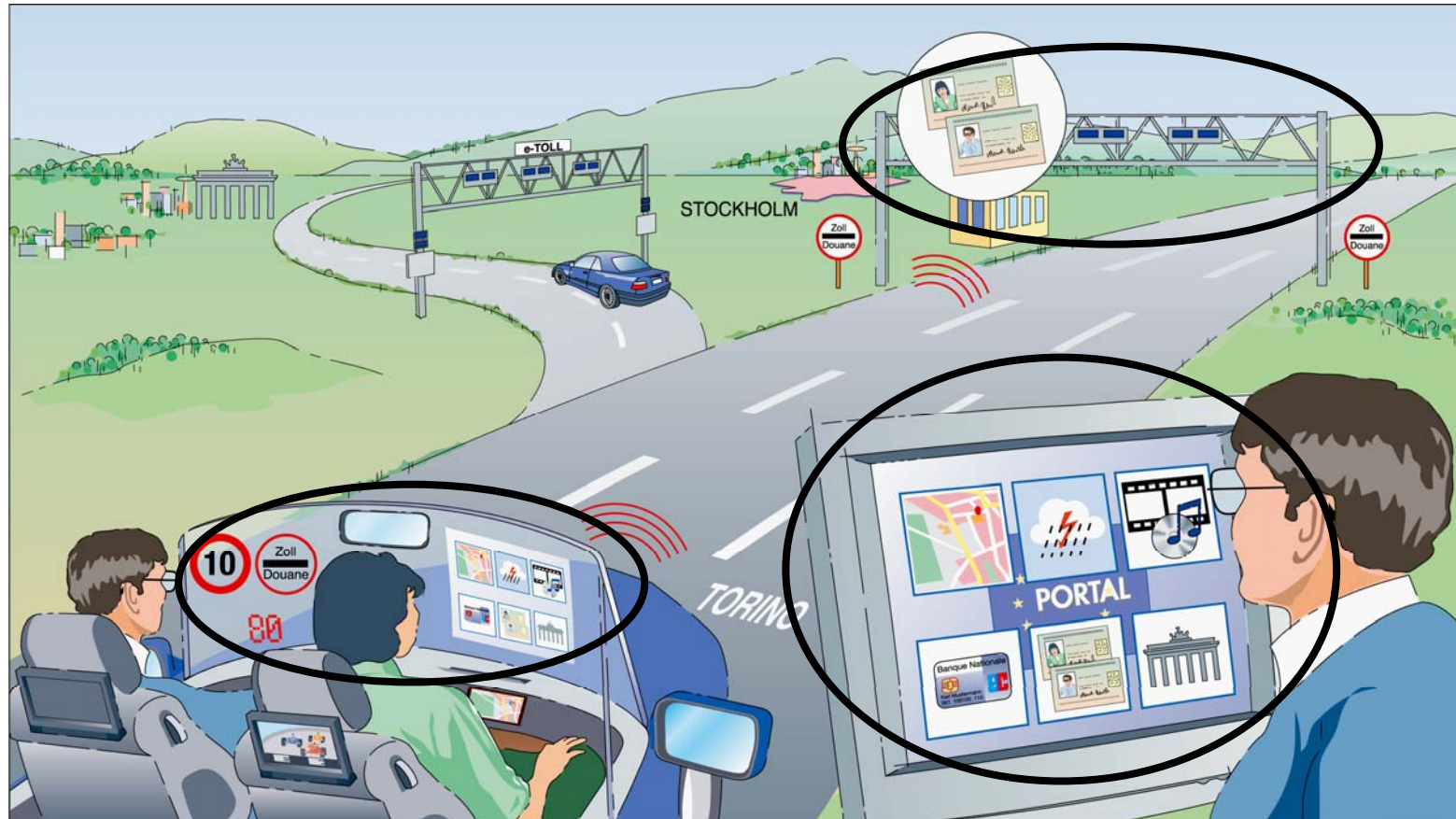


## Agenda

- **Motivation**
- **Collaboration Scenarios**
- **Formalization**
- **Outlook**



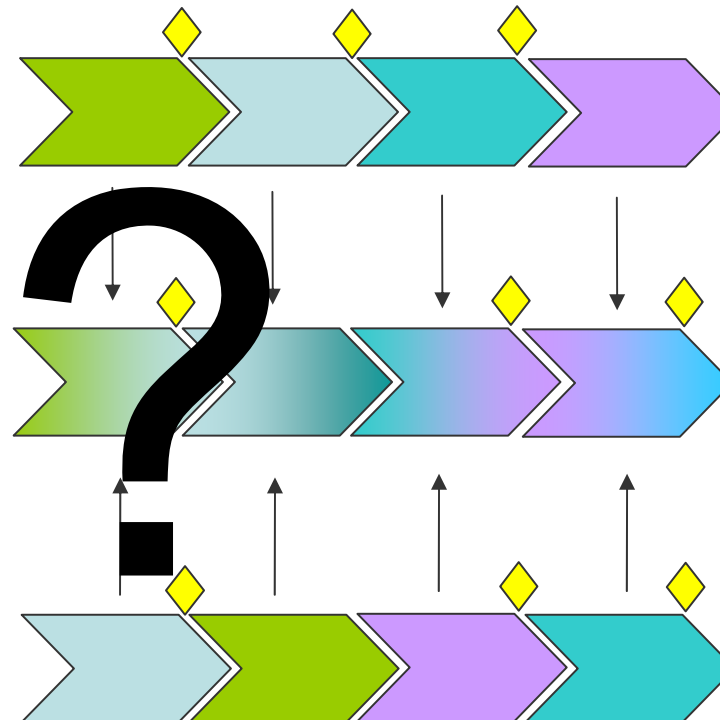
# Motivating Example “Extended Automated Toll-Collect”





# Motivation

Software/Hardware development process of organization "A"



Common software development process of organization A and B

Software/Hardware development process of organization "B"

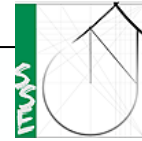
## Assumption

Goal of this approach:

- Enabling to construct syntactical correct processes and workflows
- Provide guideline for collaboration processes

What is not considered:

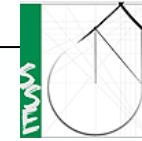
- Business decision support concerning the rationality of processes
- Lack of personnel
- Hand-off failures



## **Collaboration Scenarios**

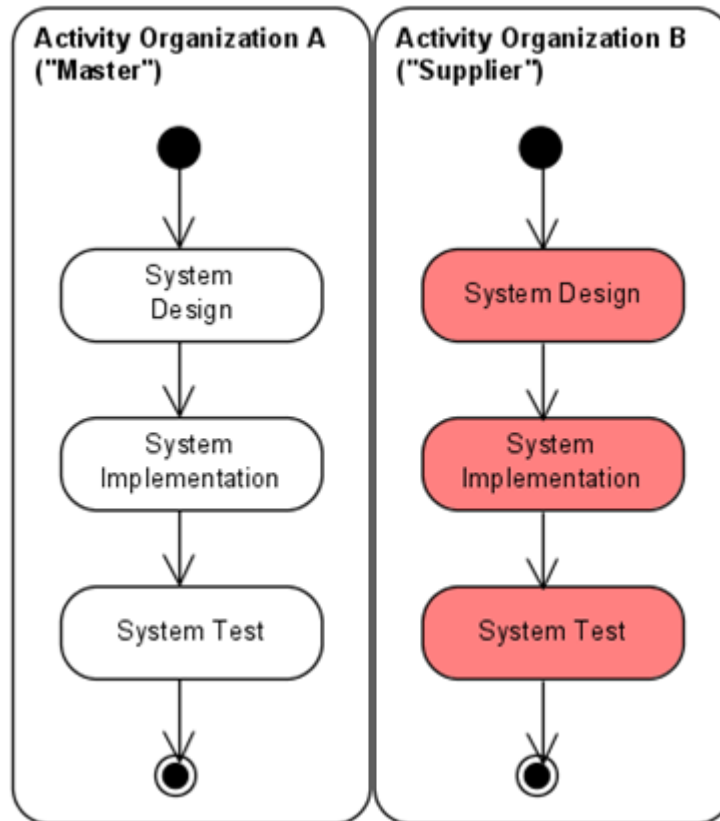
Four collaboration scenarios have been identified:

1. Collaboration with semantically equivalent processes
2. Collaboration with semantically different processes
  - a. Horizontal Integration
  - b. Additive vertical integration
  - c. Alternative vertical integration

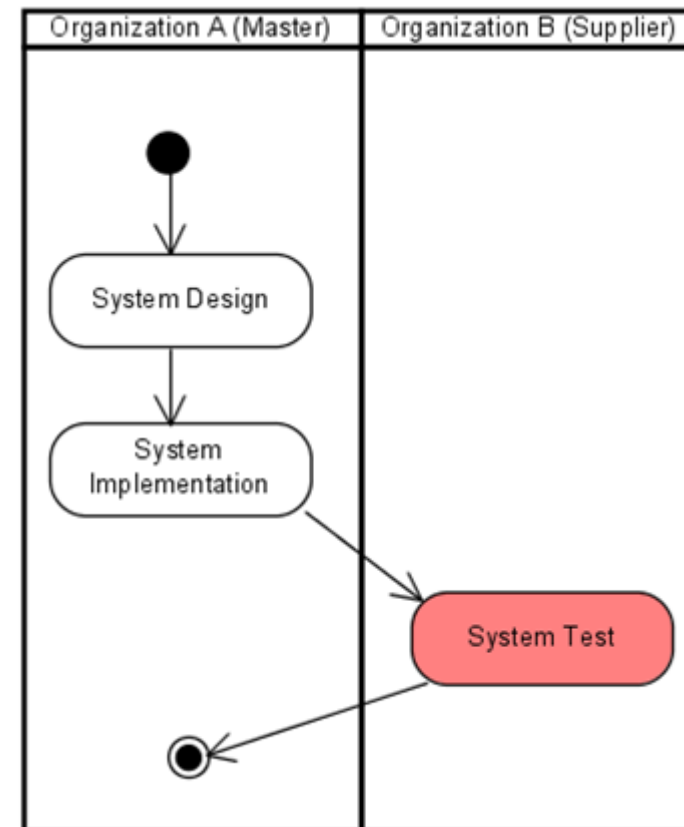


# 1. Collaboration with semantically equivalent processes

Initial Situation



Collaborative Process

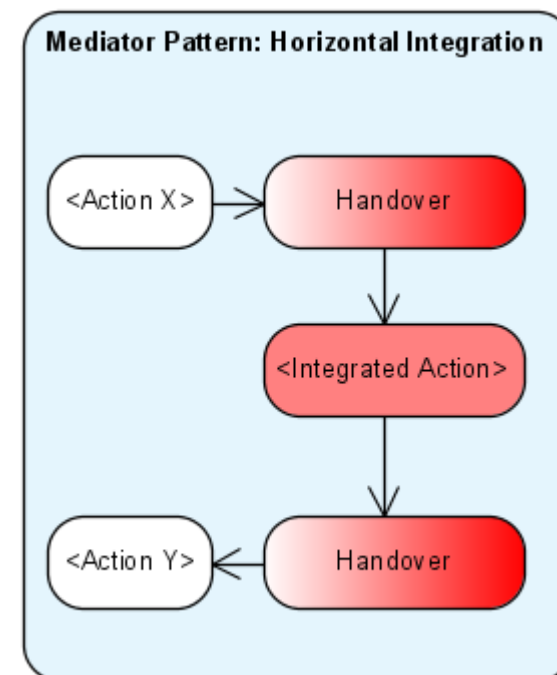


## 2. Semantically Different Processes

Definition of “Mediator Pattern”:

- “Function” bringing together two or more semantic different actions/workflows
- UML terminology is used for mediator definition, (however, UML rules are not fulfilled)
- Auxiliary character

Mediator Example:



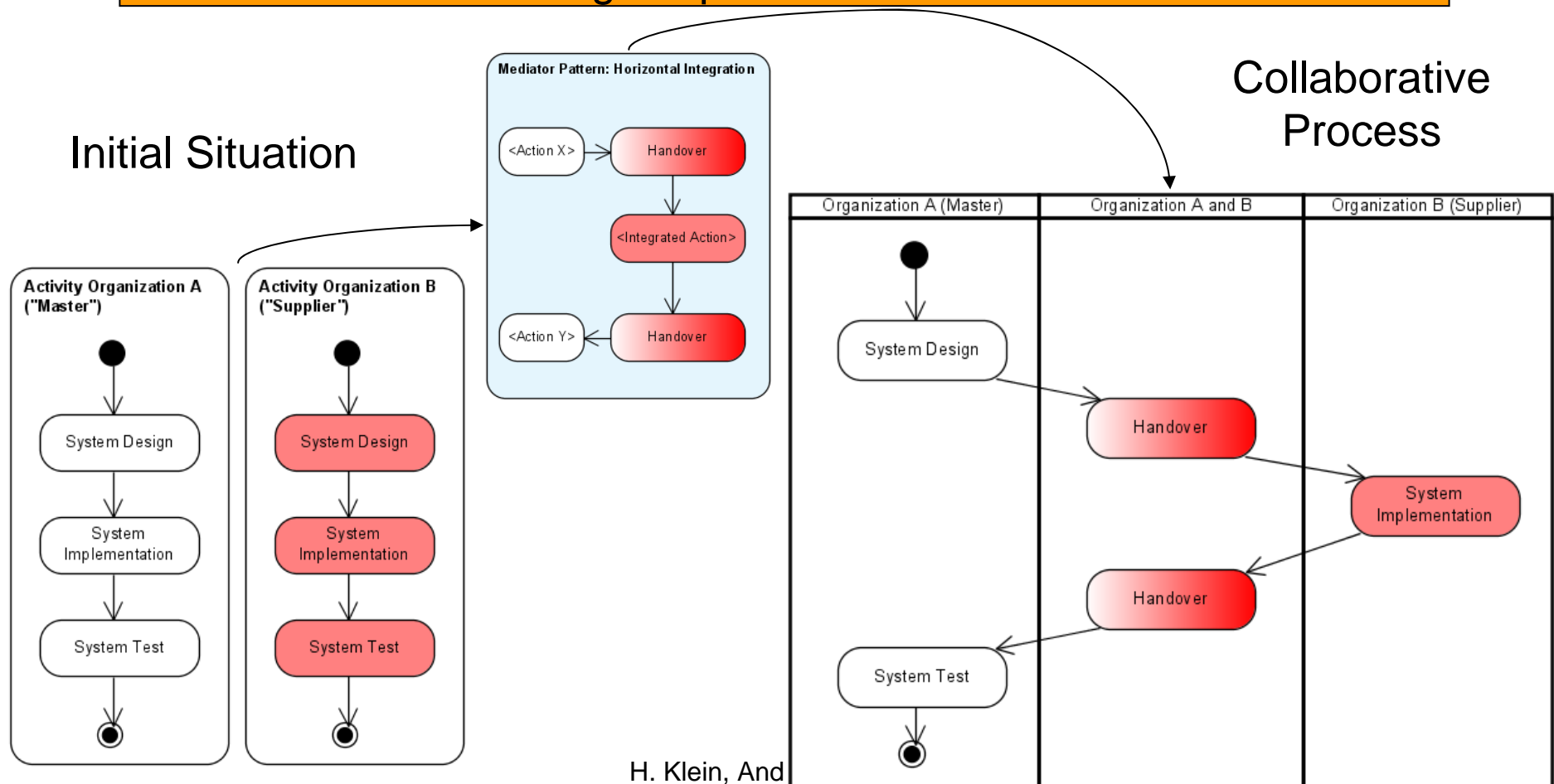
1.

- Similar to „Sub-contractor relationship“ in e.g. VM XT®, but...  
→ organizations have different process models defined

a.

- “Hand Over” activities clarify e.g....  
→ coding guidelines, templates to be used  
→ understanding of specification

IS



2.

- Parallelization of development activities
- “decomposition” clarifies WHAT parts a supplier should take over, not HOW (like in “hand-over” but based on specialized knowledge)
- “integration” is the integration of different domain results, e.g. software and hardware

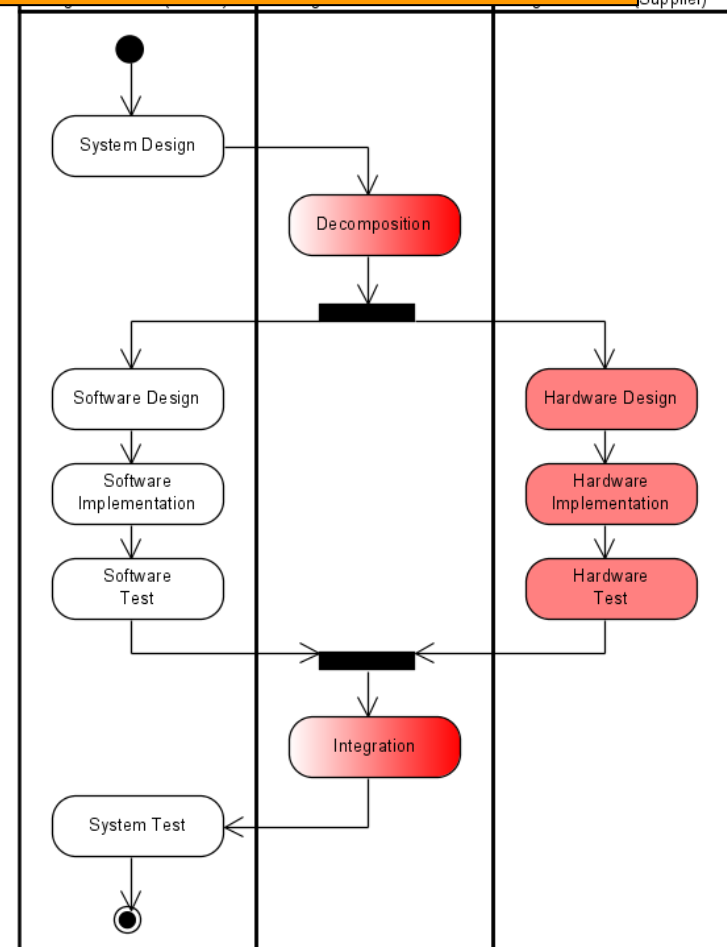
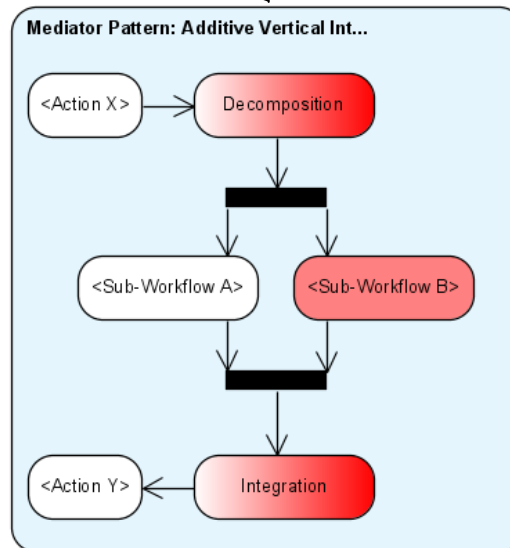
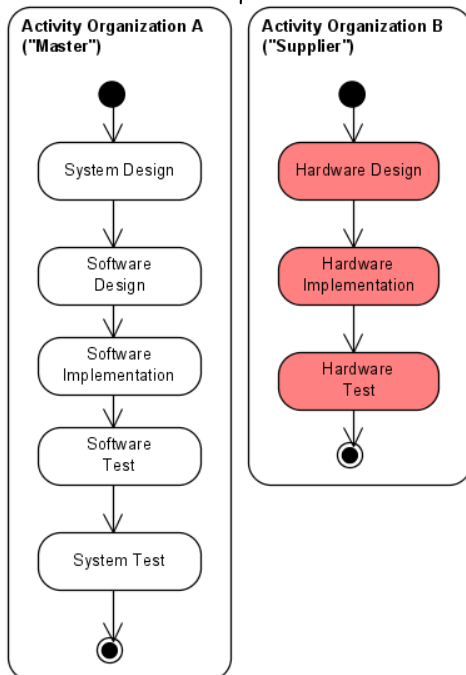
b.

S

e

Supplier)

Initial Situation

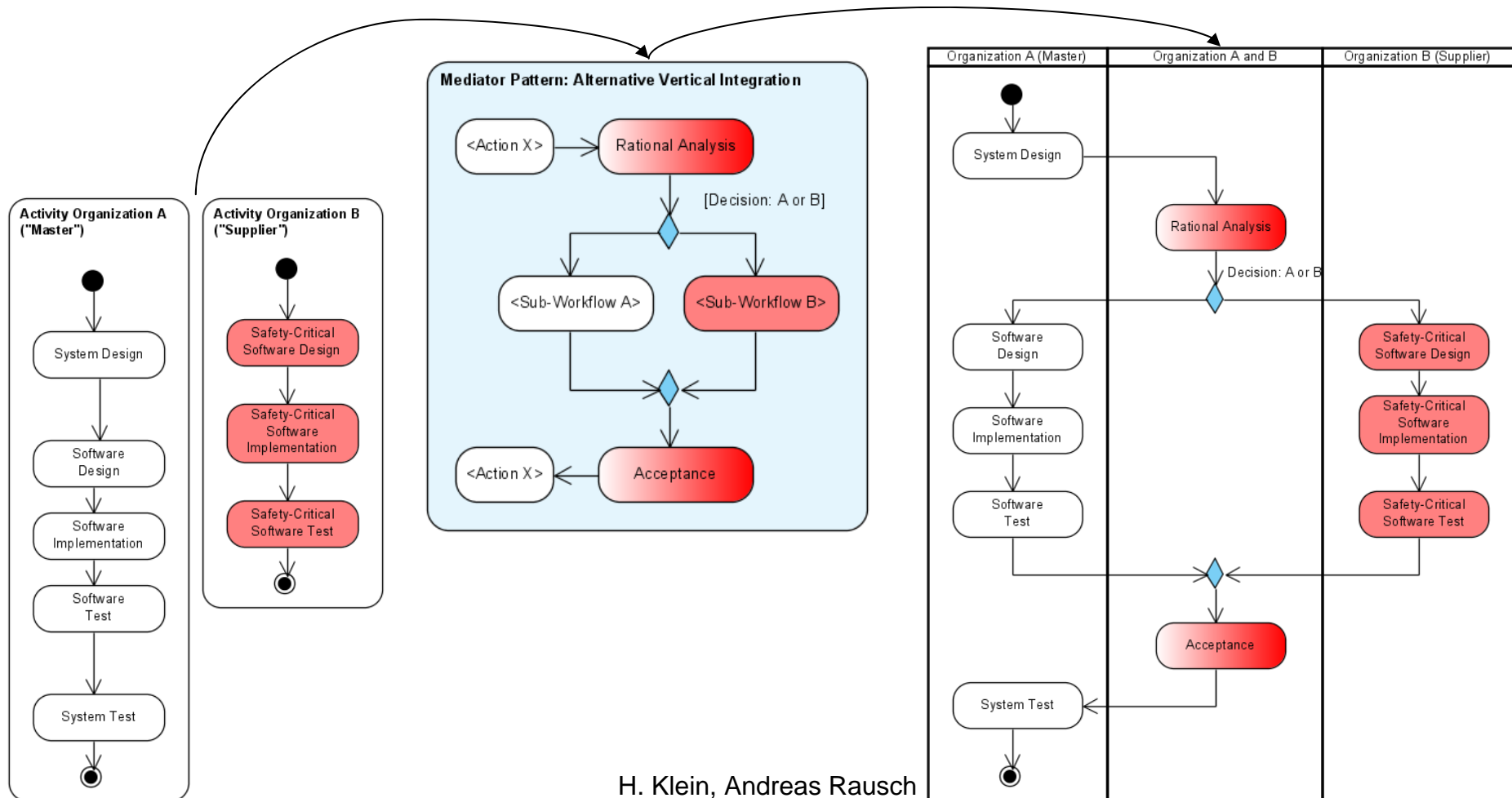


2.

- Alternative development paths are used → no parallelization
- Rational Analysis is used to allocate sub-product parts

NS

### c. Alternative Vertical Integration

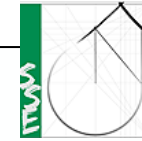


## Formalization

Reasoning for Formalization:

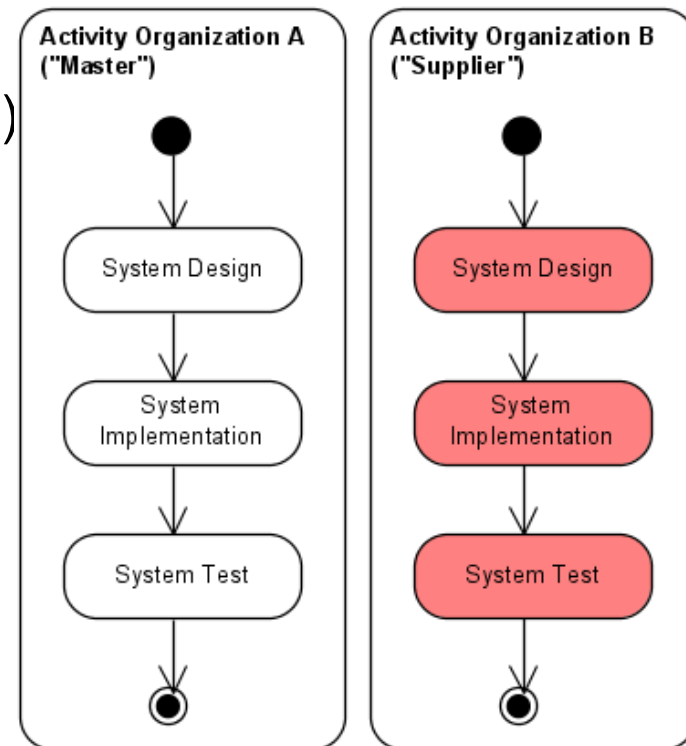
- Tool-based creation of collaboration scenarios (by directly sourcing V-Model XT®)
- Automated syntactical checks

Formalization uses a graph-based representation.



# Formalization

## Basic Definitions and Ideas (II)

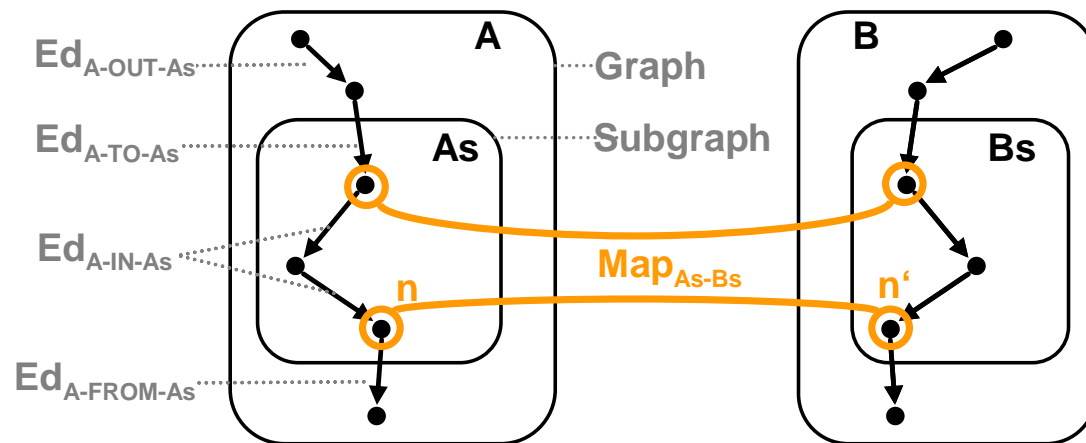
$$A = ( \{$$
$$\quad (Start, y_{start}), (End, y_{end}), (SystemDes., y_{action}),$$
$$\quad (SystemImpl., y_{action}), (SystemTest, y_{action})$$
$$\},$$
$$\{$$
$$\quad ((Start, y_{start}), (SystemDes., y_{action})),$$
$$\quad ((SystemDes., y_{action}), (SystemImpl., y_{action})),$$
$$\quad ((SystemImpl., y_{action}), (End, y_{end}))$$
$$\} )$$




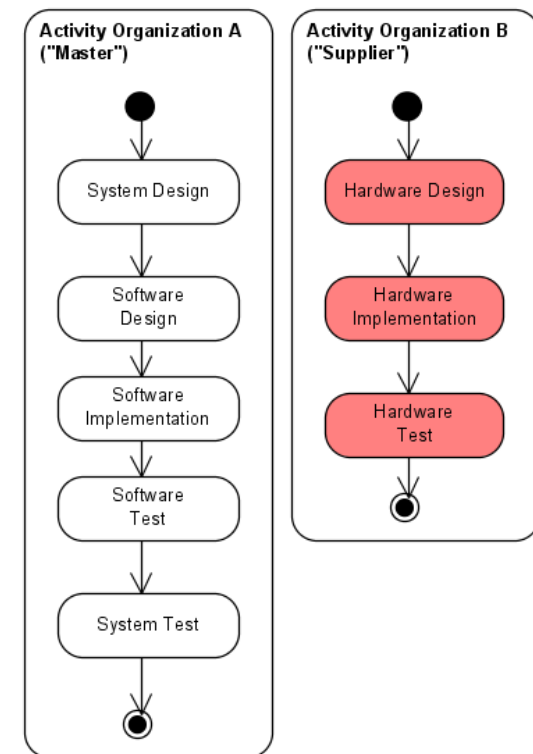
# Formalization „Closed Sub-graph“

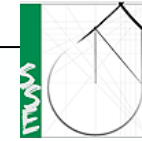
$$A_s \in AD \text{ is\_closed\_sub\_graph\_of } A \in AD \Leftrightarrow$$

$$Nd_{A_s} \subseteq Nd_A \wedge Ed_{A_s} \subseteq Ed_A.$$



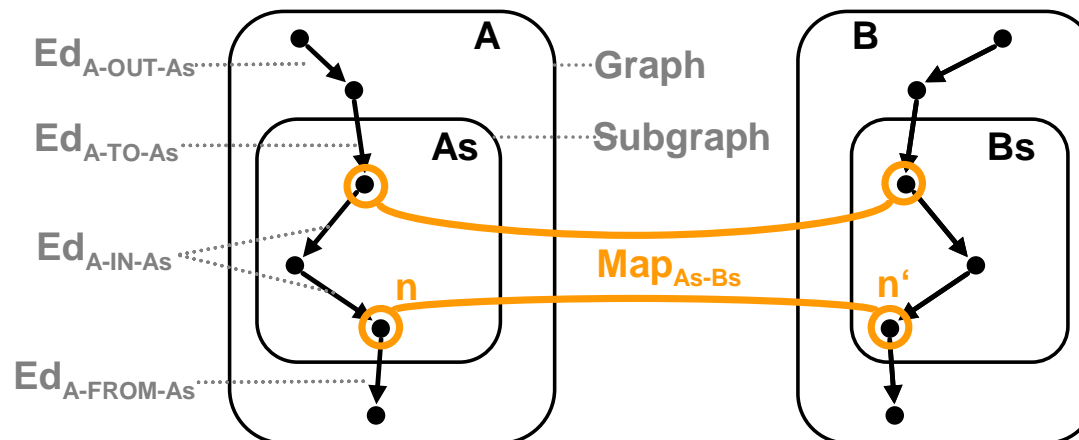
- $Ed_{A-OUT-As} :=_{DEF} \{ (s,t) \in Ed_A \mid s \notin Ed_{As} \wedge t \notin Ed_{As} \}$
- $Ed_{A-TO-As} :=_{DEF} \{ (s,t) \in Ed_A \mid s \notin Ed_{As} \wedge t \in Ed_{As} \}$
- $Ed_{A-IN-As} :=_{DEF} \{ (s,t) \in Ed_A \mid s \in Ed_{As} \wedge t \in Ed_{As} \}$
- $Ed_{A-FROM-As} :=_{DEF} \{ (s,t) \in Ed_A \mid s \in Ed_{As} \wedge t \notin Ed_{As} \}$

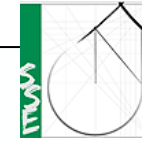




# Formalization „Mapping Operation“

$$\text{MAP}_{As-Bs} :=_{\text{DEF}} \{ \text{Map}_{As-Bs} \subseteq \text{Nd}_{As} \times \text{Nd}_{Bs} \mid$$
$$\forall (s,n) \in \text{Ed}_{A\text{-TO-}As} \Rightarrow \exists (n,n') \in \text{Map}_{As-Bs},$$
$$\forall (n,t) \in \text{Ed}_{A\text{-FROM-}As} \Rightarrow \exists (n,n') \in \text{Map}_{As-Bs} \}$$





# Formalization

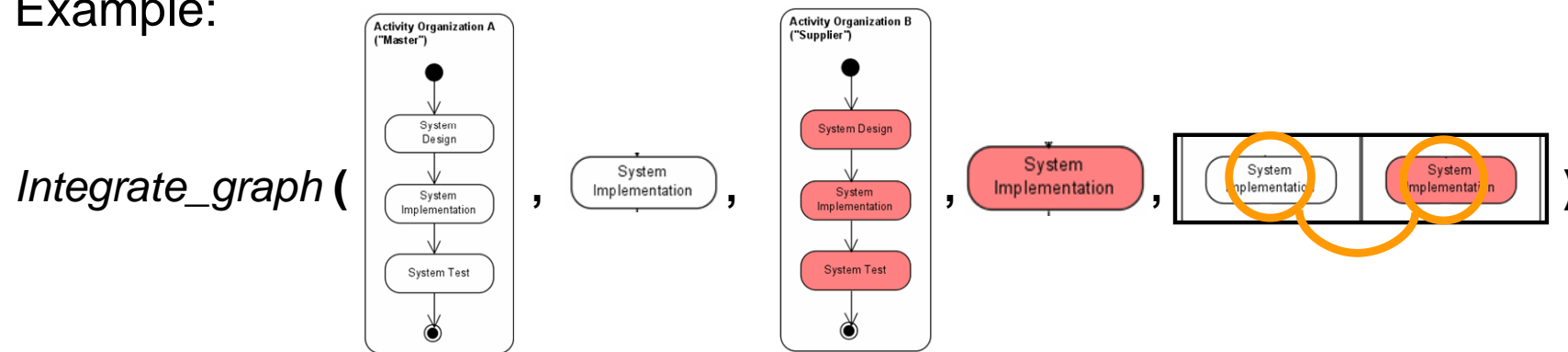
## Function: "Integration Operation"

$integrate\_graph(A, A_s, B, B_s, Map_{A_s-B_s}) :=_{DEF} AB,$

where

- $Nd_{AB} = ((Nd_A \cup Nd_B) \setminus Nd_{Del}) \cup Nd_{Add}$
- $Ed_{AB} = ((Ed_A \cup Ed_B) \setminus Ed_{Del}) \cup Ed_{Add}$
- $Nd_{Del}$  to be defined by concrete integration operation
- $Nd_{Add}$  to be defined by concrete integration operation
- $Ed_{Del}$  to be defined by concrete integration operation
- $Ed_{Add}$  to be defined by concrete integration operation

Example:





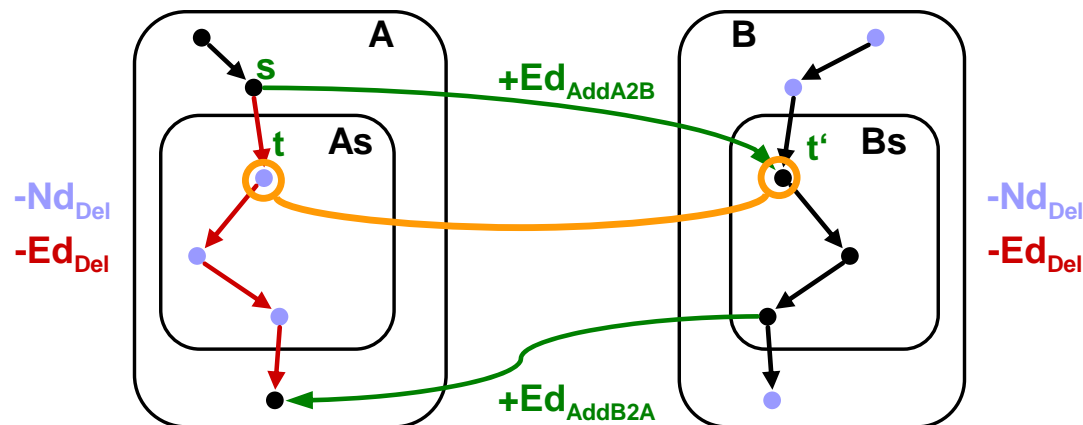
# Formalization

Operation: „Semantically equal processes“

*integrate\_SemEq* extends *integrate\_graph*

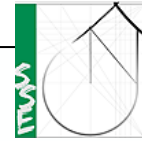
where

- $Nd_{Del} = Nd_{As} \cup (Nd_B \setminus Nd_{Bs})$
- $Nd_{Add} = \emptyset$
- $Ed_{Del} = Ed_{A-TO-As} \cup Ed_{A-FROM-As} \cup Ed_{A-IN-As} \cup Ed_{B-TO-Bs} \cup Ed_{B-FROM-Bs} \cup Ed_{B-OUT-Bs}$
- $Ed_{Add} = Ed_{AddA2B} \cup Ed_{AddB2A}$
- $Ed_{AddA2B} = \{ (s,t') \in ED \mid \exists (s,t) \in Ed_{A-TO-As}, (t,t') \in Map_{As-Bs} \}$
- $Ed_{AddB2A} = \{ (s',t) \in ED \mid \exists (s,t) \in Ed_{A-FROM-As}, (s,s') \in Map_{As-Bs} \}$



## Outlook and further activities

- Refinement and extension of collaboration diagrams (e.g. data flow diagrams)
- Empirical validation of this approach
- Application to development models and approaches e.g. V-Model XT®, CMMI®, RUP®



**SIEMENS**

**Thank you very much for your attention!**

# Questions?

