

# Semantic Business Process Engineering

Summer School Reasoning Web 2010

SYSTEMATIC THOUGHT LEADERSHIP FOR INNOVATIVE BUSINESS



Dr. Jens Lemcke, SAP Research Center Karlsruhe

## Overview

- Part 1: Background: Semantic Business Process Engineering
- Part 2: Definitions: Grounding and Refinement
- Part 3: Practical Experience: Survey
- Part 4: Petri Net Solution
- Part 5: OWL-DL Solution
- Part 6: Comparison

# Part 1

## Semantic Business Process Engineering

Starting Point: Standard Business Software

Documents the procedures of a company

- Example: Order-to-cash process
  - Receiving orders, Managing production or procurement, Shipment to customer, Billing

“Standard” means ‘general-purpose’

- Industry best practices can be implemented, but software must be customized to match specific business procedures of clients

No monolithic block, but modularized as components, for example based on department boundaries:

- Enterprise resource planning (ERP)
- Supply chain management (SCM)
- Customer relationship management (CRM)
- Human resources (HR)

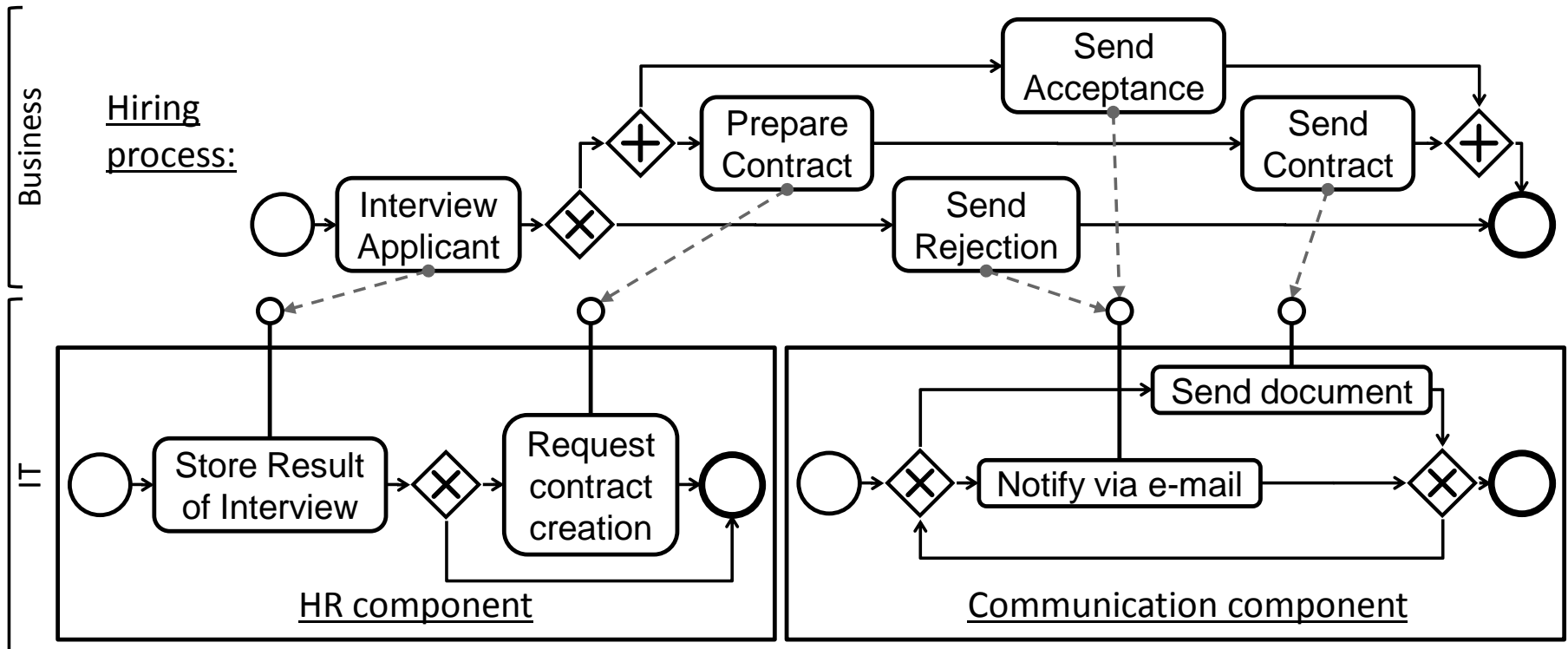
# Components are stateful

Procedures of the company are the procedures of the departments

→ **Components are no black boxes, but stateful**

- Possible future behavior depends on previous actions

Example:



# Usage supported by business process management (BPM) tools



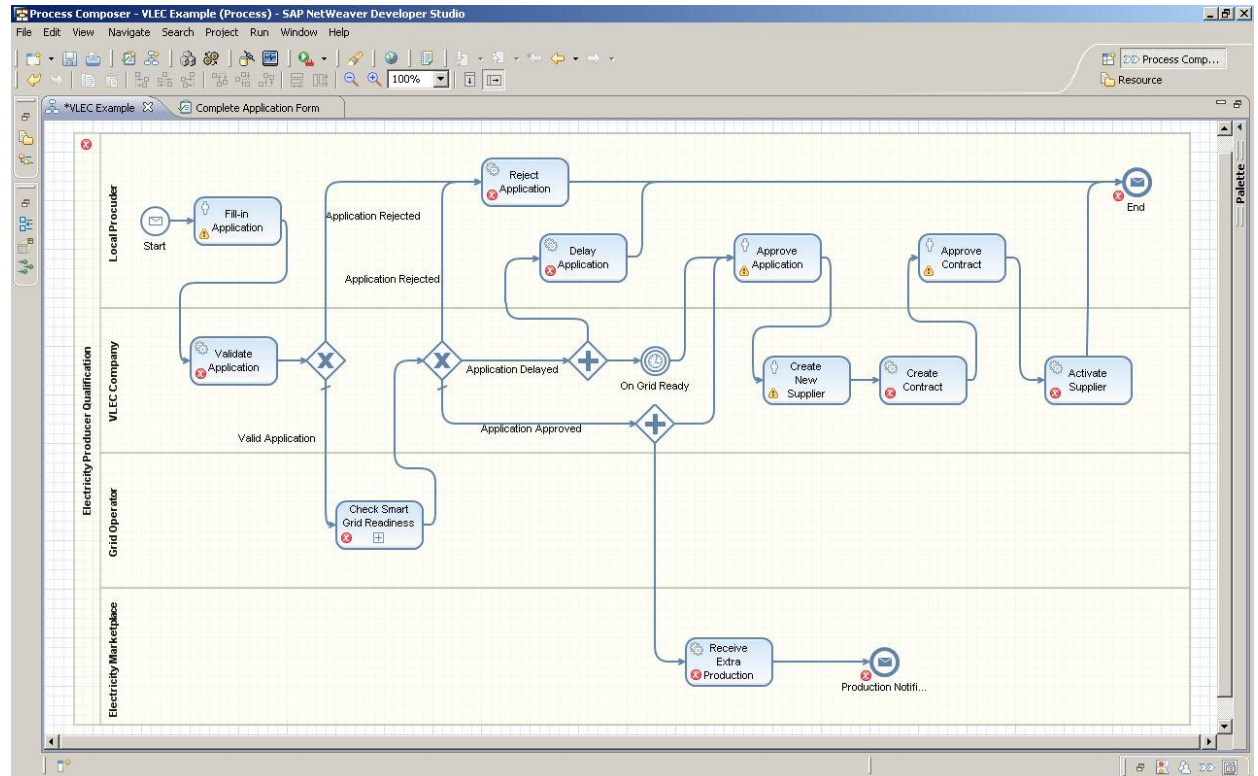
Business process management tools, such as SAP NetWeaver BPM:

- Modelling a business process, graphical language: business process model and notation (BPMN) [OMG, 2009]
- **“Grounding”**: Connecting business process to Web service endpoints
- Web service interfaces exposed by SAP or 3<sup>rd</sup> party software components
- Execution and monitoring

➔ **1<sup>st</sup> Challenge: Is the grounding feasible?**

Purpose of BPM

- Unite business and IT roles
- ➔ **2<sup>nd</sup> Challenge: How to combine their differently abstract views of a process?**



# Part 1

## Semantic Business Process Engineering

Terms and Related Research Areas

## Semantic business process engineering

Formal methods

### Business process engineering

Software engineering

Business process management

# Business process management builds the basis



“A **business process** is a collection of related, structured activities that produce a specific service or product for a particular customer or customers.”

“A **business process model** is an explicit formulation of the behaviors of a business process.”

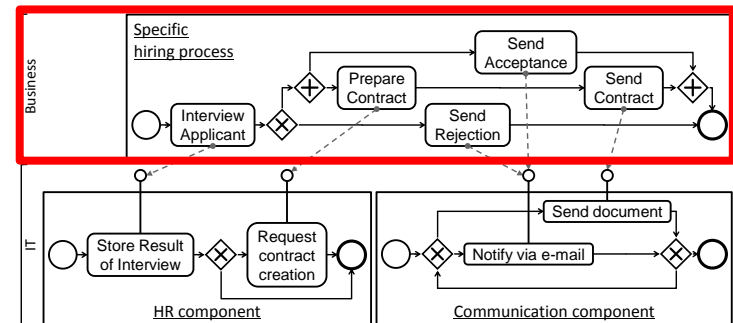
Phases:



Involved roles

- Business expert (more business-oriented)
- Business analyst
- Process architect (more IT-oriented)

- ➔ Mapping process on existing IT components
- ➔ Business and IT-related roles
- ➔ Process-centric (one layer of abstraction)



# Software engineering adds interconnected models

“**Software engineering** is a profession dedicated to designing, implementing, and modifying software so that it is of higher quality, more affordable, maintainable, and faster to build.”

## Software development processes

- Strictly sequential, e.g., “waterfall” [Royce, 1987]:



- Iterative, e.g.:

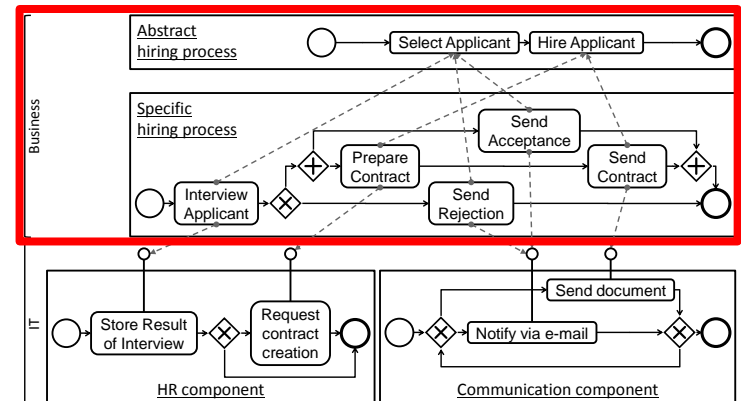
- Spiral model [Boehm, 1988]
- Extreme programming [Beck, 1999]
- Scrum method [Deemer et al., 2010]

- Commonalities

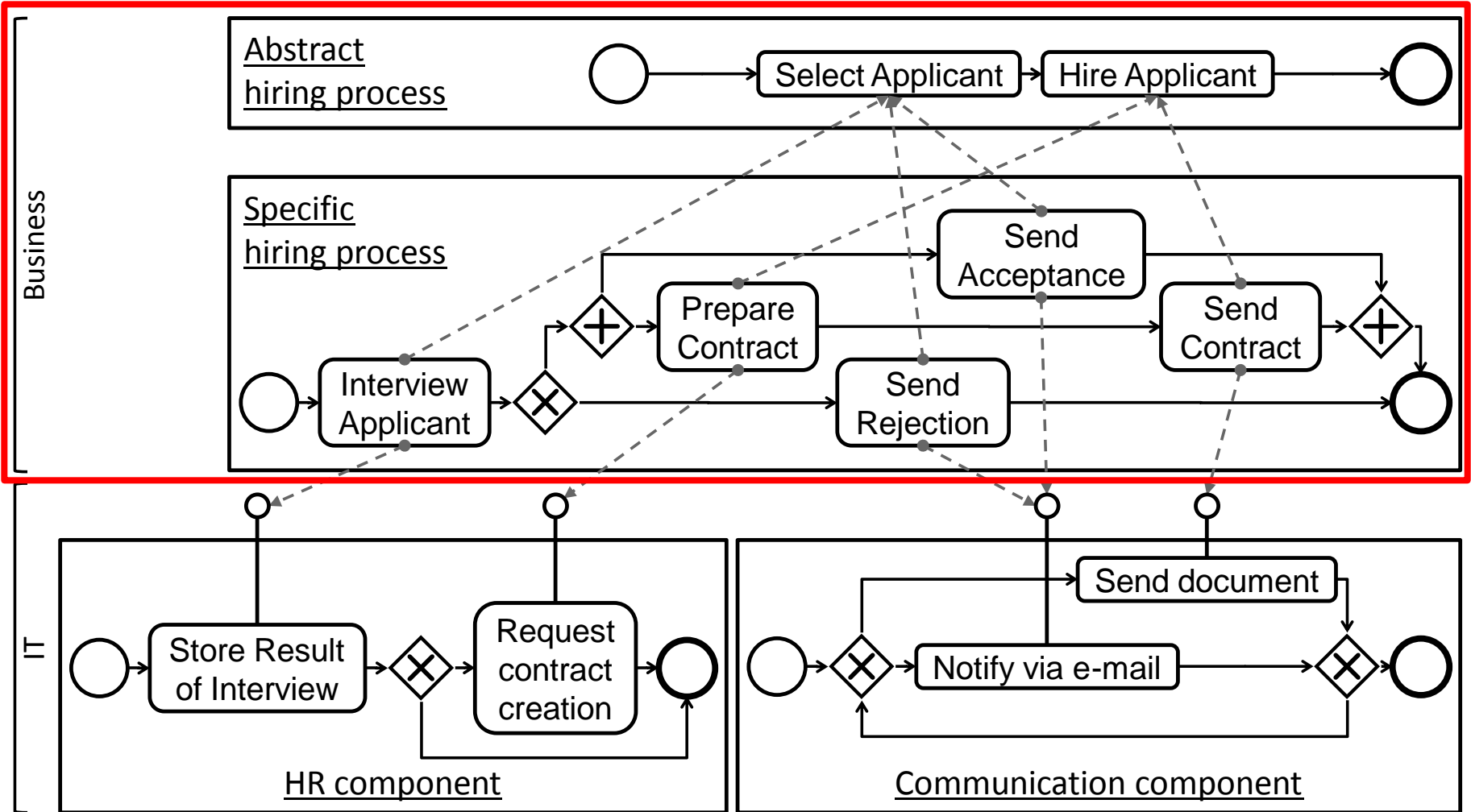
- Principle steps are the same
- Artifacts created refer to artifacts from previous steps

➔ **Structured development approach**

➔ **From higher-level to lower-level artifacts (requirements → architecture → code)**



# Refinement of business processes



# Formal methods ensure validation of consistency

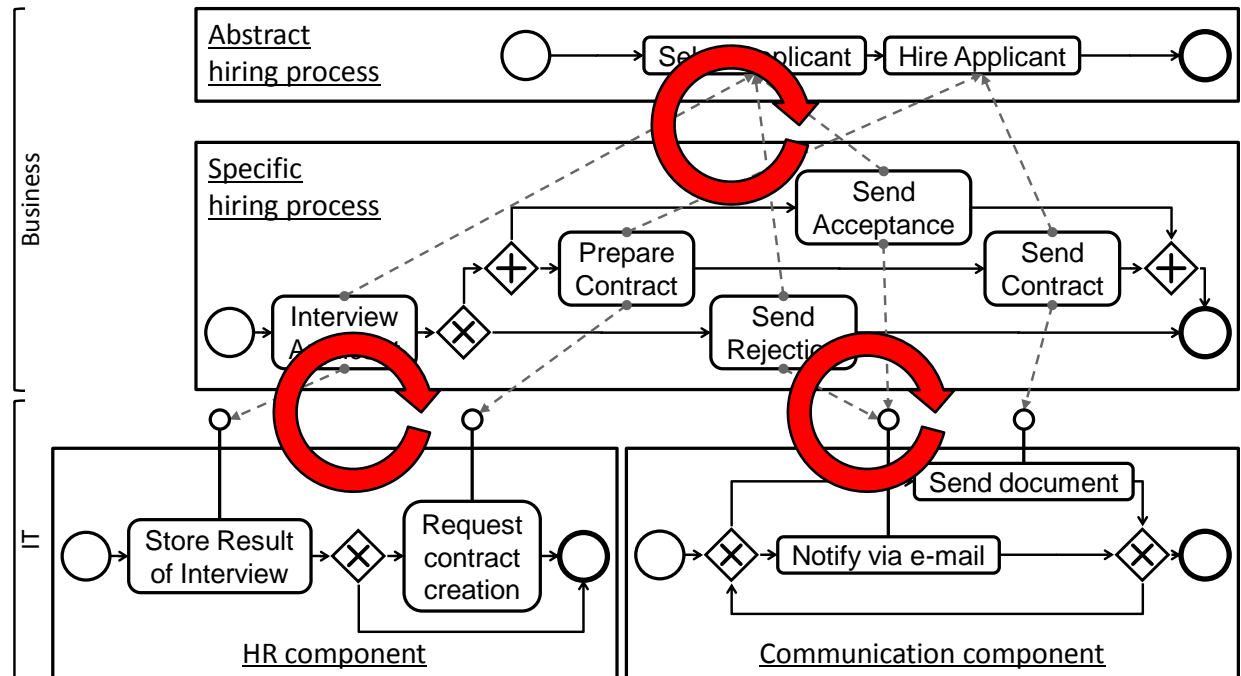
“**Formal methods** are a particular kind of mathematically-based techniques for the specification, development and verification of software and hardware systems”

## Classes of techniques

- Logic calculi
- Formal languages
- Automata theory
- Program semantics
- Type systems
- Algebraic data types

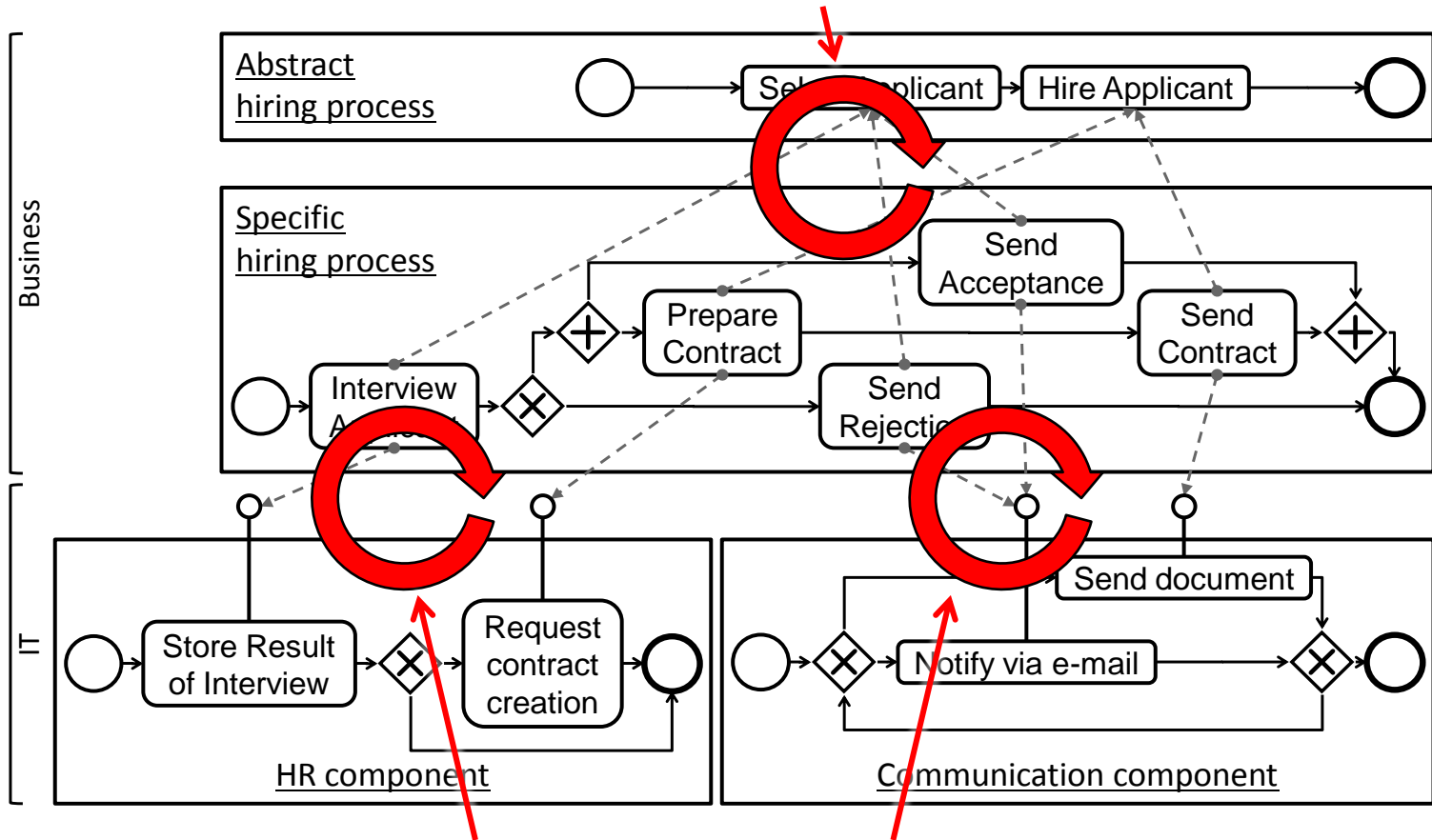
## Examples

- Model checking
- Petri nets
- OWL-DL



→ **Rigorous consistency between artifacts**

→ 2nd Challenge “Refinement validation”:  
How to combine the differently abstract views of a process?



→ 1st Challenge “Grounding validation”:  
Is the grounding feasible?

## Part 2

# Grounding and Refinement

## Grounding

The challenges are cumbersome and error-prone when performed manually.

→ That can be experienced by the audience in part 3.

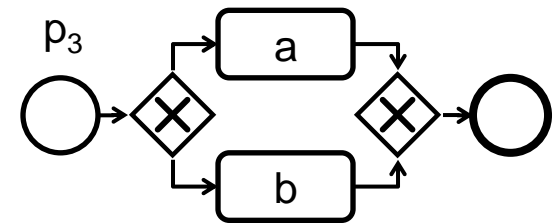
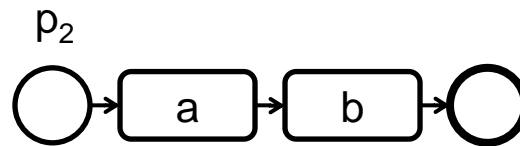
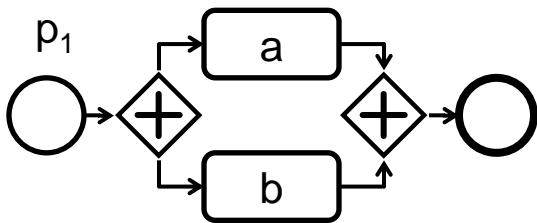
Prerequisite: We now explain in part 2 how consistency is defined theoretically.

## Syntax of process and component model

- Described using BPMN
- Graph-based
- Elements: Start and end events, activities (set A), parallel, exclusive, and implicit gateways
- For component models, activities are called operations (set O)

## Semantics of process models given as (possibly infinite) execution set

- Process p, execution set denoted  $ES_p$ , (sequential) execution expressed by  $[a_1, \dots, a_n]$
- Examples:
  - $ES_{p_1} = \{[a,b],[b,a]\}$
  - $ES_{p_2} = \{[a,b]\}$
  - $ES_{p_3} = \{[a],[b]\}$



## Grounding function

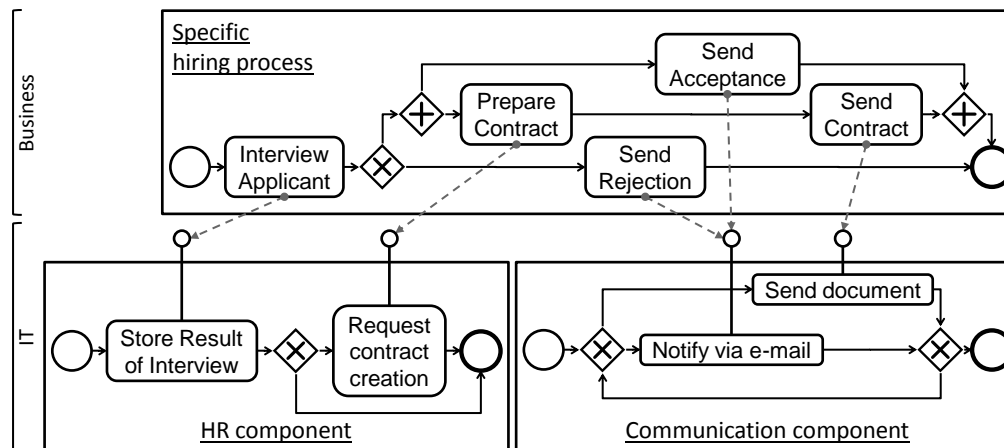
- grnd:  $A \rightarrow O$

## Grounding completeness

- A process is **completely grounded** iff every activity is grounded on some operation  
→ **Purely syntactic check (no semantics involved)**

## Grounding correctness (of a process)

- A process is **correctly grounded** iff it grounds correctly on each component



# Preliminary for grounding correctness of specific pairs of process and component

Maximal execution set semantics [Wyner and Lee, 2003]

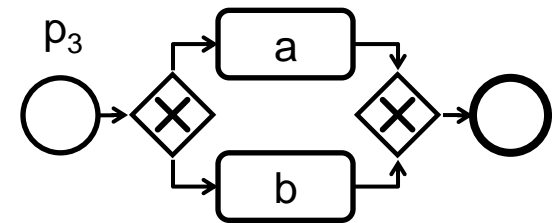
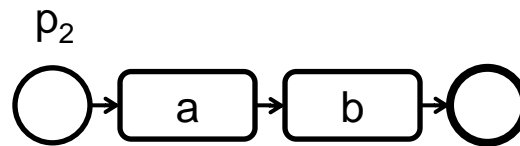
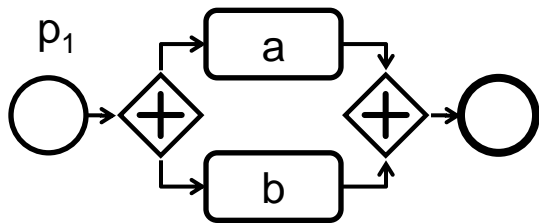
- A process  $p$  subsumes another process  $q$  under maximal execution set semantics iff  $ES_q$  is a subset of  $ES_p$

Question:

Which process subsumes another under maximal execution set semantics?

- Recap of execution sets:

- $ES_{p_1} = \{[a,b],[b,a]\}$
- $ES_{p_2} = \{[a,b]\}$
- $ES_{p_3} = \{[a],[b]\}$



# Preliminaries and grounding correctness for specific pairs of process and component



Extend signature of  $\text{grnd}$  to executions

- $\text{grnd}([a_1, a_2, \dots, a_n]) = [o_1, o_2, \dots, o_n]$  iff for all  $a_i$   $\text{grnd}(a_i) = o_i$

Projection of execution

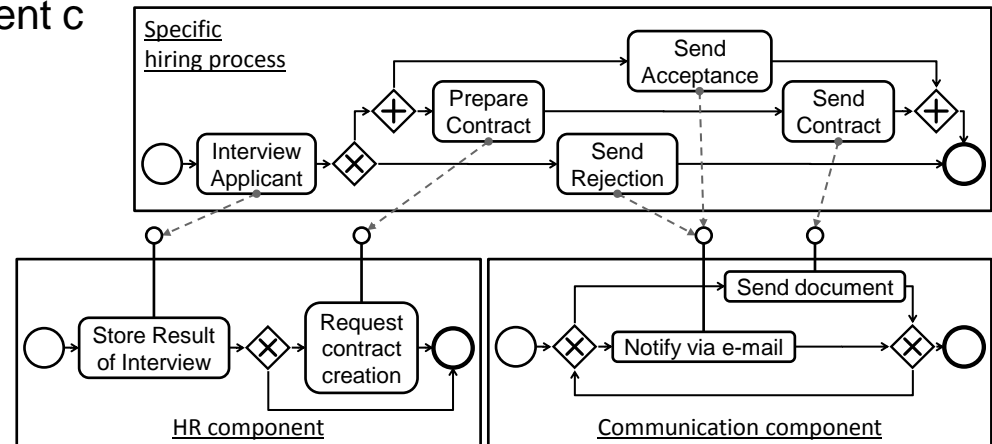
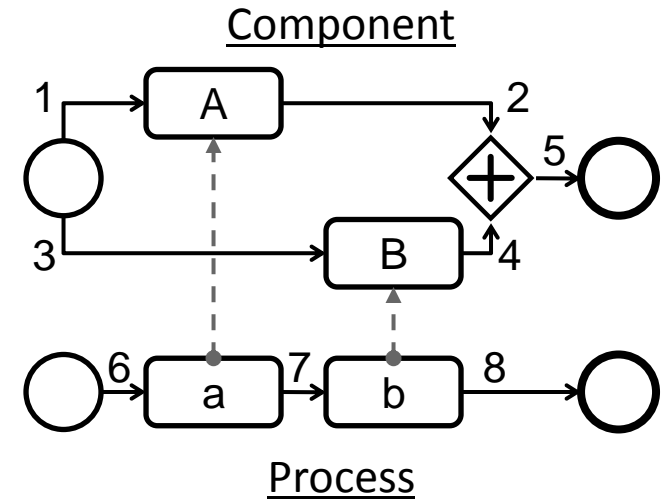
- Let  $c$  be a component, then  $[o_1, o_2, \dots]/c = [ \text{only } o_i \text{ contained in } c ]$

Projection of execution sets

- $\text{ES}/c = \{e/c : e \text{ in } \text{ES}\}$

Grounding correctness (of a process and a specific component)

- A process  $p$  correctly grounds on component  $c$  iff  $\text{grnd}(\text{ES}_p)/c$  is a subset of  $\text{ES}_c$ .



# Part 2

## Grounding and Refinement

Refinement

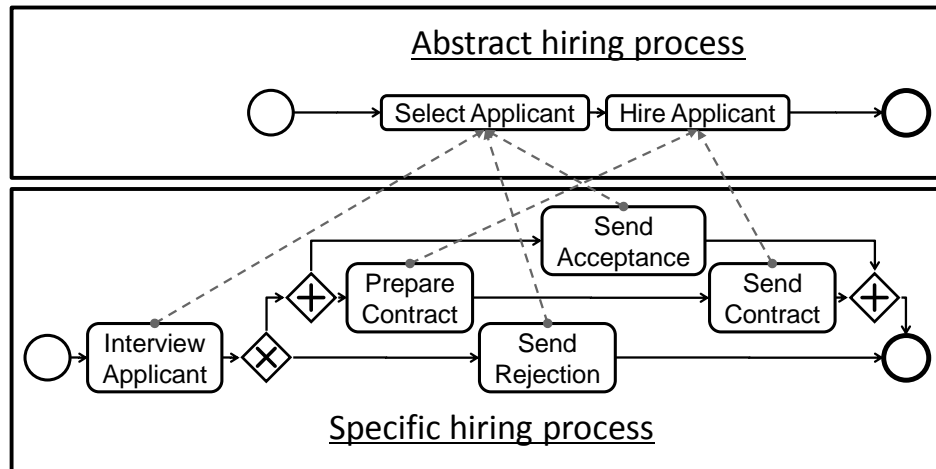
## Refinement function

- orig:  $A \rightarrow A$

## Refinement completeness

- Two processes: abs and spec,
  - each activity of spec refines some activity of abs and
  - each activity of abs is refined by at least one activity of spec

→ **Purely syntactic check (no semantics involved)**

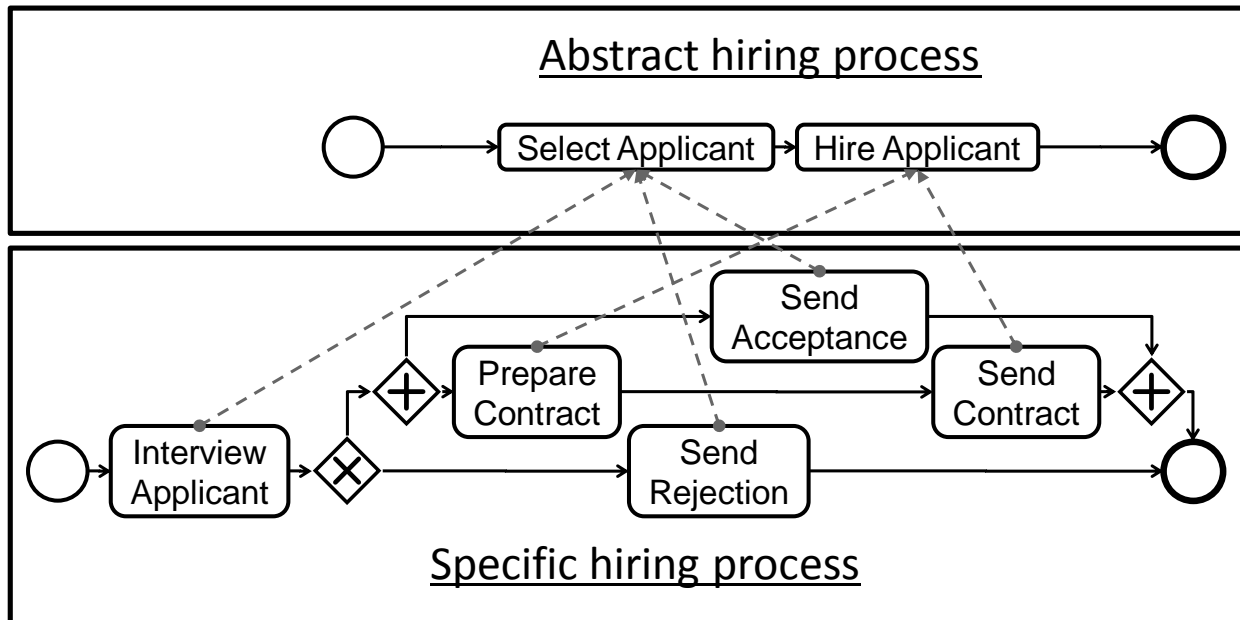


# Challenge for refinement correctness

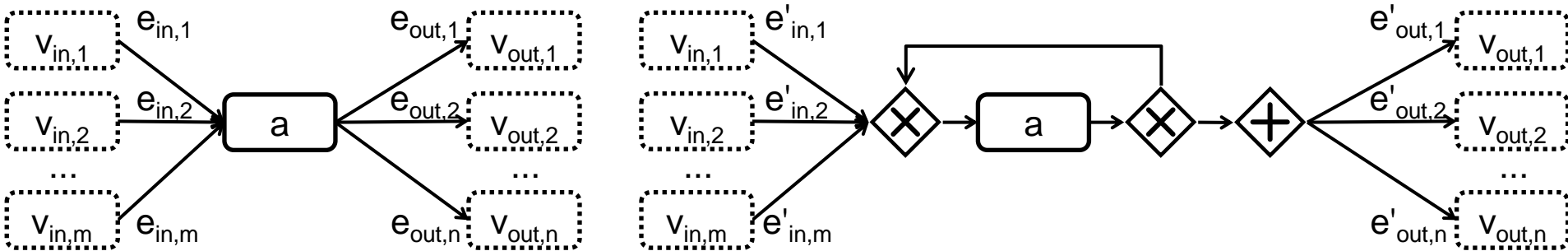
An abstract activity can be refined by multiple specific activities.

In other words:

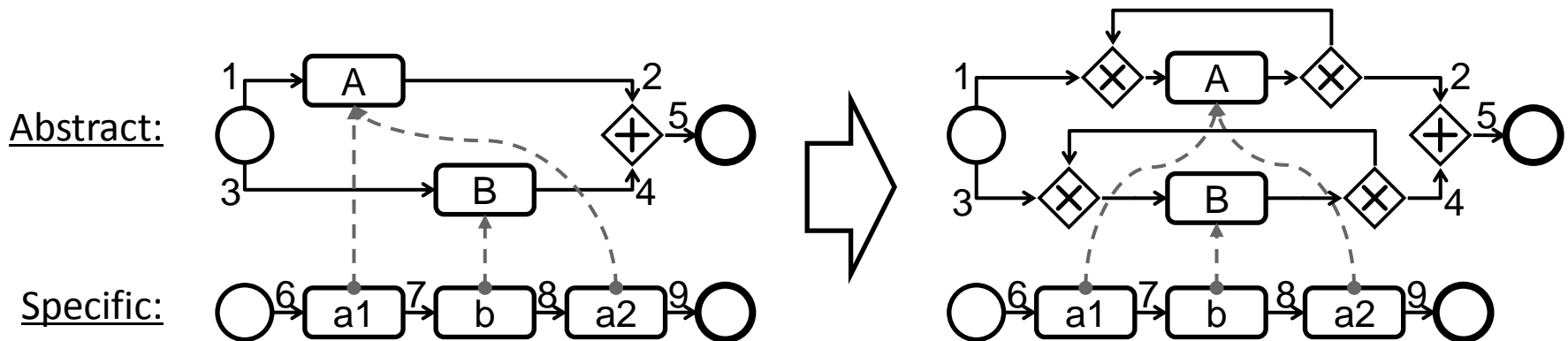
An abstract activity can be active while multiple specific activities are active



Let  $p$  be a process, then the decomposable process, denoted by  $p^D$ , is constructed from  $p$  by constructing a loop around every activity of  $p$  (see figure)



➔ Decomposable process allows abstract activities to happen repeatedly for multiple refined activities; **Example:**



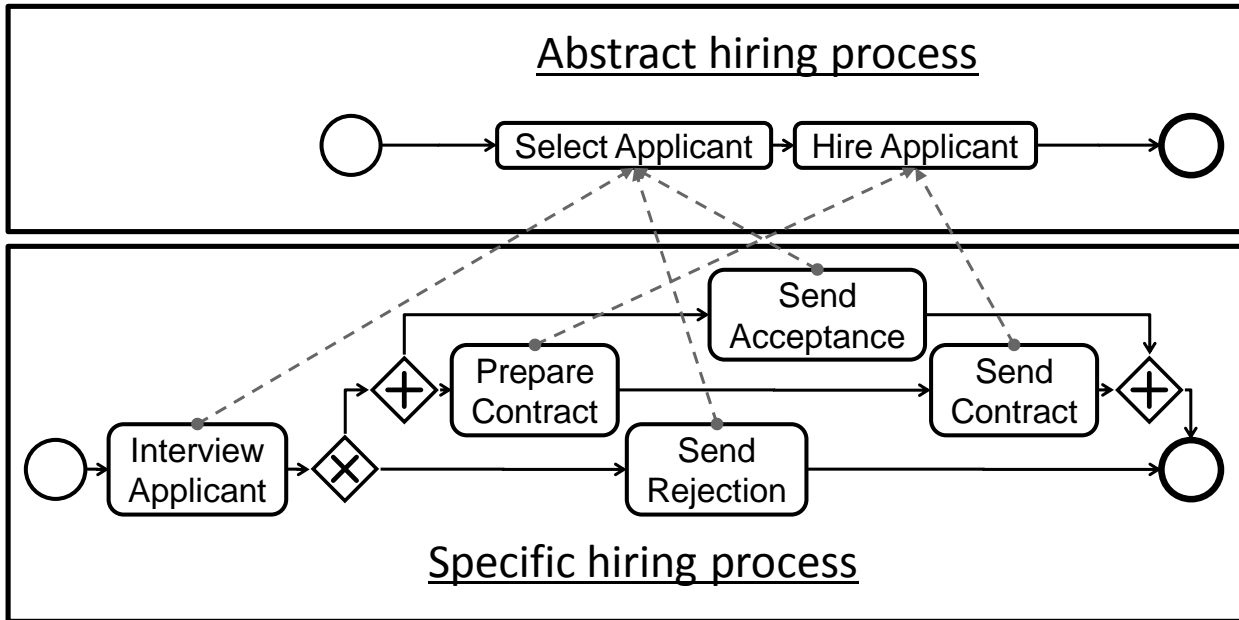
# Example for refinement correctness

Preliminary: Extend signature of orig to executions as for grd

Refinement correctness

- Process spec correctly refines process abs iff  $\text{orig}(ES_{\text{spec}})$  is a subset of  $ES_{\text{absD}}$

Example:



Question: Which refinement errors exist between the processes?

# Part 3

## Practical Experience

### Introduction

The audience may experience the difficulties of refinement validation in a short, practical experiment

At the same time, we would like to store the performance data for our future evaluation

Performance of a previous experiment will be shared afterwards

## Structure

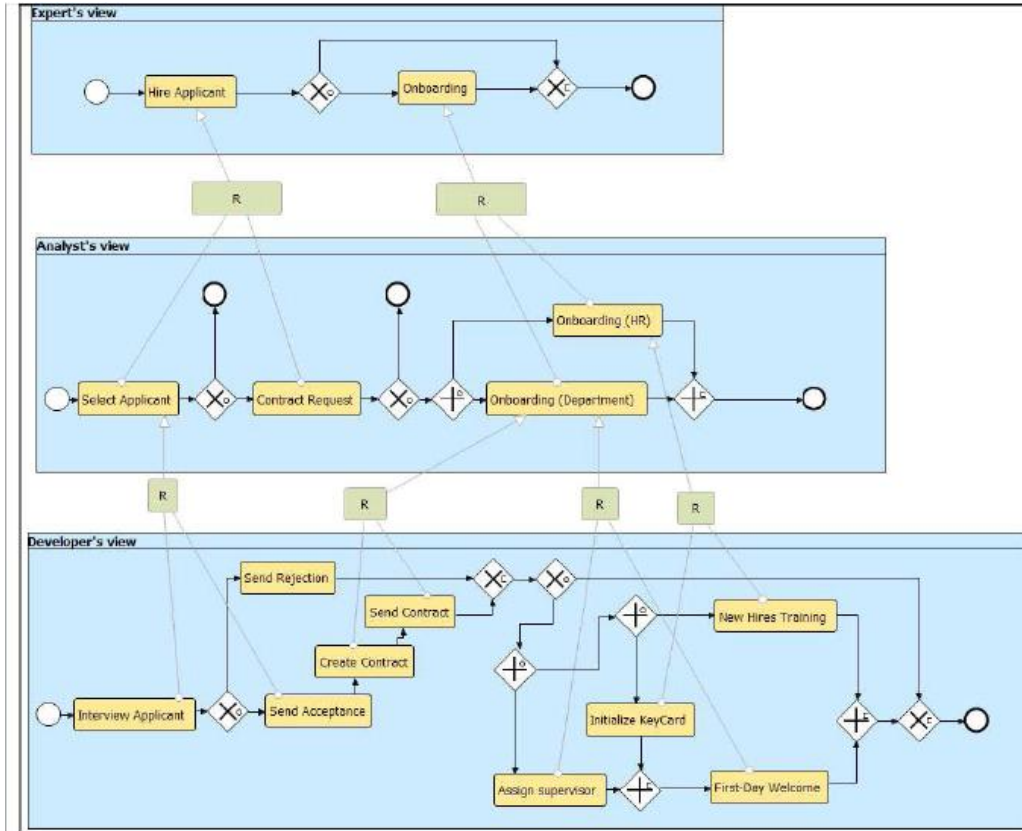
- 2 main surveys:
  - Survey 1 with no support to modeller
  - Survey 2 with all errors highlighted in the BPMN diagrams
- Each survey consists of 2 sub-surveys with 3 questions each
- Each question
  - 3 processes in a refinement hierarchy
  - Model real-life business processes from Curran et al. (1999):  
“*SAP R/3 Business Blueprint: Understanding Enterprise Supply Chain Management*”
  - Multiple-choice: 2-4 answer options, multiple or no correct options
  - Some models have multiple errors

## Infrastructure

- Use your own laptop to perform the survey → URL: <http://www.most-survey.tiricompas.de>
- Copy token to clipboard and paste it in every sub-survey
- For paper-based test only: Please note the start time of each group and very end time

Available time: 15 minutes

# Example for survey question without support



Please choose all that apply:

- Some refinement errors exist between the expert and analyst view.
- Some refinement errors exist between the analyst and developer view.
- No errors exist.

# Example for survey question with support

**Expert's view**

**Analyst's view**

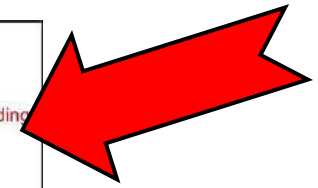
**Developer's view**

**Refinement Flaws:  
(Analyst's View ← Developer's View):**

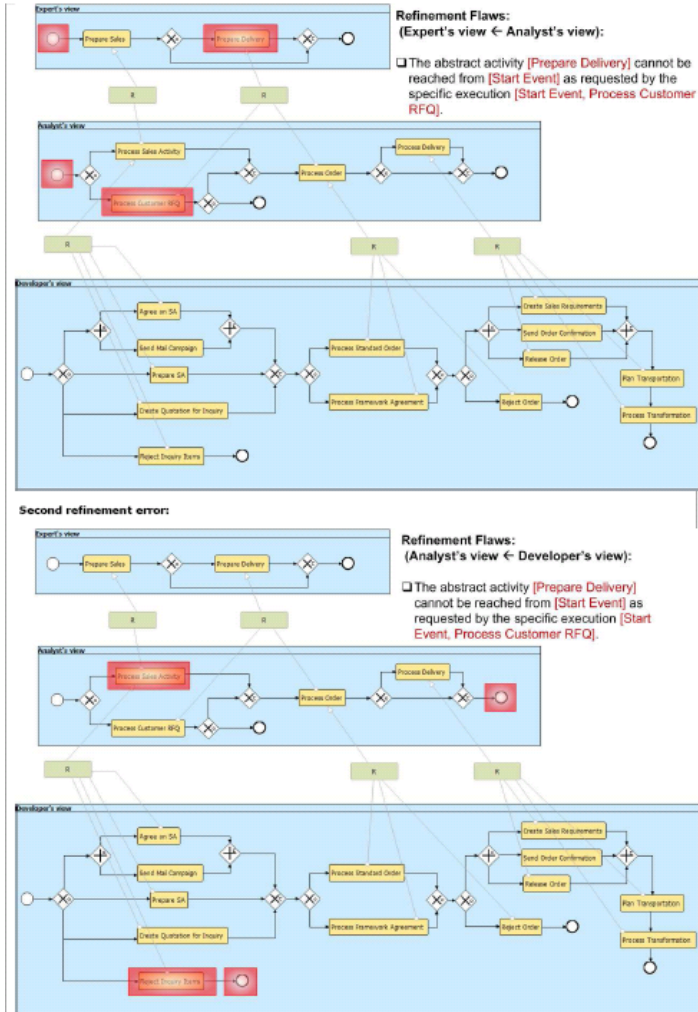
- The ordering of the abstract activities [Onboarding (HR), Onboarding (Department)] does not correspond to [Assign supervisor, First-Day Welcome] and [New Hires Training, Initialize KeyCard] as requested by the specific execution.

Please choose **all** that apply:

- Some refinement errors exist between the expert and analyst view.
- Some refinement errors exist between the analyst and developer view.
- No errors exist.



# One question contains 2 errors



- The fact that the activity Prepare Delivery cannot be reached directly after the Start Event in the expert's view causes no refinement errors.
- The fact that the activity Process Customer RFQ can be executed directly after the Start Event in the analyst's view causes a refinement error with respect to the developer's view.
- The fact that the activity End Event can be executed directly after the activity Reject Inquiry Items in the developer's view causes a refinement error with respect to the analyst's view.
- The ordering of the activities Prepare SA and Process Standard Order in the developer's view causes a refinement error with respect to the analyst's view.
- None of the above answers holds true.

# Part 3

## Practical Experience

### Hands-on

URL: <http://www.most-survey.tiricompass.de>

- Copy token to clipboard and paste it in every sub-survey
- For paper-based test only: Please note the start time of each group and very end time

Available time: 15 minutes.

If you want to see your (anonymized) performance on our Web site,

- Take your token to your notes if you do the laptop-based test.
- Write some characters on your paper if you take the paper-based test.

# Part 3

## Practical Experience

Performance in Previous Experiment

# Result of previous survey: Refinement validation is beneficial



## Setup

- Paper-based multiple choice test
- 13 experts from ontology & model-driven software development
- Longer version: 2 sets (S – support, N – no support) of 20 questions
- 1.5 hours

## Data gathered

- $C_S, C_N$  – correct answers with / without support
- $W_S, W_N$  – wrong answers with / without support
- $t_S, t_N$  – time taken for all questions with / without support

## Measure: Quality improvement per person

- $Q = \frac{C_S / (C_S + W_S)}{C_N / (C_N + W_N)} - 1$       average improvement over all persons: **avg(Q) = 70%**

## Measure: Productivity improvement per person

- $P = \frac{C_S / t_S}{C_N / t_N} - 1$       average improvement over all persons: **avg(P) = 378%**

# Part 4

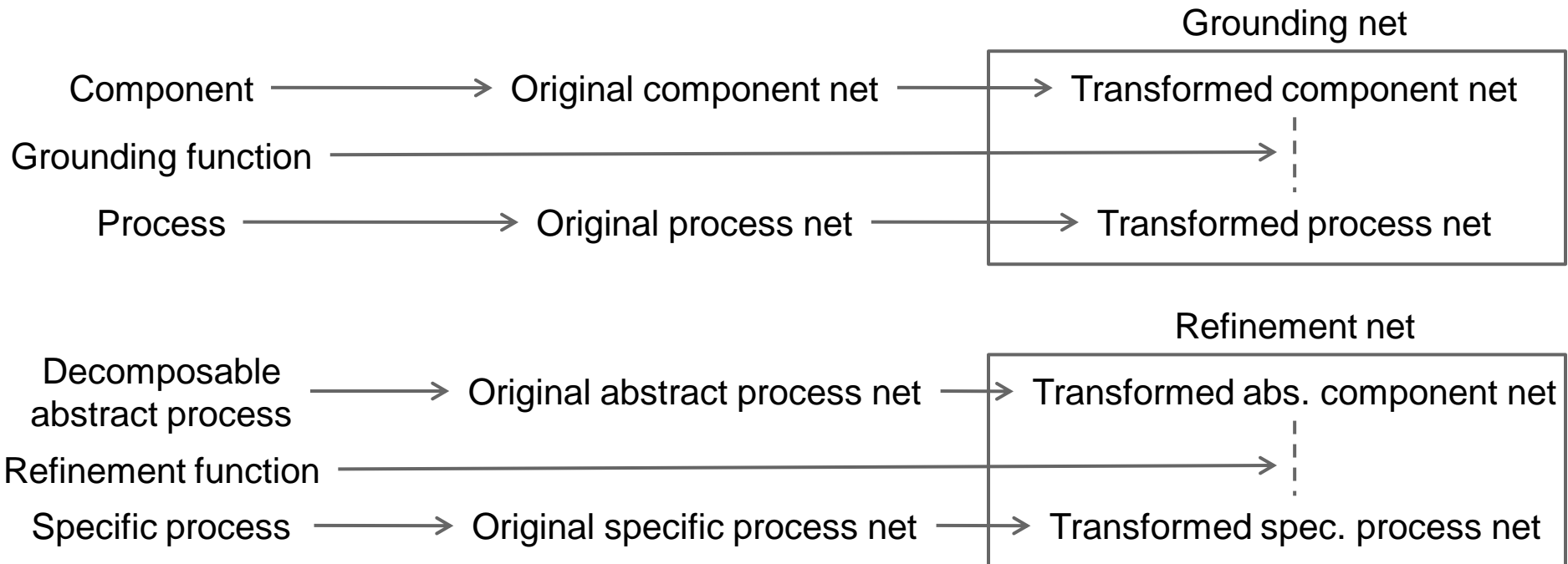
## Petri Net Solution

Overview & Grounding Validation

Two steps for transformation:

## From a process or component model to a Petri net

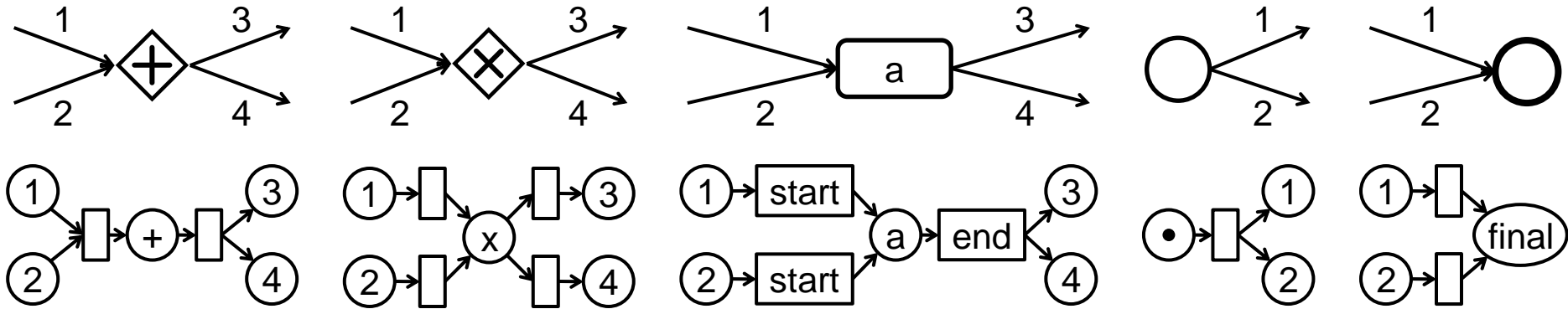
## Constructing the grounding or refinement net



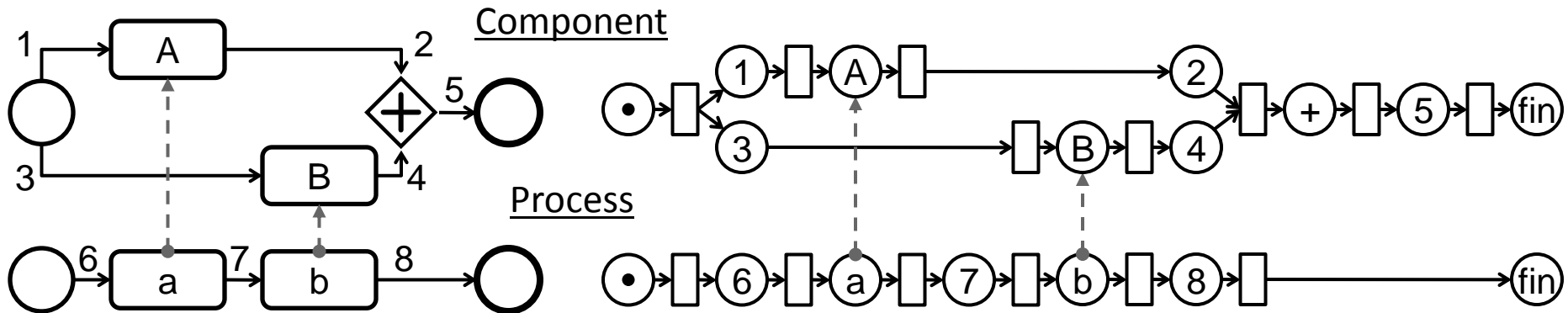
# From a process or component model to a Petri net



Transformation for every basic BPMN construct supported:



Example:

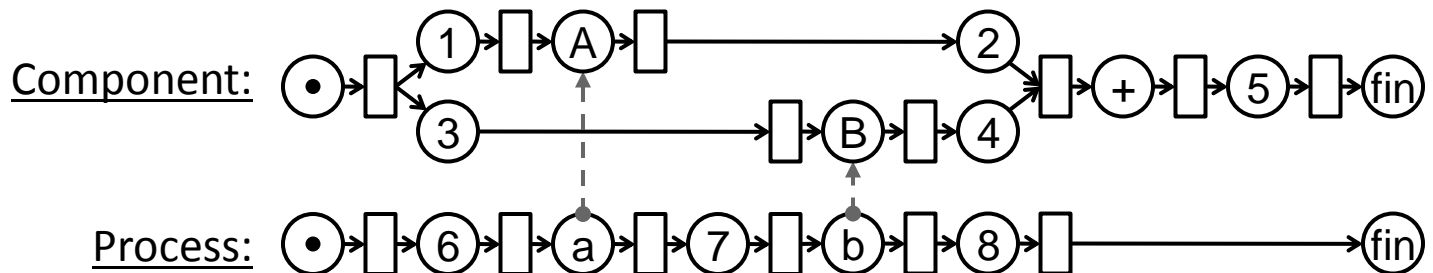


Idea:

- Compare execution sets by simulating synchronized execution of component and process
- Create Petri net that deadlocks if grounding is incorrect

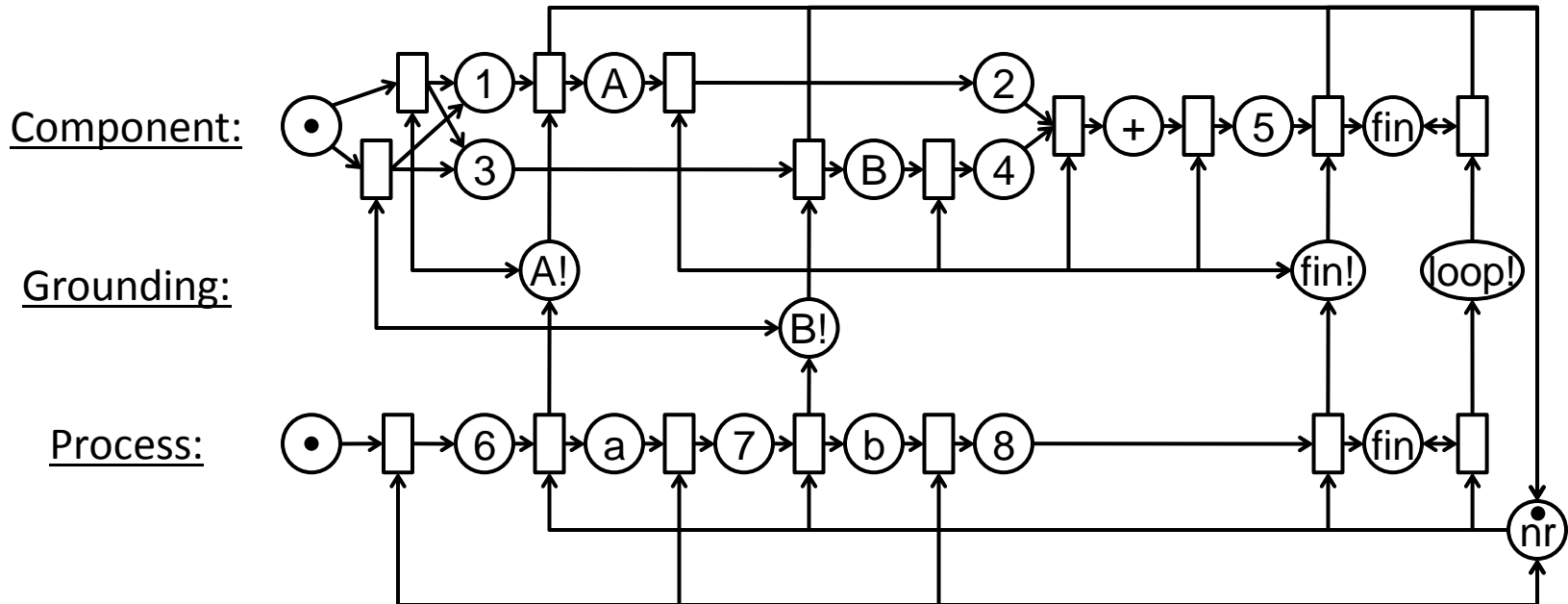
## → Desired properties of Petri net:

- Every step performed by the transformed component (process) model must correspond to BPMN's token flow semantics.
- The next step performed by the transformed component net must be the respective operation o if activity a grounds on o and the previous step performed by the transformed process net was a.
- No deadlock must exist in the transformed component (process) net.
- The transformed component net must never perform a step on its own.
- The transformed process net must be able to perform all allowed executions.
- Transformed component and process net must never have an active transition at the same time.



## Main features

- Notion of requests: request places and no-request place
- Final loop
- Substituted transitions in abstract part



# Part 4

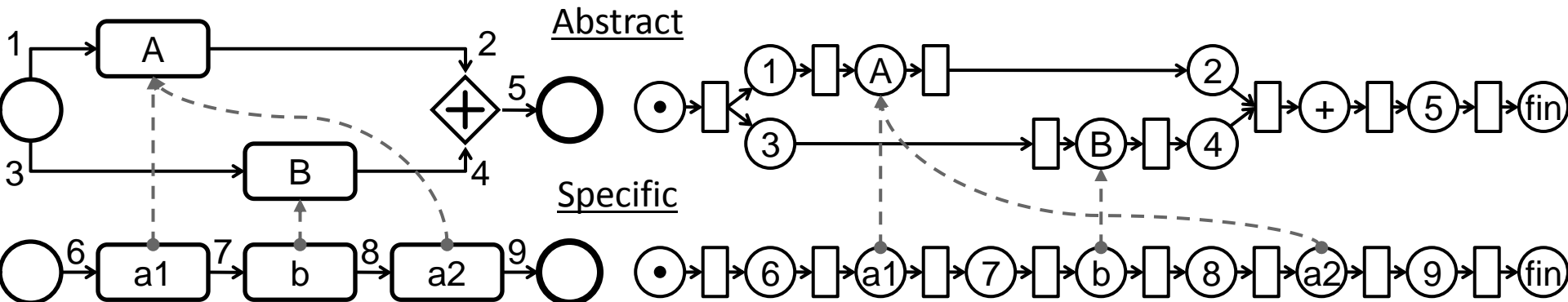
## Petri Net Solution

Constructing the refinement net

Slightly changed properties reflect additional loops in decomposable process

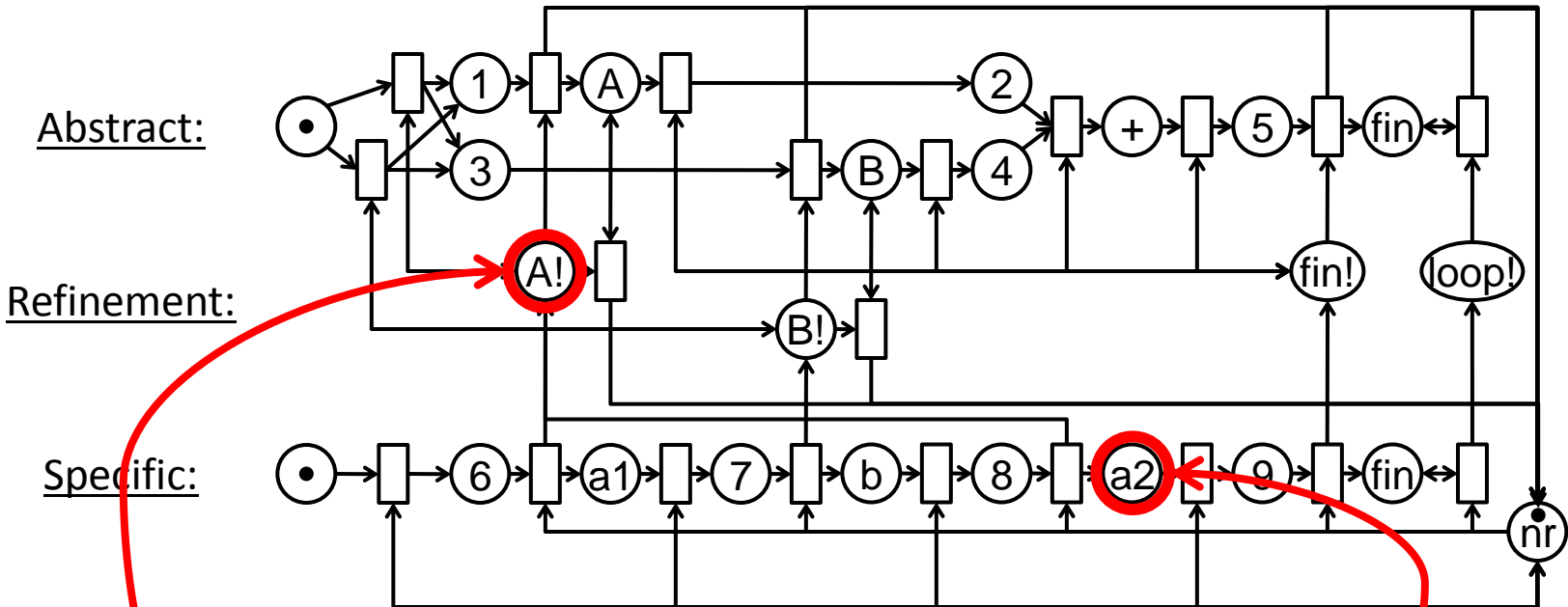
- The next step performed by the transformed abstract process net must be the abstract activity *abs* if the specific activity *spec* refines *abs* and the previous step performed by the transformed specific process net was *spec*.
- **If the transformed abstract process already performs *abs*, the transformed abstract process may also stay in that activity.**

Different example (containing refinement):



Main new feature:

- Still-in transitions



**If a deadlock would occur, the causing abstract & specific activities can be determined**

# Part 4

## Petri Net Solution

Evaluation

## Properties of deadlock detection in safe Petri nets

- Theoretical complexity is exponential (PSPACE-complete)
- State space explosion in presence of parallelism

## Specific advantage

- Naturally handles structured, unstructured, and arbitrarily nested blocks

## Specific challenge

- Detects only first error

# Part 5

## OWL-DL Solution

Explanation

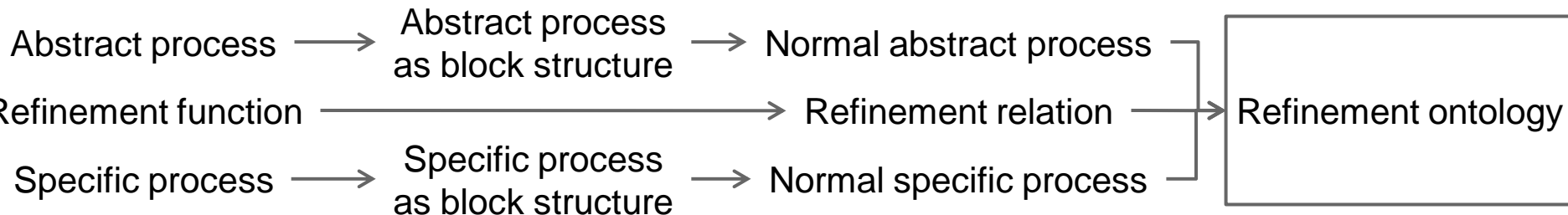
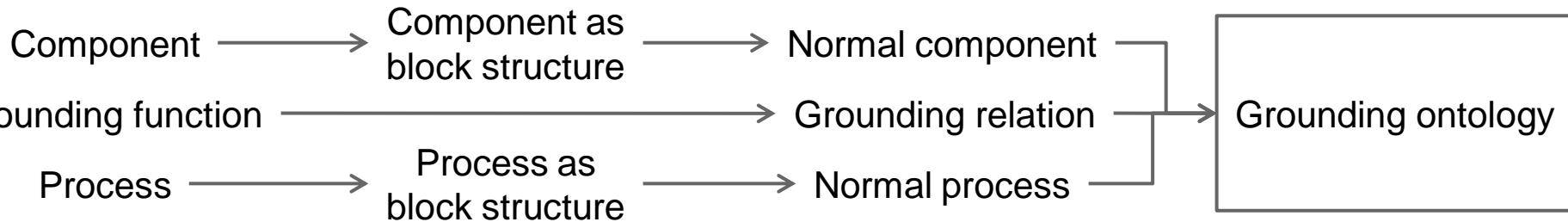
# Ontology can only be created for normal processes



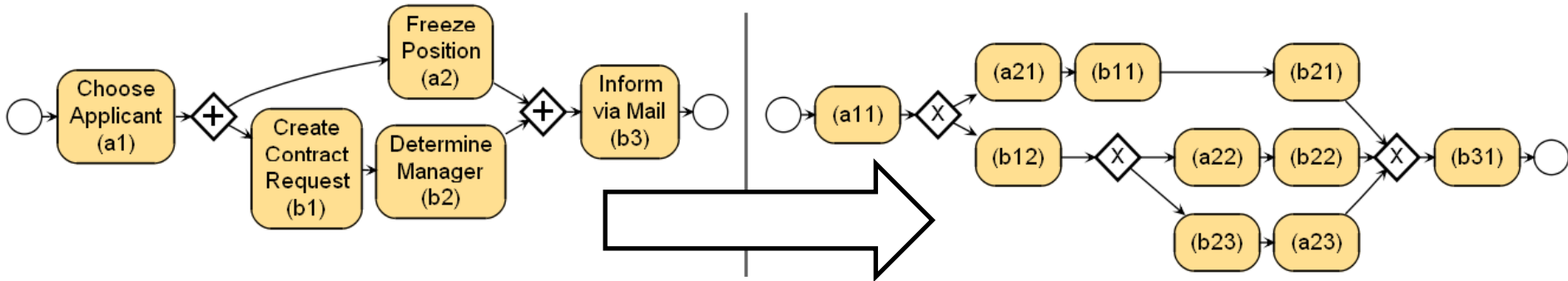
## Block detection (optional)

## Normalization

## Constructing the grounding or refinement ontology



A normal process contains no parallel gateways.



Two ways to normalize a process:

Via block detection

- Only structured blocks
- No nested non-simple blocks in parallel blocks

Via direct gateway substitutions

- No nested loops in parallel flow

➔ **Challenges: Loop handling and factorial complexity:  $O(n!)$**

## Guiding principles

- Operations (activities) are represented as concepts.
- Ordering and composition relations are represented as properties from and to.
- The component (abstract process) is represented by universal restriction.
- The (specific) process is represented by existential restriction.
- Uniqueness is represented by disjointness.

## Four steps

1. Description of the abstract process
2. Description of the specific process
3. Expressing the refinement relation
4. Asserting uniqueness

Description of the abstract process:

Start  $\sqsubseteq \forall$  to. A

A  $\sqsubseteq \forall$  to. (A  $\sqcup$  B)

B  $\sqsubseteq \forall$  to. (B  $\sqcup$  End)

A  $\sqsubseteq \forall$  from. (Start  $\sqcup$  A)

B  $\sqsubseteq \forall$  from. (A  $\sqcup$  B)

End  $\sqsubseteq \forall$  from. B

Description of the specific process:

Start  $\sqsubseteq \exists$  to. a11

a11  $\sqsubseteq (\exists$  to. a21)  $\cap$  ( $\exists$  to. b12)

a21  $\sqsubseteq \exists$  to. b11

b12  $\sqsubseteq (\exists$  to. a22)  $\cap$  ( $\exists$  to. b23)

a11  $\sqsubseteq \exists$  from. Start

a21  $\sqsubseteq \exists$  from. a11

b12  $\sqsubseteq \exists$  from. a11

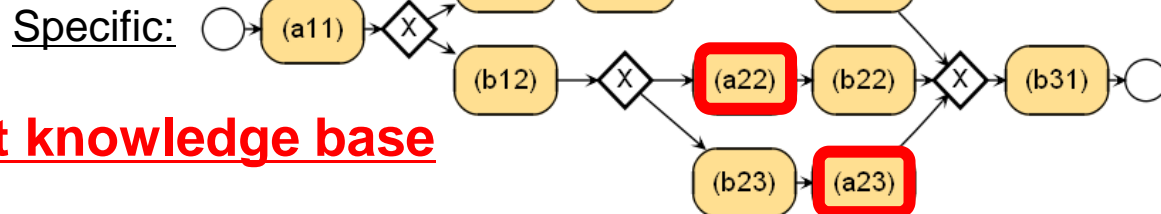
b11  $\sqsubseteq \exists$  from. a21

Expressing the refinement relation

- a11  $\sqsubseteq$  A, a21  $\sqsubseteq$  A,
- a22  $\sqsubseteq$  A, b11  $\sqsubseteq$  B, b12  $\sqsubseteq$  B
- ...

Asserting uniqueness

- disjoint (Start, A, B, End)
- disjoint (a11, a21, a22)
- disjoint (b11, b12, b23, ...)



**Inconsistent knowledge base**

# Part 5

## OWL-DL Solution

Evaluation

## Properties of normalization

- Factorial explosion of normal process in presence of parallelism

## Properties of OWL-DL reasoning (see <http://www.cs.man.ac.uk/~ezolin/dl/>)

- Complexity depends on complexity of logic fragment
- Particular fragment: ALC; theoretical complexity is exponential (PSPACE-complete)

## Specific advantage

- Detects all errors immediately

## Specific challenge

- No solution for normalizing parallel flow with nested loops
- Factorial complexity of transformation

# Part 6

## Comparison

OWL-DL was developed for **static** semantics

- Expensive transformation of parallel flow necessary
- Finding a suitable translation of grounding and refinement validation to OWL-DL reasoning is not straight-forward (multiple normalization approaches exist, each having pros & cons)
- ➔ **Restriction of model complexity (with current approaches)**
- However, all violations are found immediately

Petri net formalism was developed for **dynamic** semantics

- Native constructs for alternatives, parallelism, and loops
- Natural translation of grounding and refinement validation to Petri net analysis
- ➔ **No restrictions of model complexity**
- However, only first violation can be reported

Both approaches need two rough steps:

- Creating the formal model: the Petri net vs. the OWL-DL TBox
- The reasoning: Deadlock detection in Petri nets vs. classification reasoning in OWL-DL

Petri net approach

- Creating the formal model: Quadratic,  $O(n^2)$
  - The reasoning: PSpace-complete (exponential),  $O(e^n)$
- **Overall complexity: PSpace-complete**

OWL-DL approach

- Creating the formal model: Factorial,  $O(n!)$
  - The reasoning: PSpace-complete (exponential),  $O(e^n)$
- **Overall complexity: Factorial**
- However, in practice, reasoning consumes more resources than normalization (see next slide)

# Practical performance of OWL-DL approach and potential for improvement

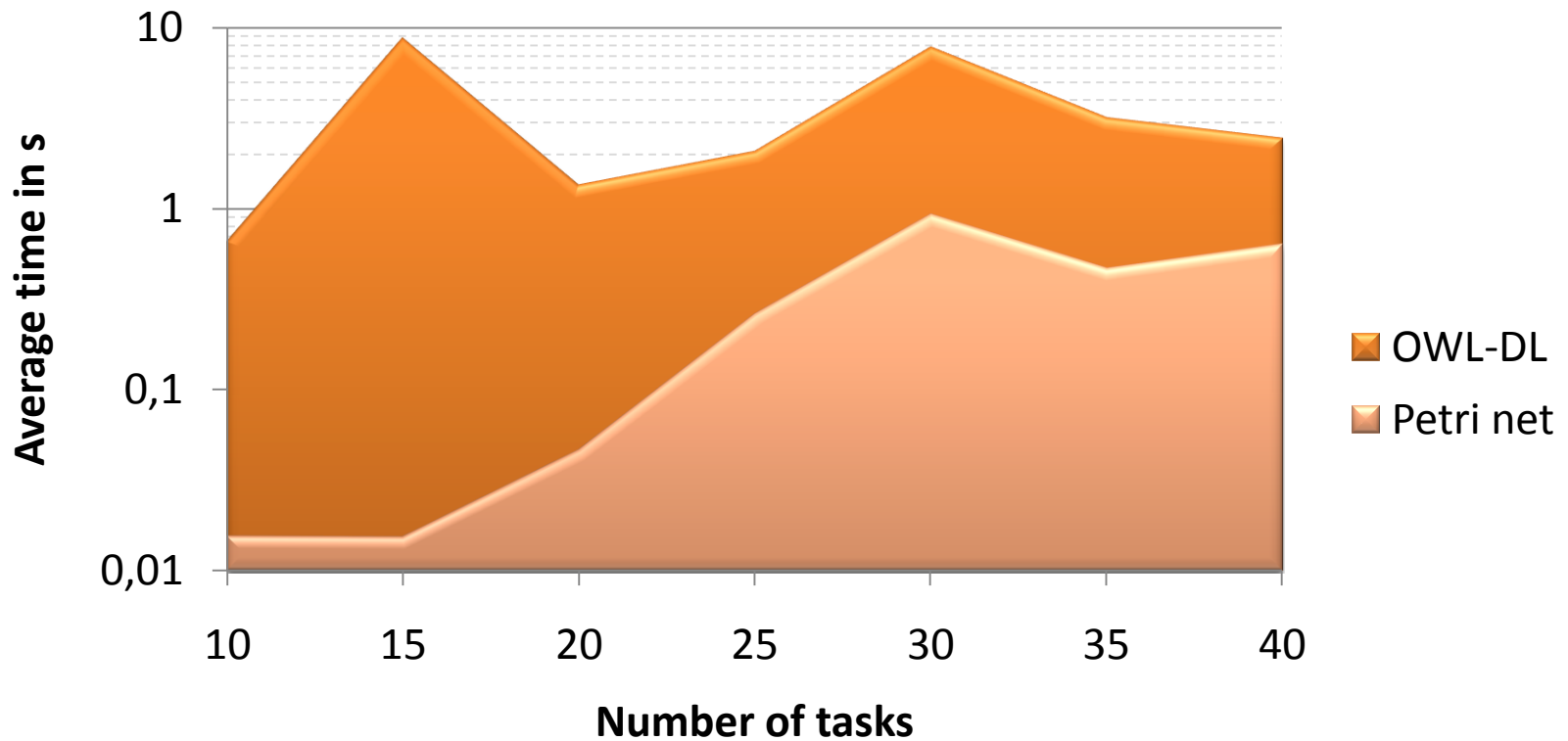
Experiment with 1239 generated abstract and specific processes:

	Total activities	Transformation time	Reasoning time
<b>Average</b>	23	4 ms (0.1%)	2.8 s (99.9%)
<b>Maximum</b>	69	0.4 s (0.2%)	3.4 min (99.8%)

- Although transformation has a worse computational complexity, reasoning takes a substantial portion (> 99%) of the overall time.
- Factorial theoretical complexity irrelevant in practice
- Reasoner efficiency has largest potential for improvement

We analyzed 300 refinement scenarios

→ Concerning practical reasoning time, both approaches are close together



## Petri net approach

- Creating the Petri net
  - Optimization of irrelevant patterns in Petri net shortens time of analysis
- Reasoning
  - Explore ways to report not only first error, for example, by fixing the error like in compiler construction

## OWL-DL approach

- Normalization
  - Concentrate on broader coverage of all kinds of nested flows
- Reasoning
  - Concentrate on more efficient reasoning
  - For example, by approximation techniques as implemented in REL (Univ. Aberdeen)

Manual grounding and refinement validation are tedious and error-prone.

Automatic grounding and refinement validation...

- Benefit the business process engineer.
- Can be handled naturally by Petri net analysis.
- Can be mapped on OWL-DL reasoning.

Both mappings have advantages and limitations due to the complexity of the problem.

Further research may improve the particular challenges of either approach.

Both approaches can benefit from understanding challenges and limitations of the other.

# Thank you!

Additional information: [jens.lemcke@sap.com](mailto:jens.lemcke@sap.com)

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

SAP, R/3, mySAP, mySAP.com, xApps, xApp, SAP NetWeaver, Duet, Business ByDesign, ByDesign, PartnerEdge and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other product and service names mentioned and associated logos displayed are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.

The information in this document is proprietary to SAP. This document is a preliminary version and not subject to your license agreement or any other agreement with SAP. This document contains only intended strategies, developments, and functionalities of the SAP® product and is not intended to be binding upon SAP to any particular course of business, product strategy, and/or development. SAP assumes no responsibility for errors or omissions in this document. SAP does not warrant the accuracy or completeness of the information, text, graphics, links, or other items contained within this material. This document is provided without a warranty of any kind, either express or implied, including but not limited to the implied warranties of merchantability, fitness for a particular purpose, or non-infringement.

SAP shall have no liability for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials. This limitation shall not apply in cases of intent or gross negligence.

The statutory liability for personal injury and defective products is not affected. SAP has no control over the information that you may access through the use of hot links contained in these materials and does not endorse your use of third-party Web pages nor provide any warranty whatsoever relating to third-party Web pages

Weitergabe und Vervielfältigung dieser Publikation oder von Teilen daraus sind, zu welchem Zweck und in welcher Form auch immer, ohne die ausdrückliche schriftliche Genehmigung durch SAP AG nicht gestattet. In dieser Publikation enthaltene Informationen können ohne vorherige Ankündigung geändert werden.

Einige von der SAP AG und deren Vertriebspartnern vertriebene Softwareprodukte können Softwarekomponenten umfassen, die Eigentum anderer Softwarehersteller sind.

SAP, R/3, mySAP, mySAP.com, xApps, xApp, SAP NetWeaver, Duet, Business ByDesign, ByDesign, PartnerEdge und andere in diesem Dokument erwähnte SAP-Produkte und Services sowie die dazugehörigen Logos sind Marken oder eingetragene Marken der SAP AG in Deutschland und in mehreren anderen Ländern weltweit. Alle anderen in diesem Dokument erwähnten Namen von Produkten und Services sowie die damit verbundenen Firmenlogos sind Marken der jeweiligen Unternehmen. Die Angaben im Text sind unverbindlich und dienen lediglich zu Informationszwecken. Produkte können länderspezifische Unterschiede aufweisen.

Die in diesem Dokument enthaltenen Informationen sind Eigentum von SAP. Dieses Dokument ist eine Vorabversion und unterliegt nicht Ihrer Lizenzvereinbarung oder einer anderen Vereinbarung mit SAP. Dieses Dokument enthält nur vorgesehene Strategien, Entwicklungen und Funktionen des SAP®-Produkts und ist für SAP nicht bindend, einen bestimmten Geschäftsweg, eine Produktstrategie bzw. -entwicklung einzuschlagen. SAP übernimmt keine Verantwortung für Fehler oder Auslassungen in diesen Materialien. SAP garantiert nicht die Richtigkeit oder Vollständigkeit der Informationen, Texte, Grafiken, Links oder anderer in diesen Materialien enthaltenen Elemente. Diese Publikation wird ohne jegliche Gewähr, weder ausdrücklich noch stillschweigend, bereitgestellt. Dies gilt u. a., aber nicht ausschließlich, hinsichtlich der Gewährleistung der Marktgängigkeit und der Eignung für einen bestimmten Zweck sowie für die Gewährleistung der Nichtverletzung geltenden Rechts.

SAP übernimmt keine Haftung für Schäden jeglicher Art, einschließlich und ohne Einschränkung für direkte, spezielle, indirekte oder Folgeschäden im Zusammenhang mit der Verwendung dieser Unterlagen. Diese Einschränkung gilt nicht bei Vorsatz oder grober Fahrlässigkeit.

Die gesetzliche Haftung bei Personenschäden oder die Produkthaftung bleibt unberührt. Die Informationen, auf die Sie möglicherweise über die in diesem Material enthaltenen Hotlinks zugreifen, unterliegen nicht dem Einfluss von SAP, und SAP unterstützt nicht die Nutzung von Internetseiten Dritter durch Sie und gibt keinerlei Gewährleistungen oder Zusagen über Internetseiten Dritter ab.

Alle Rechte vorbehalten.