# ER12-Tutorial: Enabling Flexibility in Process-aware Information Systems

## Challenges, Methods, Technologies

1



BARBARA WEBER
UNIVERSITY OF INNSBRUCK

MANFRED REICHERT
ULM UNIVERSITY

ER 2012, FLORENCE

WWW.PROCESS-FLEXIBILITY.COM

© B. Weber and M. Reichert, 2012

# Content



- Part 1 – Process-aware Information Systems
- Part 2 – Flexibility Issues
- Part 3– Flexibility Support for Pre-specified Process Models
  - Pre-specified process models and flexibility-by-design
  - Process configuration
  - Flexible process execution and handling of anticipated exceptions
  - Handling unforeseen exceptions
  - Process Evolution
- Part 4– Loosely-specified Process Models
  - Loosely-specified process models
  - Constraint-based process models

© B. Weber and M. Reichert, 2012

# Business Processes and Workflows

## Process-aware Information Systems

3

BARBARA WEBER
UNIVERSITY OF INNSBRUCK

MANFRED REICHERT
ULM UNIVERSITY

ER 2012, FLORENCE

© B. Weber and M. Reichert, 2012

# A Retail Process




Welcome customer

Offer Clothes


Bill Clothes

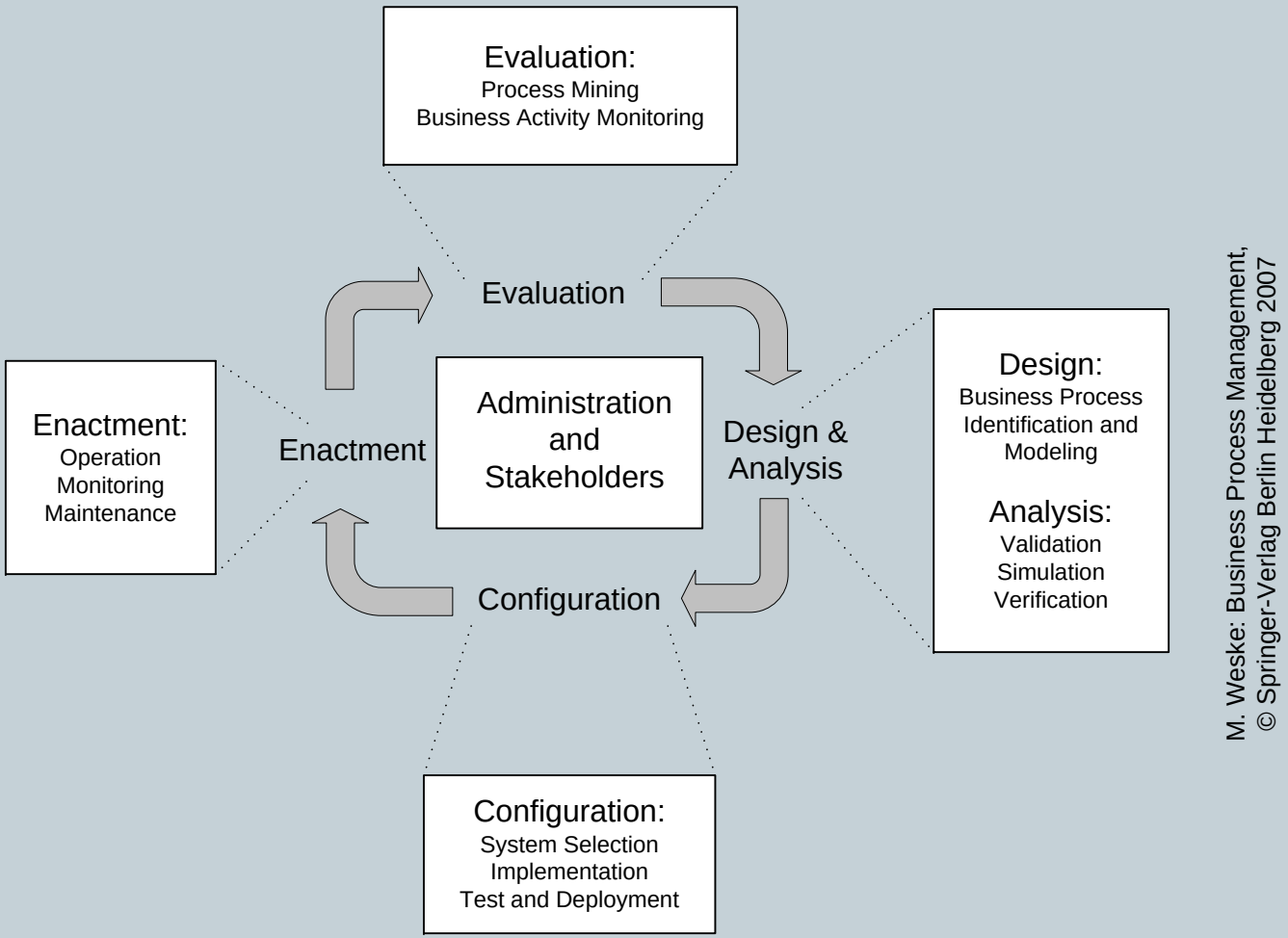

Hand over clothes

Mendling 2006

© B. Weber and M. Reichert, 2012

# Business Process Lifecycle



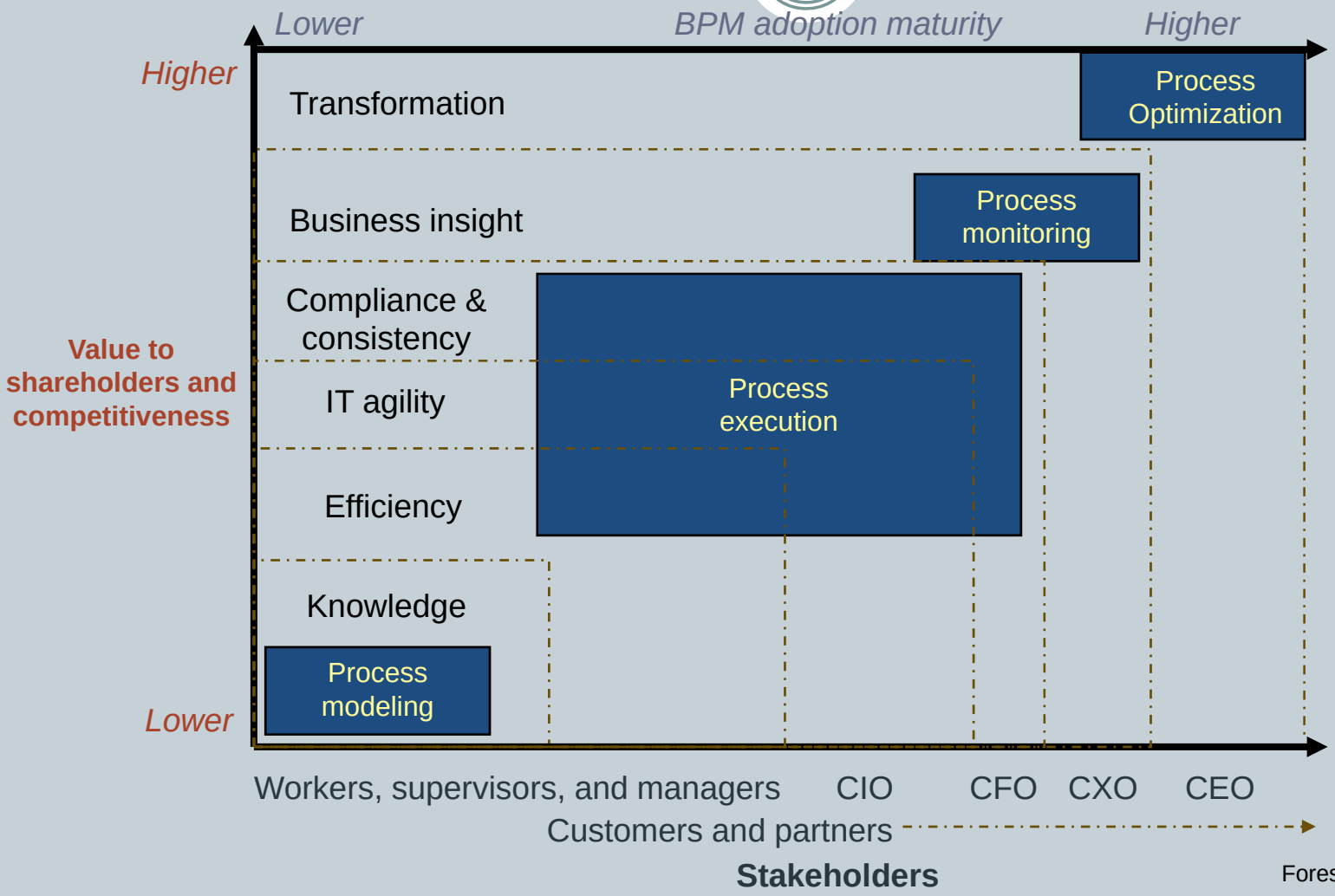



M. Weske: Business Process Management, © Springer-Verlag Berlin Heidelberg 2007

© B. Weber and M. Reichert, 2012

# BPM Value Proposition



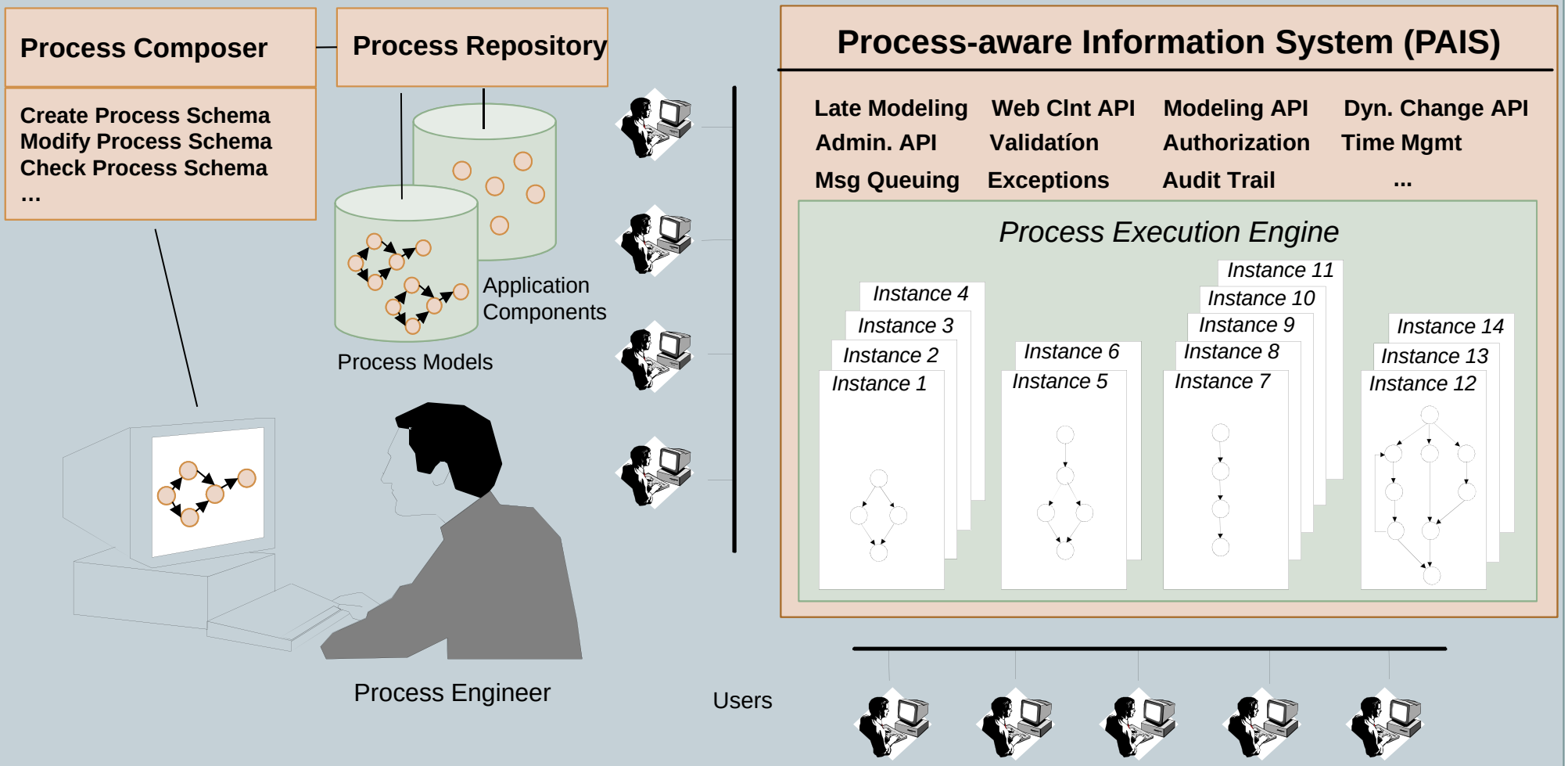



Forester 2007 BPM Market Overview

© B. Weber and M. Reichert, 2012

# Process-aware Information System



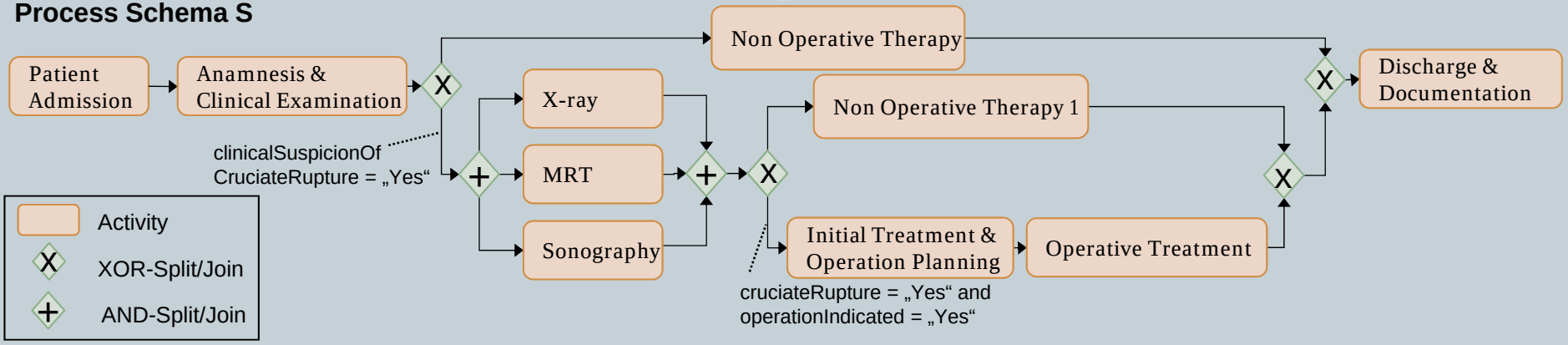



© B. Weber and M. Reichert, 2012

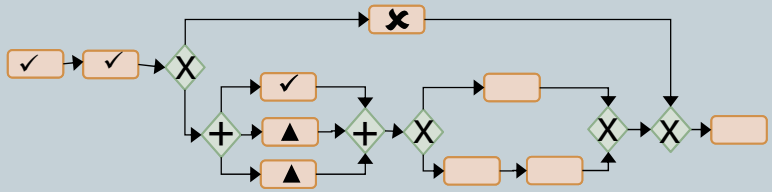
# Business Process – System Perspective



**Process Schema S**

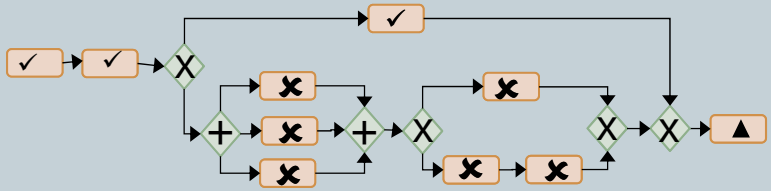



**Process Instance I1**



**Execution Trace:**
$\sigma_1$ = < „Patient Admission“, „Anamnesis & Clinical Examination“, „X-ray“>

**Process Instance I2**



**Execution Trace:**
$\sigma_2$ = < „Patient Admission“, „Anamnesis & Clinical Examination“, „Non Operative Therapy“>

**Activity States:** ▲ Enabled ✓ Completed ✘ Skipped

© B. Weber and M. Reichert, 2012

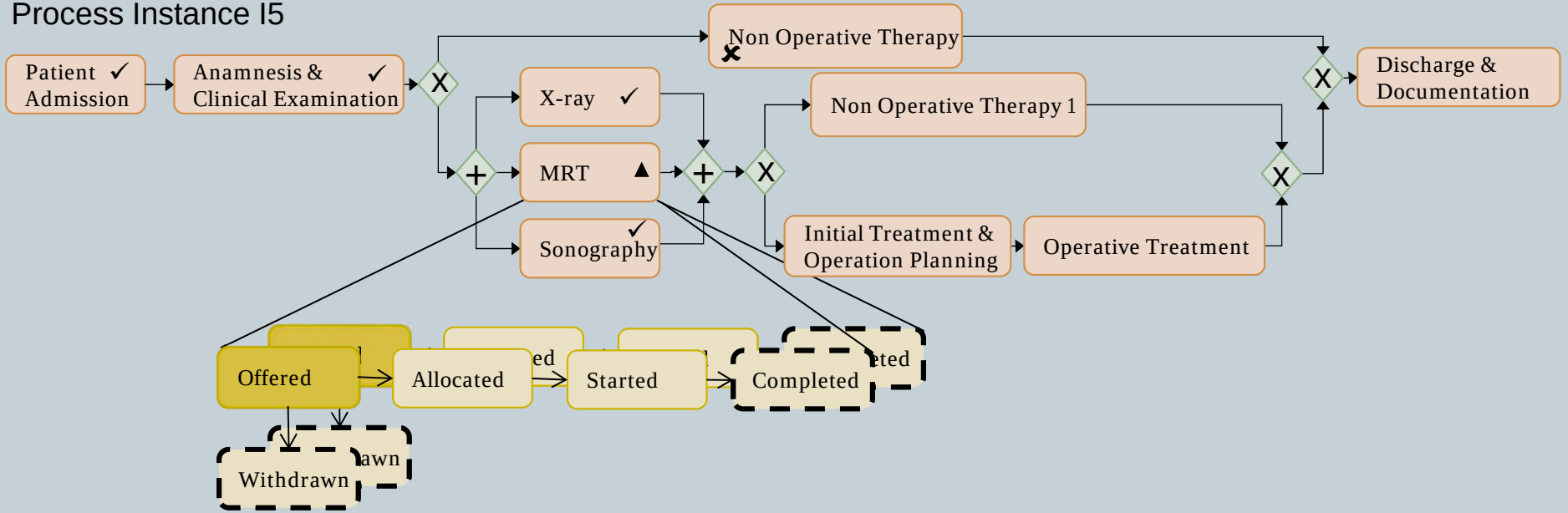
# User Perspective




Joe


Peter

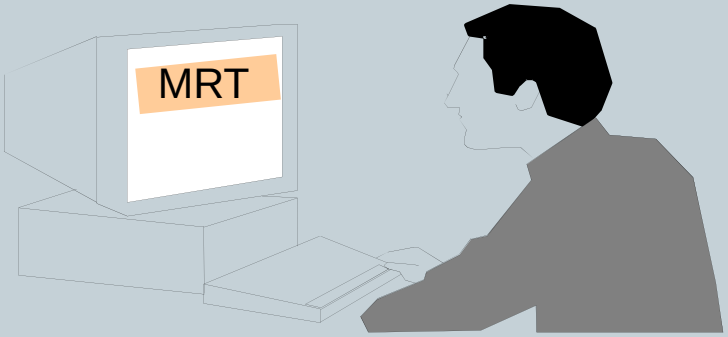
Process Instance I5




© B. Weber and M. Reichert, 2012

# User Perspective

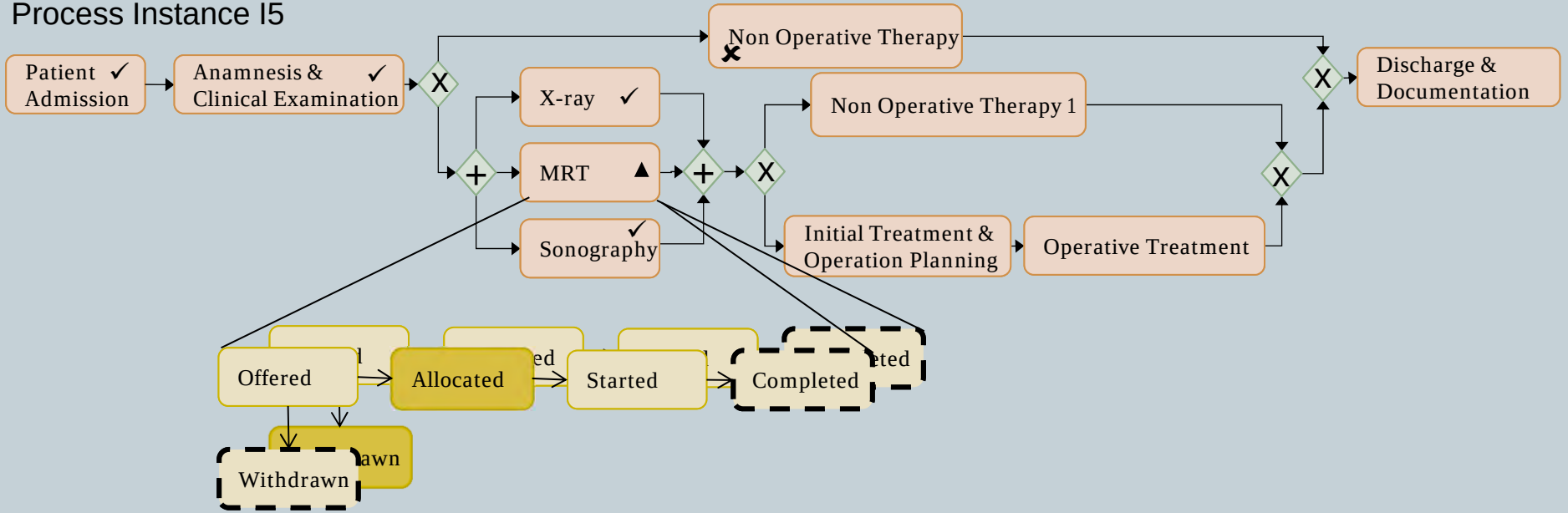



Joe

Peter

Process Instance I5




© B. Weber and M. Reichert, 2012

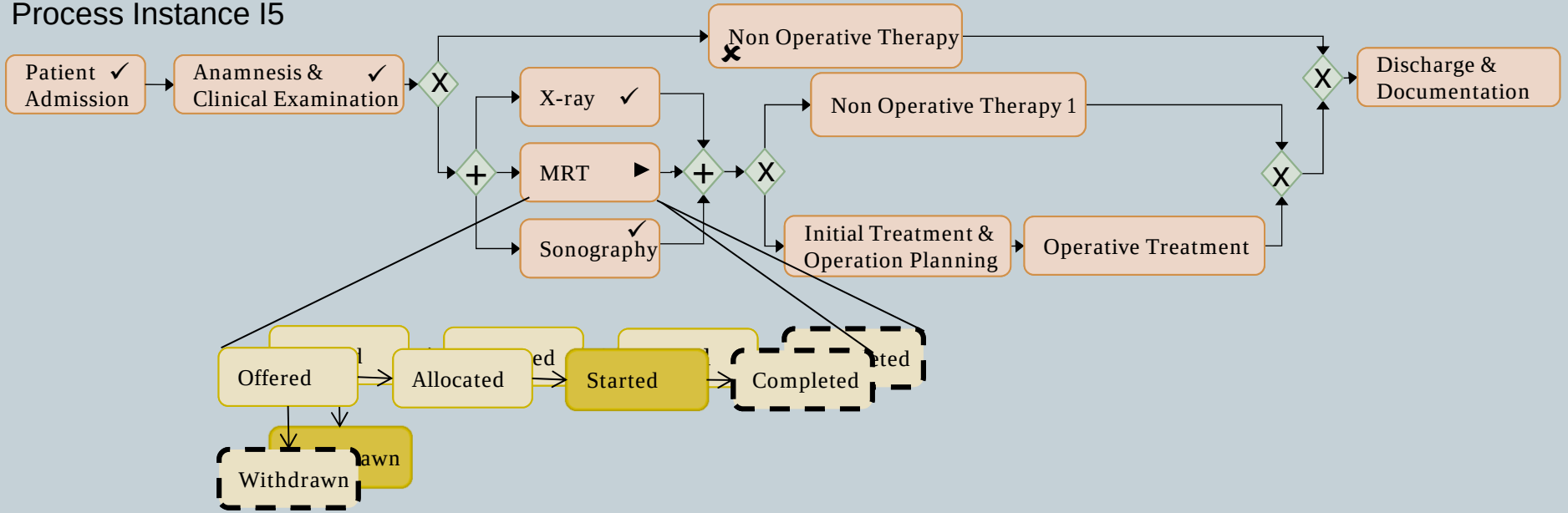
# User Perspective



Joe

Peter

Process Instance I5




© B. Weber and M. Reichert, 2012

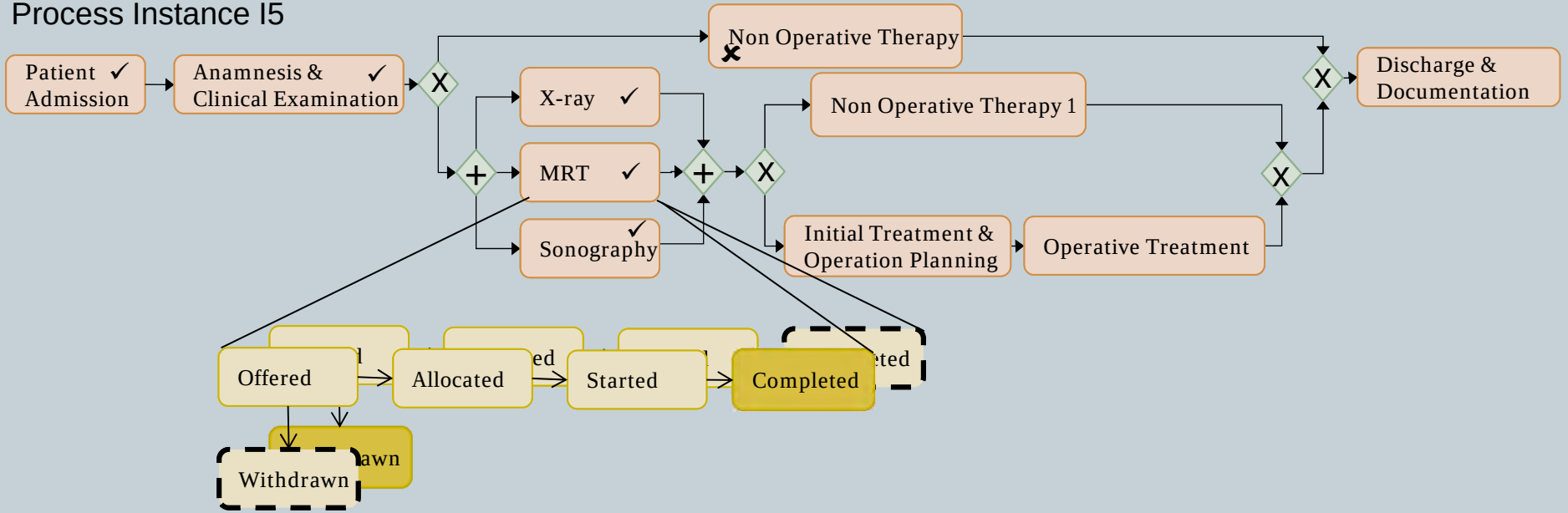
# User Perspective



Joe

Peter

Process Instance I5




© B. Weber and M. Reichert, 2012

# Business Processes and Workflows

## Flexibility Issues

13

BARBARA WEBER
UNIVERSITY OF INNSBRUCK

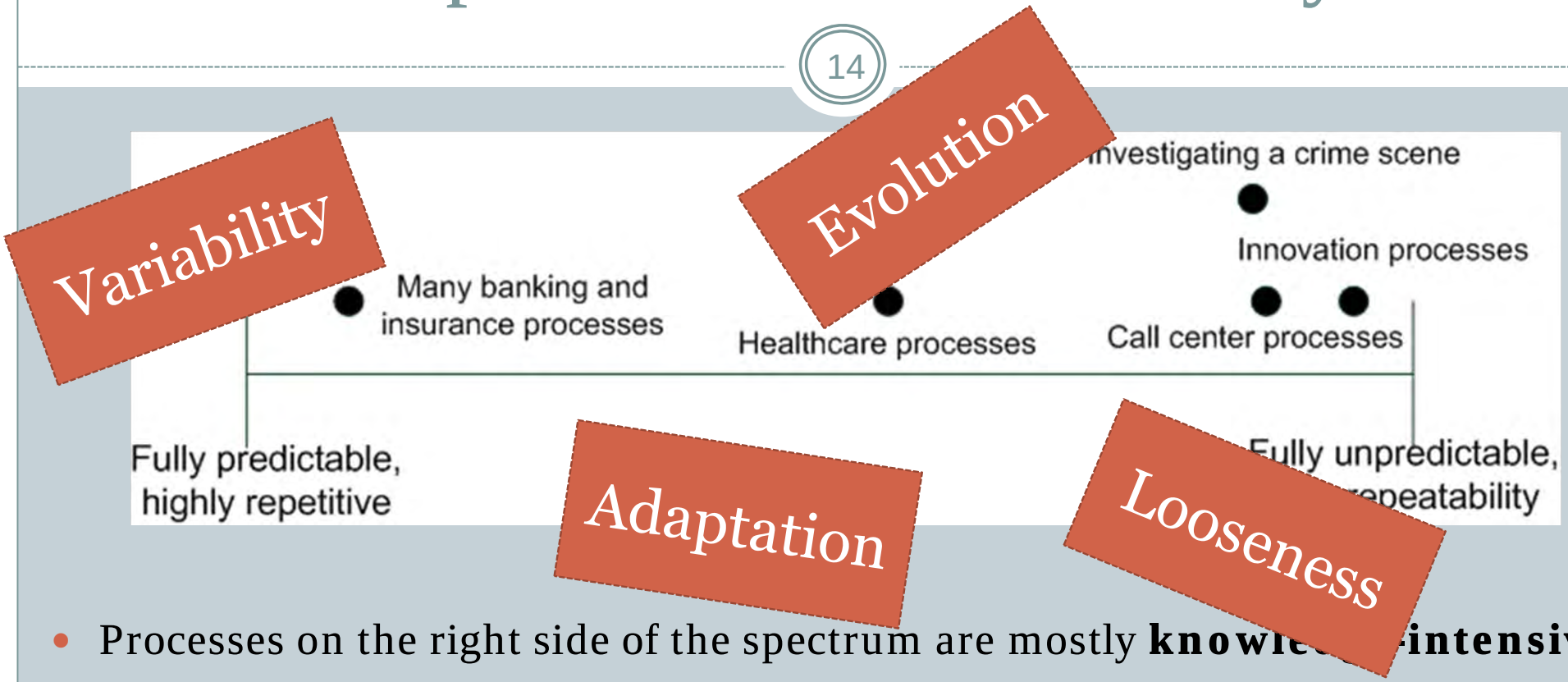
MANFRED REICHERT
ULM UNIVERSITY

ER 2012, FLORENCE

© B. Weber and M. Reichert, 2012

# Process Spectrum and Flexibility Needs






- Processes on the right side of the spectrum are mostly **knowl intensive**
  - **Unpredictability:** Course of action depends on situation-specific parameters
  - **Non-repeatability:** Two process instances hardly look the same
  - **Emergence:** Future course of action depends on knowledge gained through activity execution

© B. Weber and M. Reichert, 2012

# Variability



- Variability is typical for many domains and requires that processes are handled differently depending on the particular context

- Drivers
  - Product and service variability
  - Differences in regulations
  - Different customer groups
  - Temporal differences

Example: Vehicle Repair

© B. Weber and M. Reichert, 2012

# Looseness



- Knowledge-intensive processes cannot be fully pre-specified, but require loose specifications

- Drivers
  - Unpredictability
  - Non-Repeatability
  - Emergence

Example: Treatment Processes in a Hospital

© B. Weber and M. Reichert, 2012

# Adaptation



- Ability to adapt the process and its structure to temporary events
- Drivers
  - Special Situations
  - Exceptions
- Anticipation of Adaptation
  - Planned
  - Unanticipated

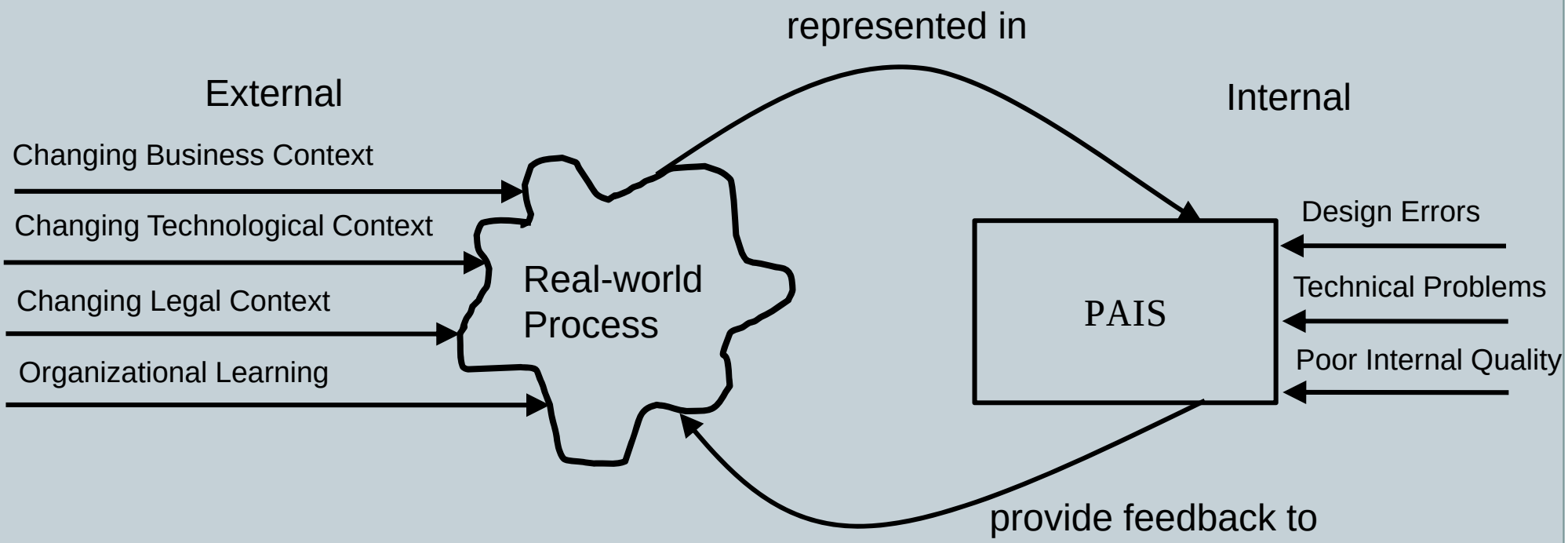
Example: Examination Procedures in a Hospital

© B. Weber and M. Reichert, 2012

# Evolution



- Ability of the implemented process to change when the business process evolves
- Drivers




© B. Weber and M. Reichert, 2012

# Evolution



- Extent of Evolution
  - Incremental
    - Continuous Process Improvement
  - Revolutionary
    - Business Process Reengineering
- Duration
  - Temporary
  - Permanent

© B. Weber and M. Reichert, 2012

# Evolution



- Swiftness
  - Deferred
    - Ongoing instances are not affected
  - Immediate
    - Ongoing instances are affected
- Visibility
  - Observable Behavior
  - Internal Structure

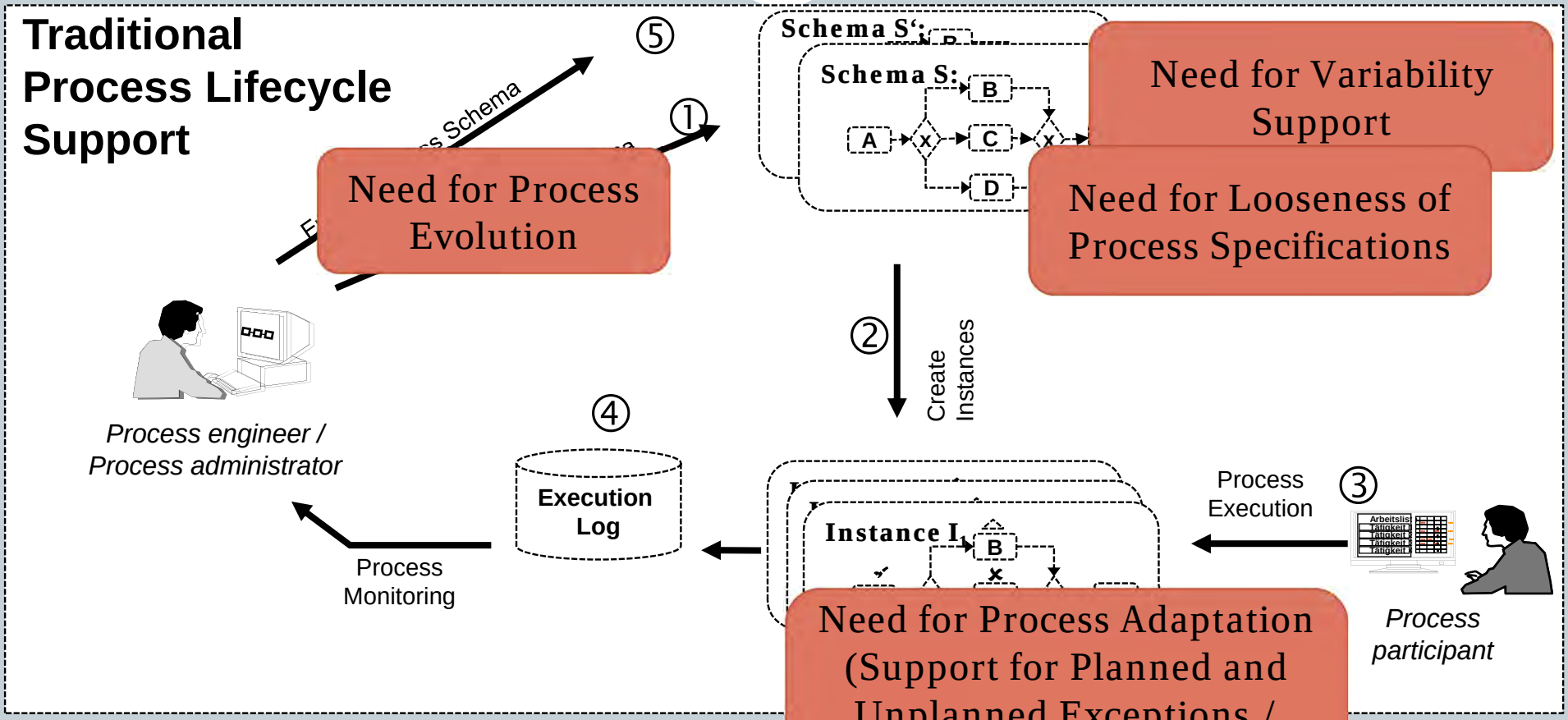
Example: Tender Preparation

Example: Inconsistent Naming of Process Models

© B. Weber and M. Reichert, 2012

# Flexibility Issues along the Process Lifecycle






[WRW+09]

© B. Weber and M. Reichert, 2012

# Flexibility Needs and Technological Requirements



| Flexibility Need | Dimension | Technological Requirement |
|---|---|---|
| Variability | | Configuration |
| Looseness | | Loosely-specified processes |
| Adaptation | Planned<br>Unplanned | Exception Handling<br>Ad-hoc Changes |
| Evolution | Deferred Evolution<br>Immediate Evolution<br>Poor Internal Quality<br>Organizational Learning | Versioning<br>Process Instance Migration<br>Refactoring<br>Monitoring, Analysis and Mining |

© B. Weber and M. Reichert, 2012

# Business Processes and Workflows

## Pre-specified Process Models and Flexibility-by-Design

23

BARBARA WEBER
UNIVERSITY OF INNSBRUCK

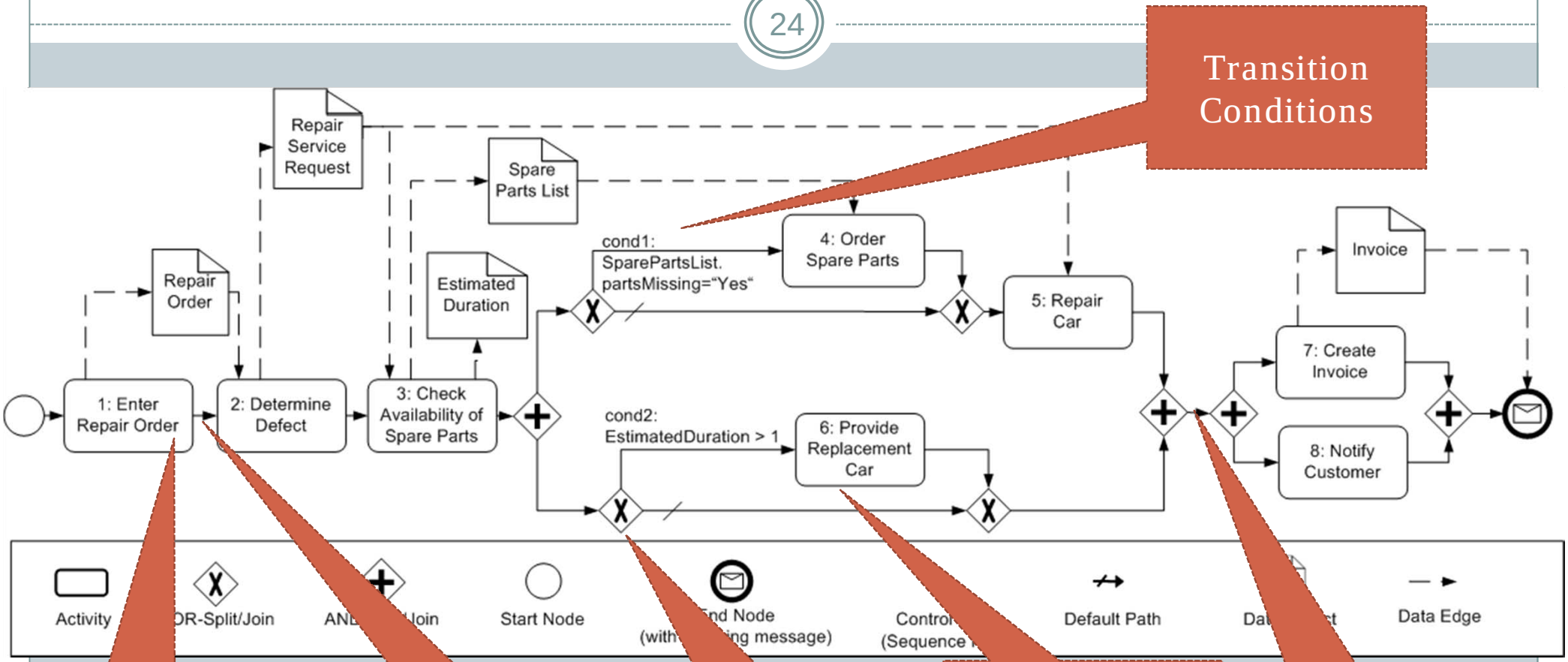
MANFRED REICHERT
ULM UNIVERSITY

ER 2012, FLORENCE

© B. Weber and M. Reichert, 2012

# Basic Control Flow Concepts & Patterns

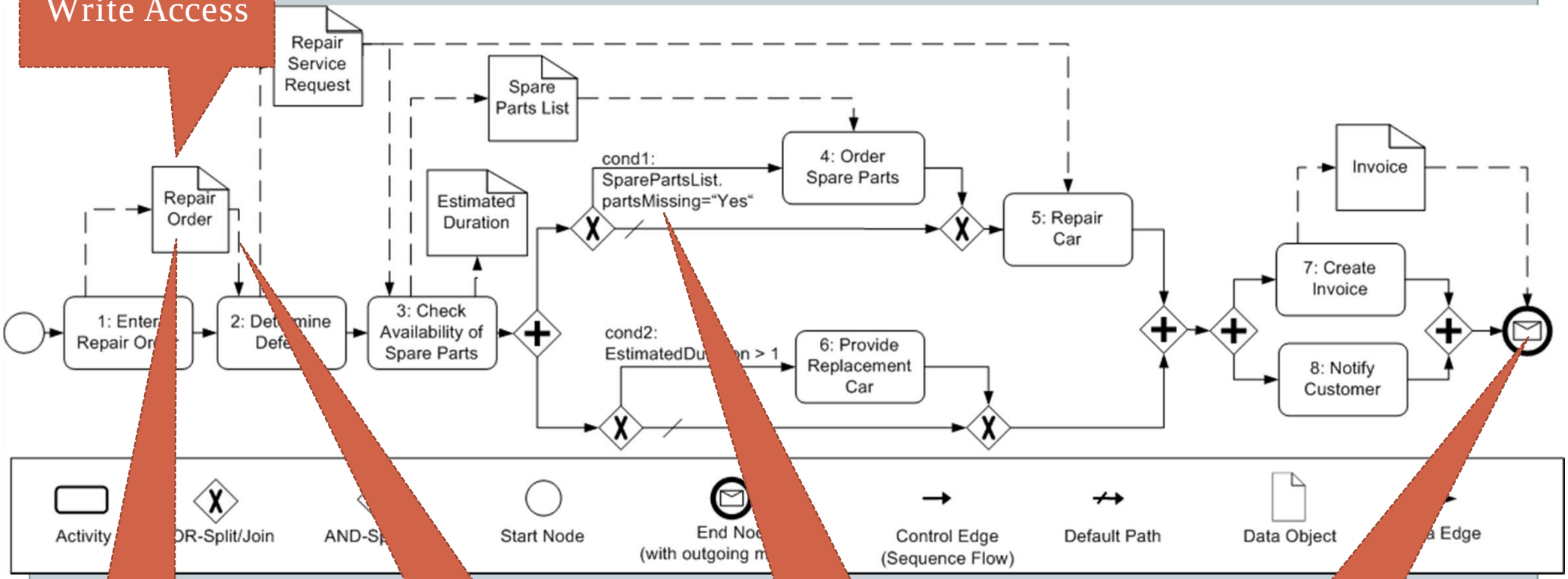





© B. Weber and M. Reichert, 2012

# Basic Data Flow Concepts & Patterns






Data Edge – Write Access

Data Object

Data Edge – Read Access

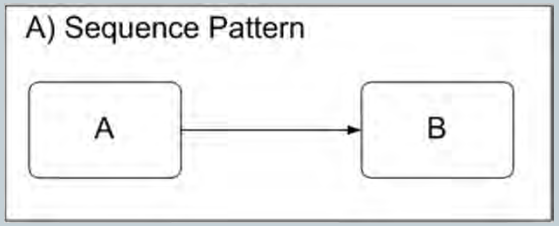
Transition Condition references *SparePartsList*

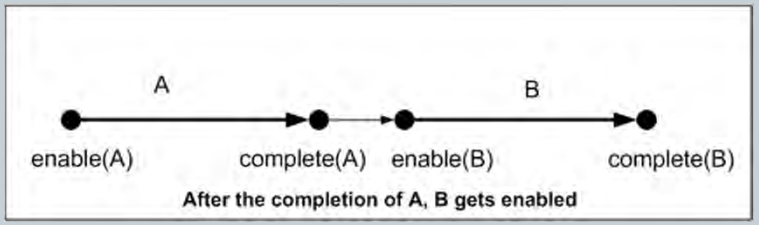
End Message with Data Obejct *Invoice*

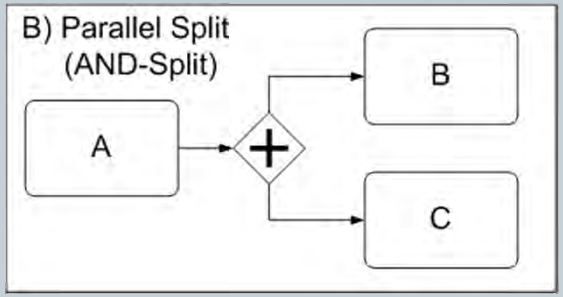
© , 2012

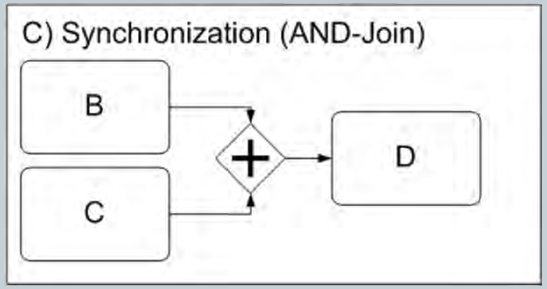
# Examples of Control Flow Patterns (1)

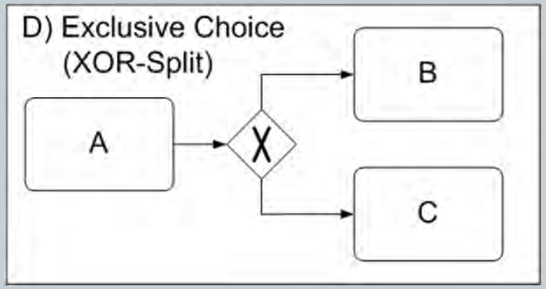


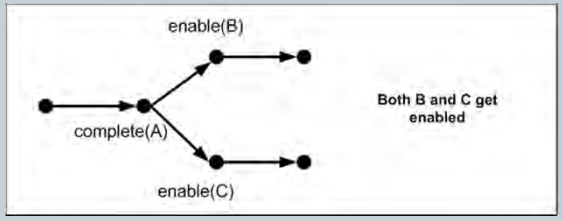


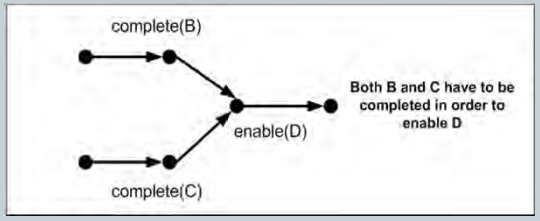



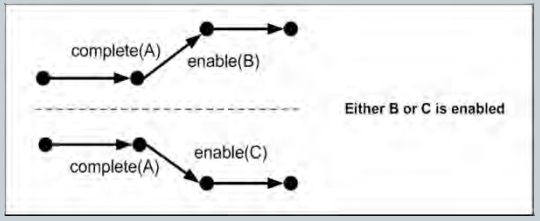



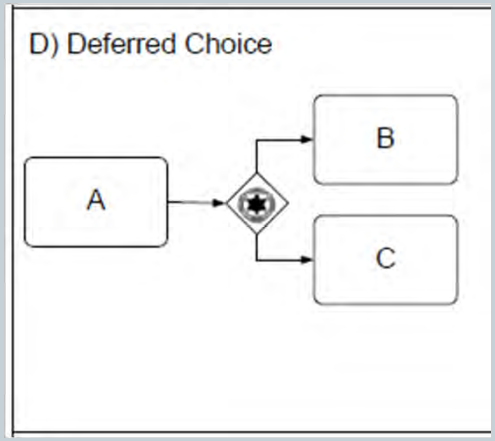



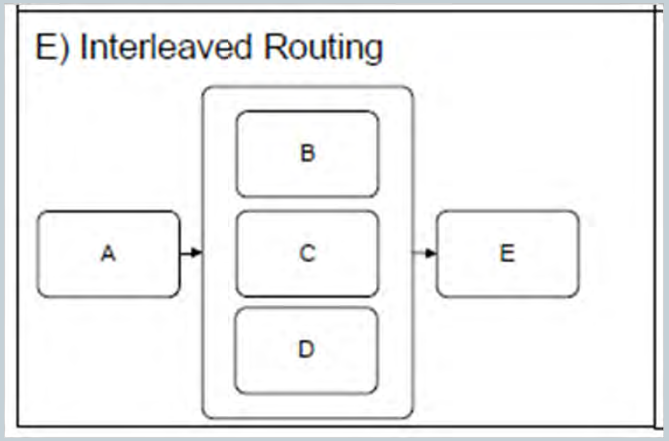



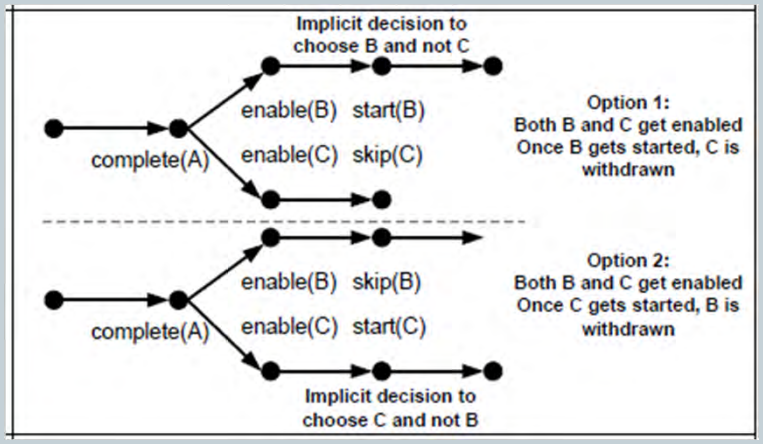

© B. Weber and M. Reichert, 2012

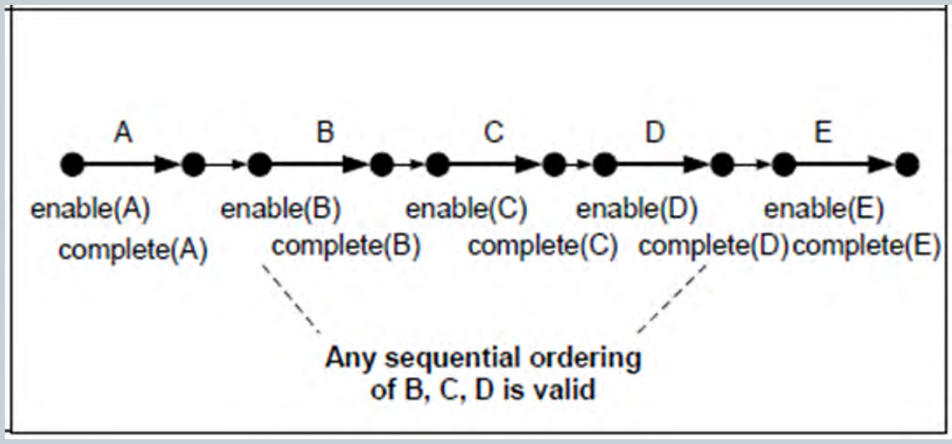
# Examples of Control Flow Patterns (2)





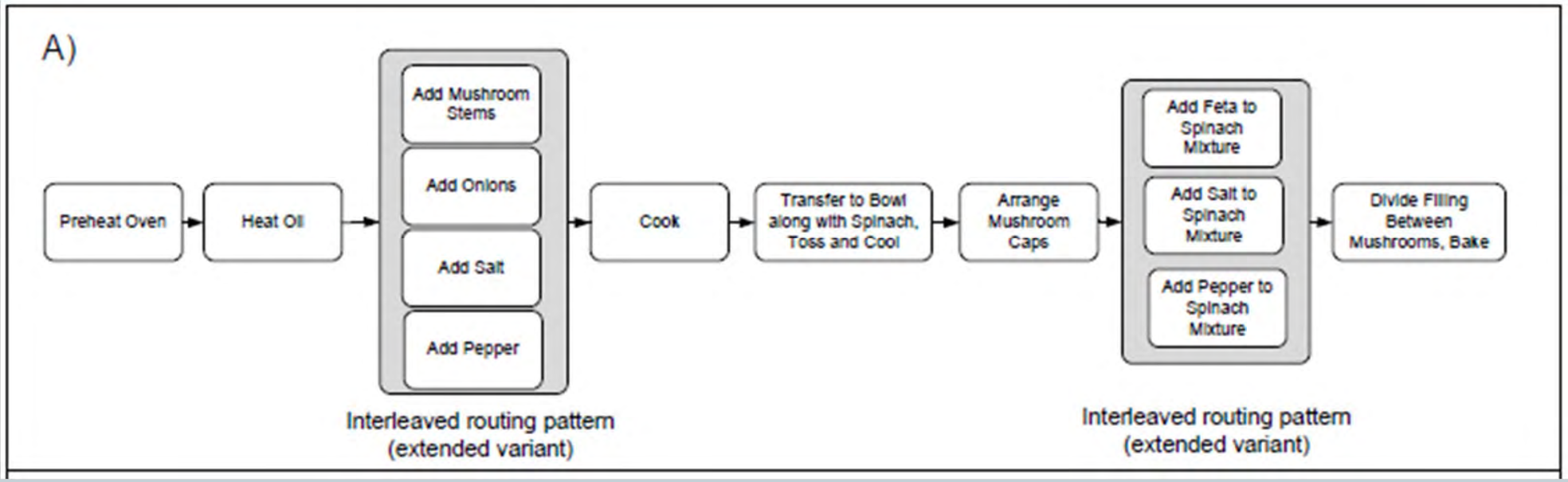




© B. Weber and M. Reichert, 2012

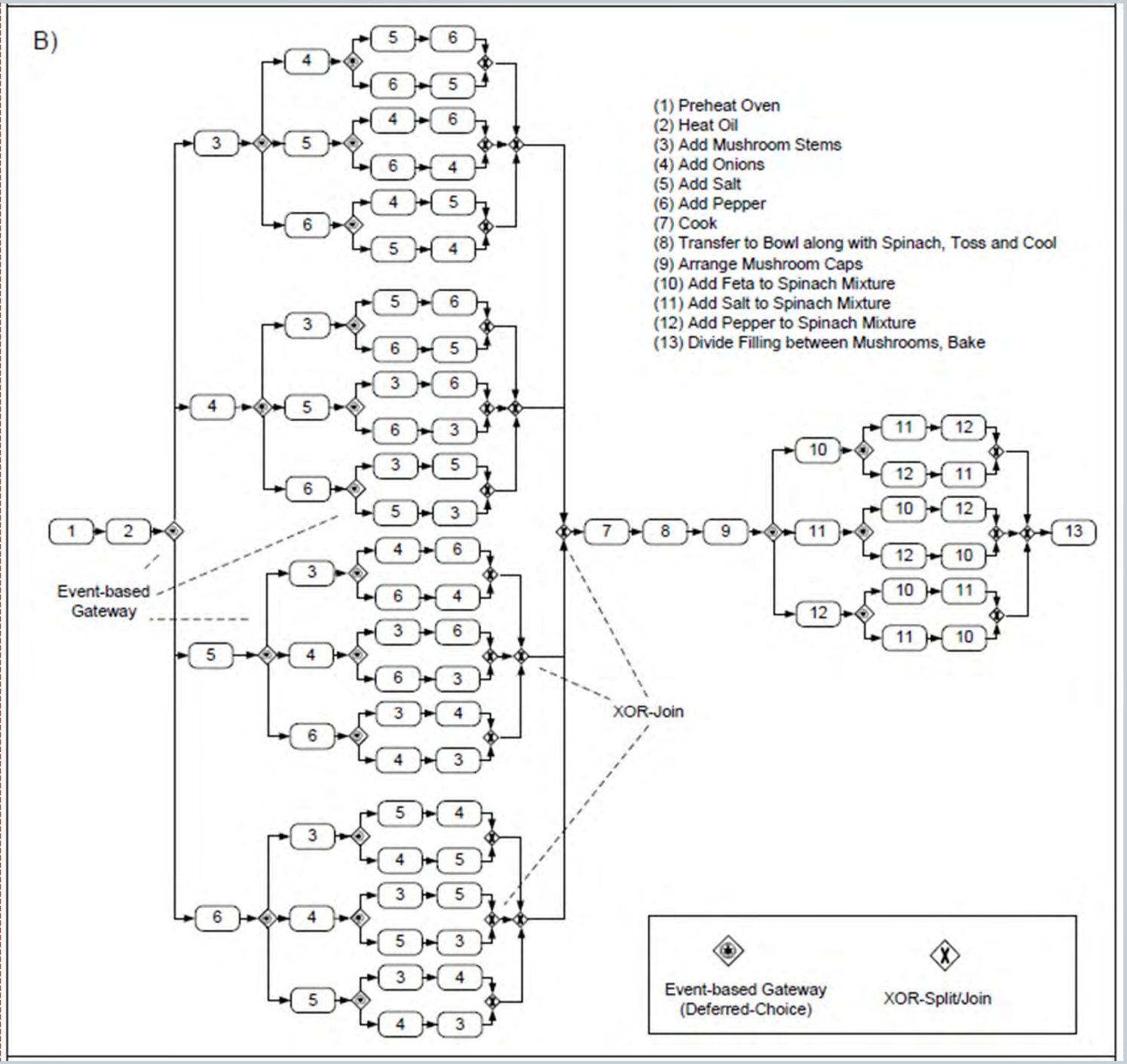
# Expressiveness and Flexibility-by-Design






© B. Weber and M. Reichert, 2012

# (Missing) Expressiveness and Flexibility by Design



# Business Processes and Workflows

## Configurable Process Models

30

BARBARA WEBER
UNIVERSITY OF INNSBRUCK

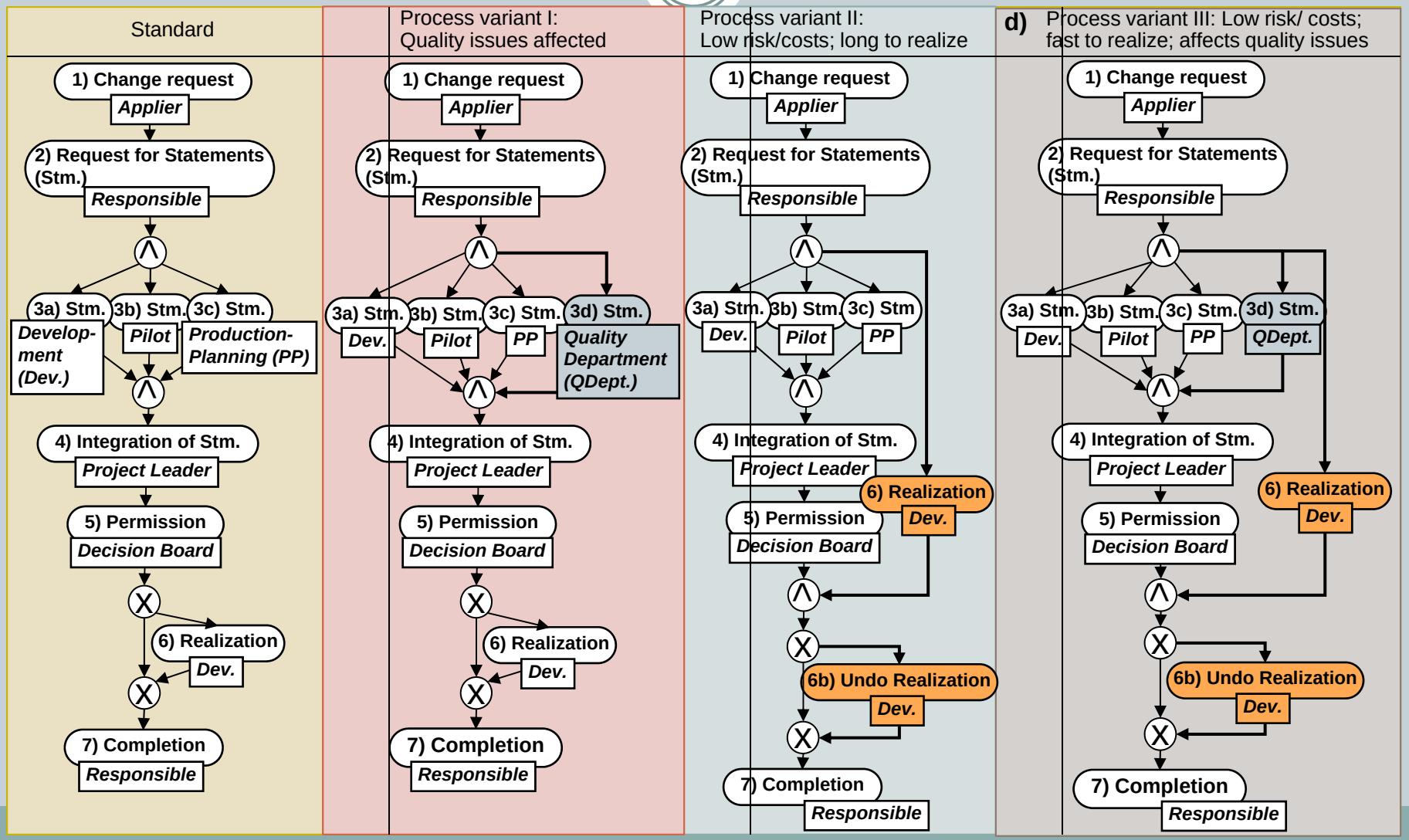
MANFRED REICHERT
ULM UNIVERSITY

ER 2012, FLORENCE

© B. Weber and M. Reichert, 2012

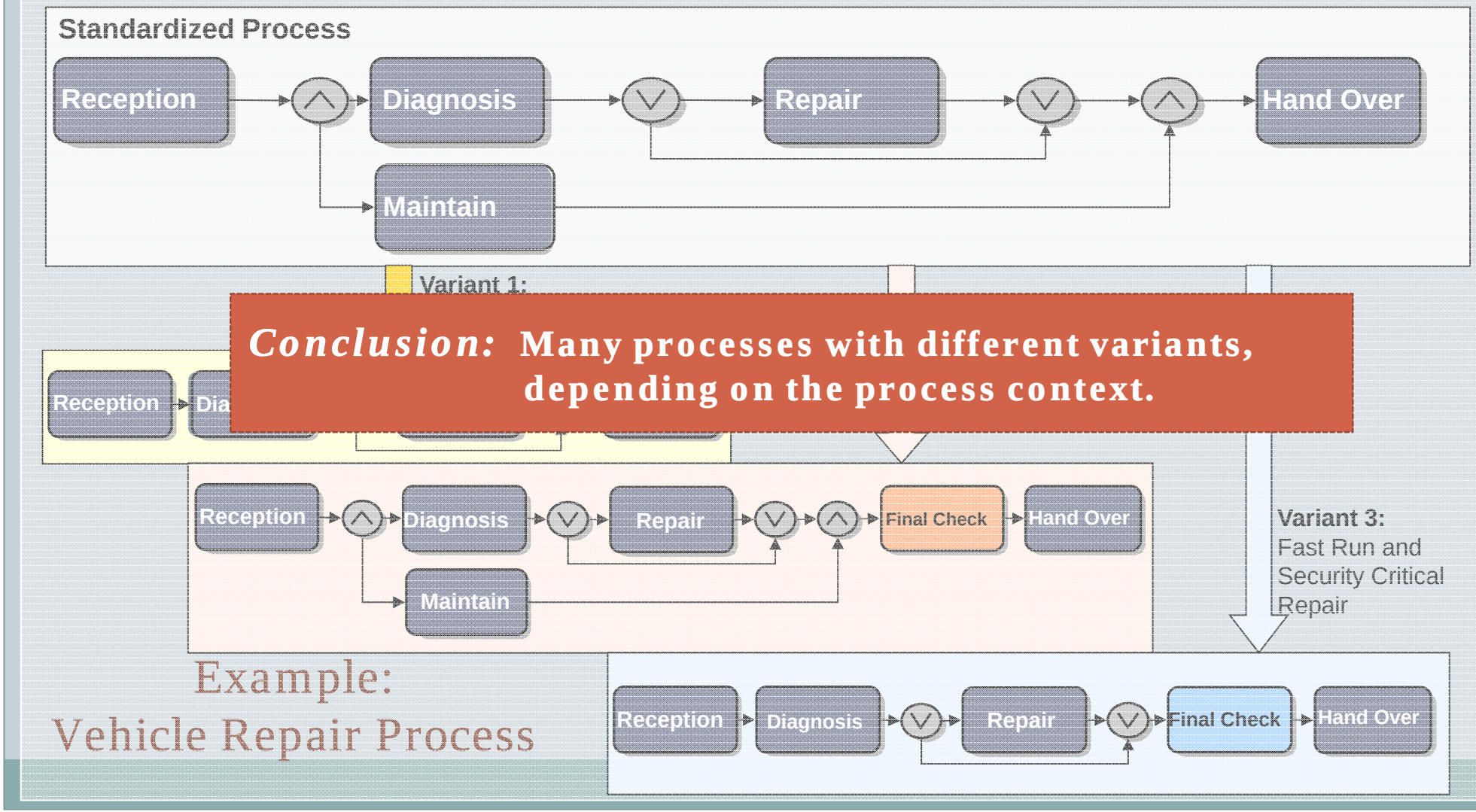
# Motivation – Change Management Process





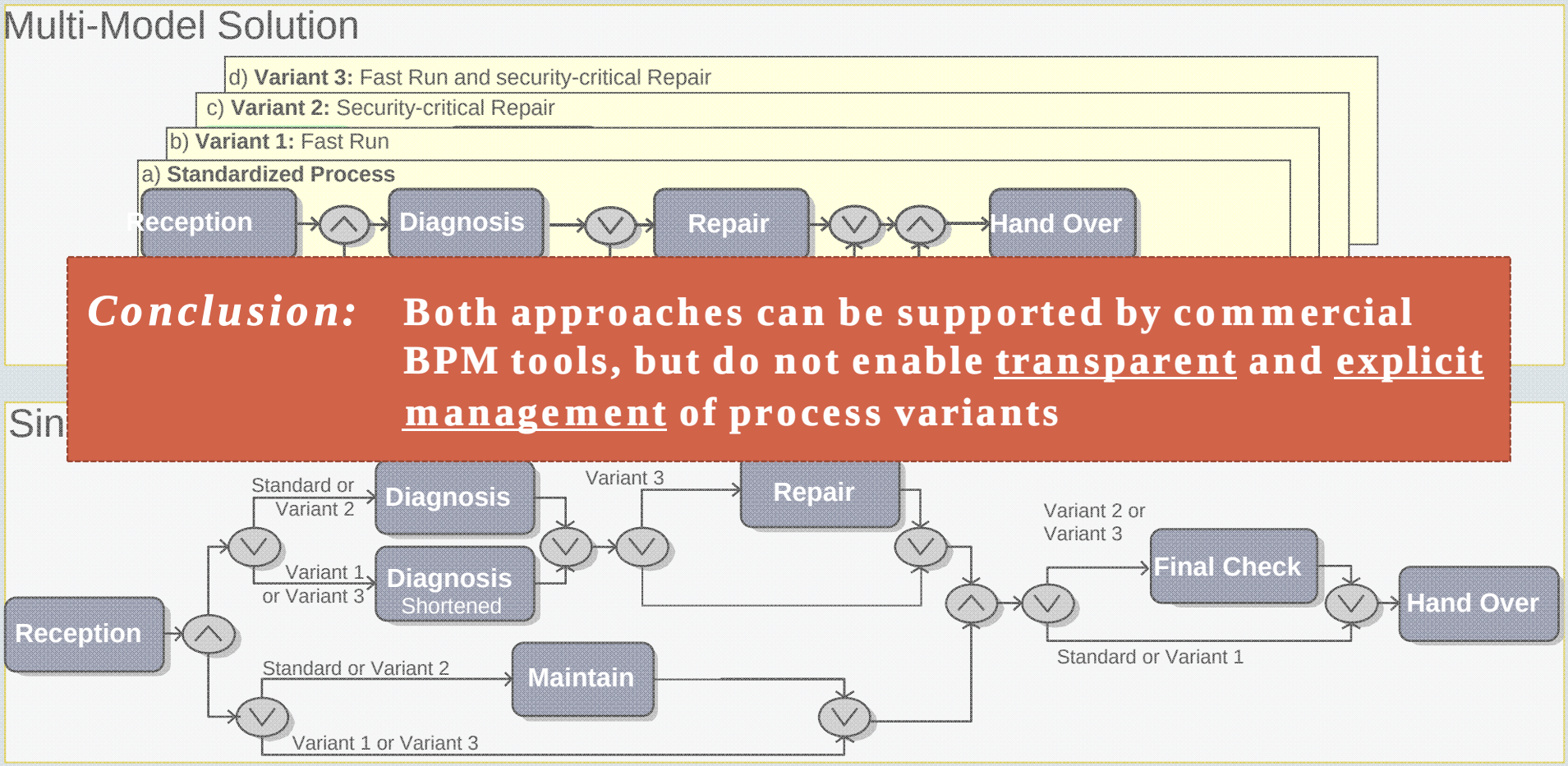
# Motivation – Vehicle Repair Process





# Configuring Process Variants in Existing BPM Tools



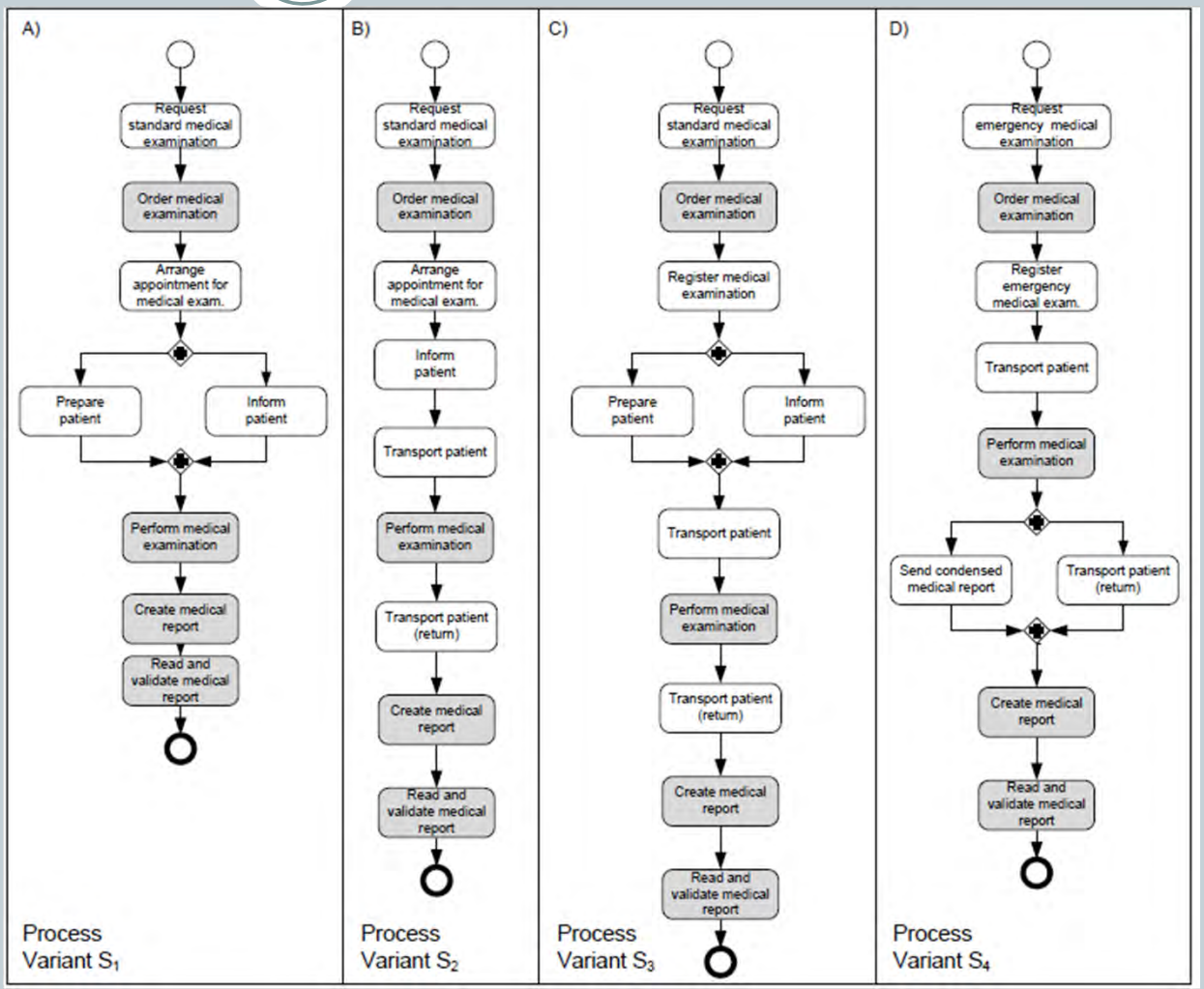


# Motivation – Handling Medical Examinations



## Variety of related variants

- Same business objective
- Commonalities
- Differences due to varying application context




(c) 2012 Barbara Weber, Manfred

# Two Main Approaches for Capturing Process Variability



- Behaviour-based Approaches
- Structural Approaches

© B. Weber and M. Reichert, 2012

# Behavior-based Approaches



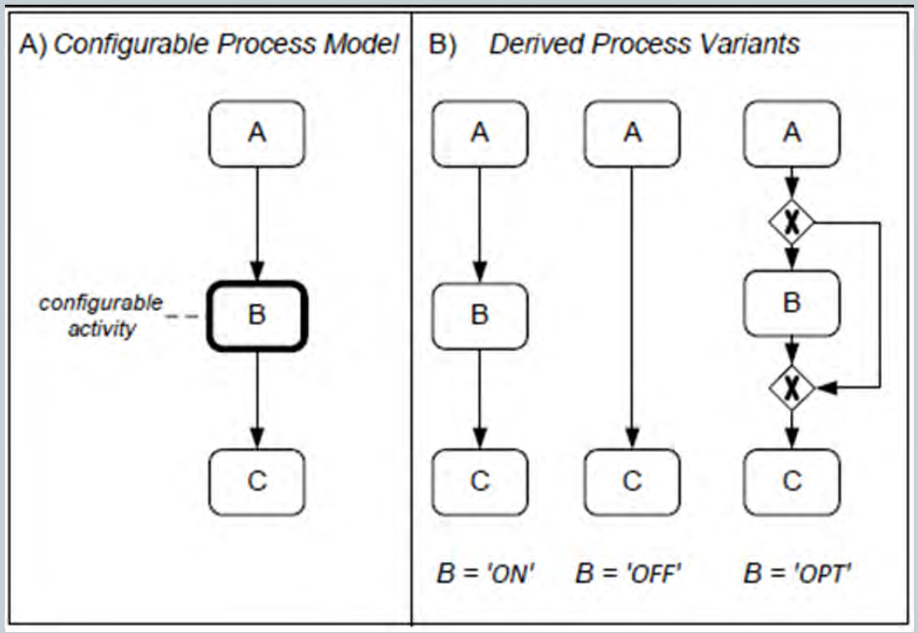
- Main idea: Merging all possible behavior in on reference model with **configurable nodes**
  - Extension of an existing process modeling language by adding configurable elements (e.g., activities, control connectors)
  - Examples: C-EPC, C-YAWL
- Configurable nodes represent **variation points** associated with configuration alternatives
- Possible combinations of configuration alternatives can be restriceted through constraints

© B. Weber and M. Reichert, 2012

# Configurable Activities



- Included (ON)
- Excluded (OFF)
- Conditional (OPT)



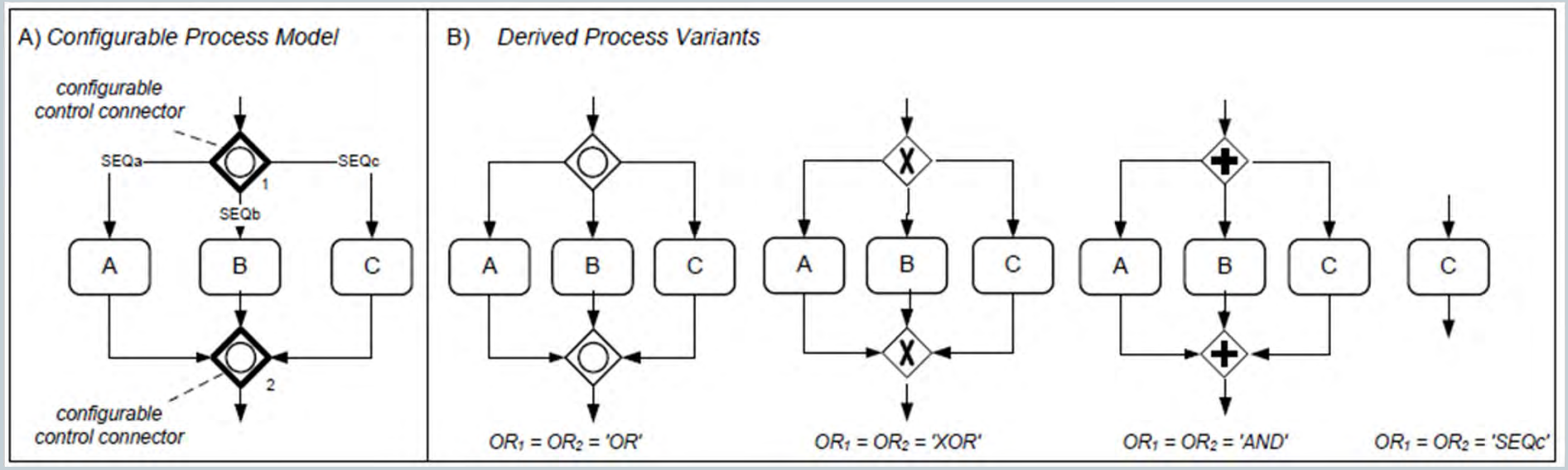

© B. Weber and M. Reichert, 2012

# Configurable Control Connectors



- **Configurable OR**
- Configurable XOR
- Configurable AND

Can be configured to a connector equally restrictive or less restrictive

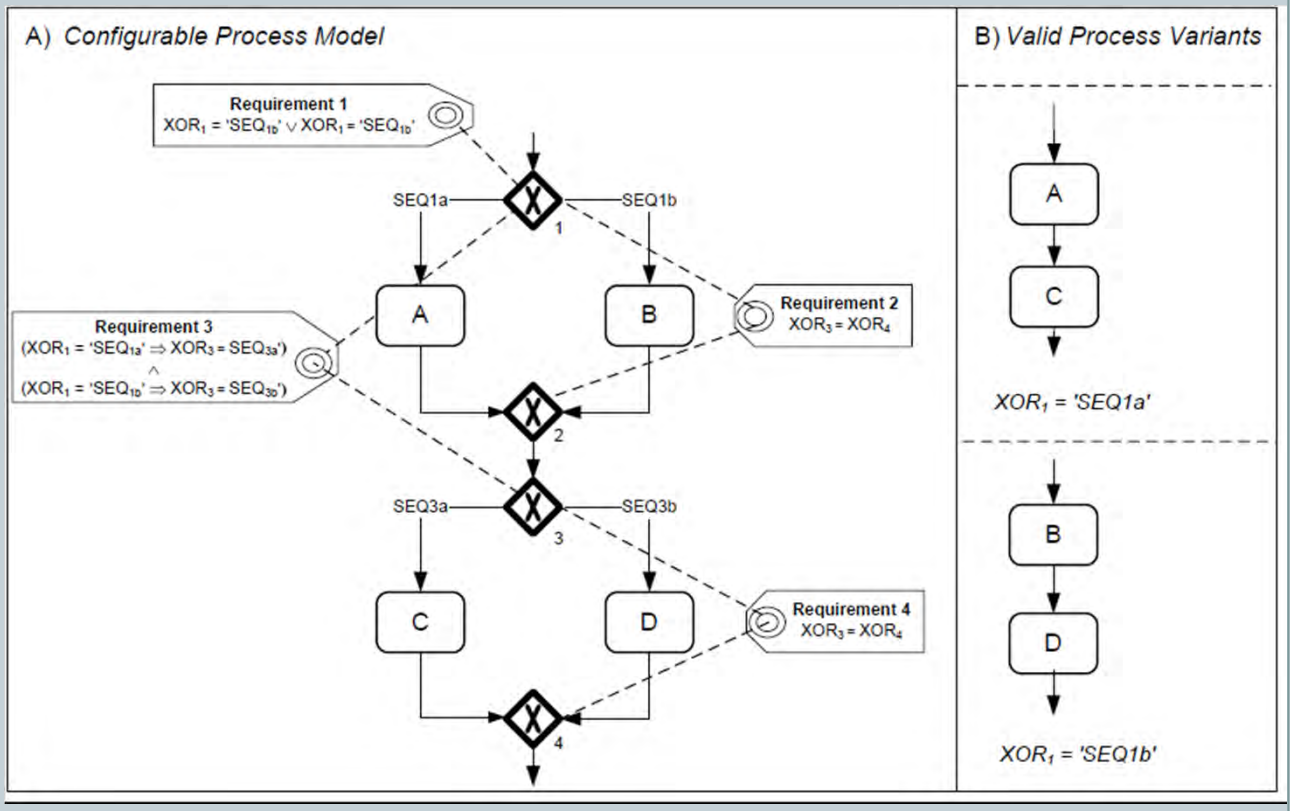



© B. Weber and M. Reichert, 2012

# Configuration Requirements and Guidelines



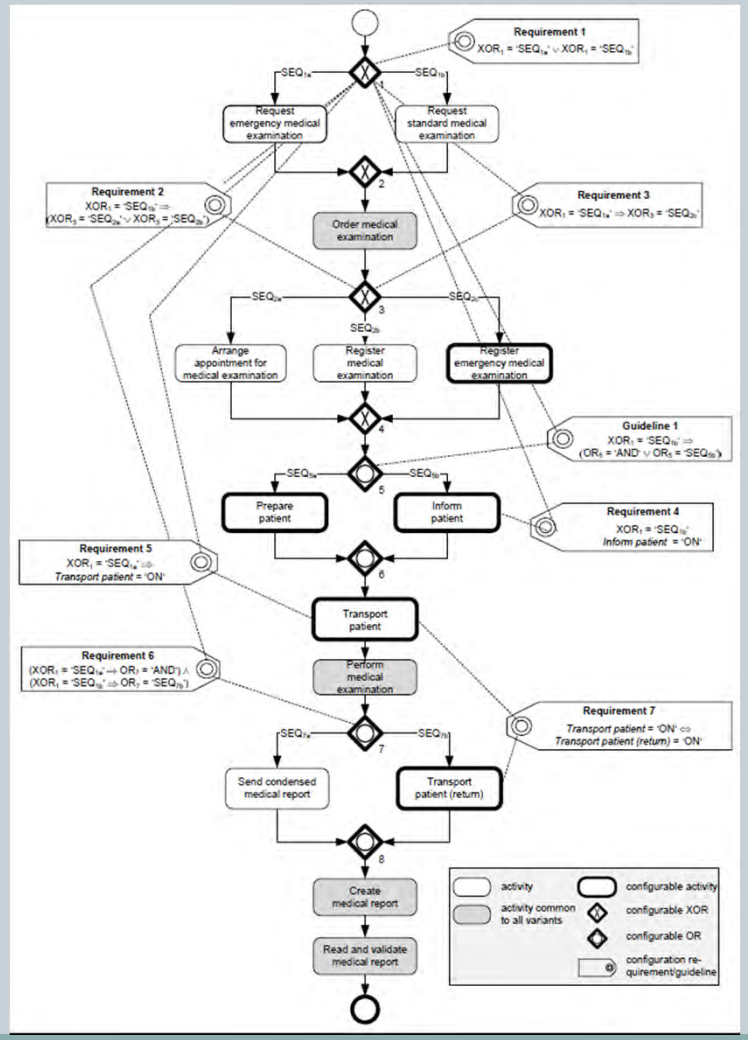
- Requirements
  - Define constraints over the configuration alternatives that may be chosen
- Guidelines
  - Do not prescribe mandatory constraints, but serve as recommendations

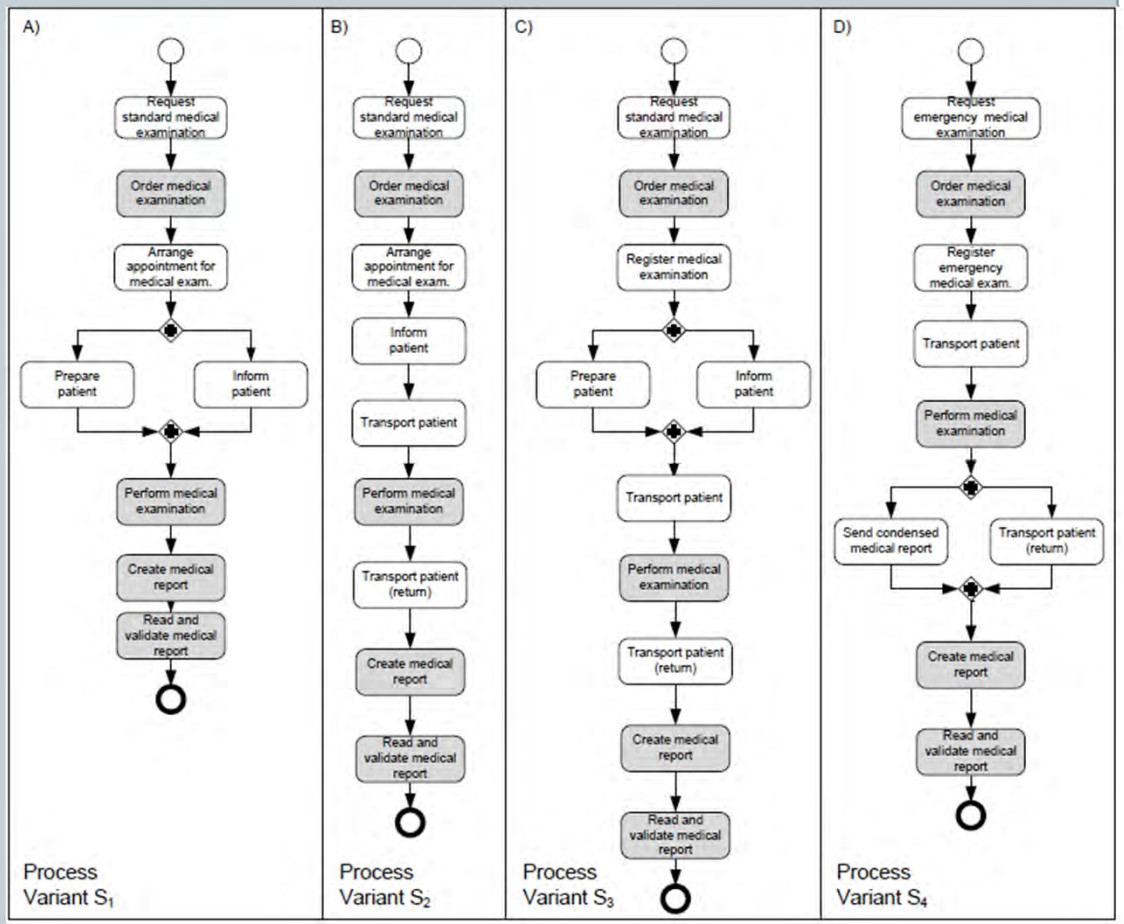



© B. Weber and M. Reichert, 2012

# Configurable Model:
## Handling Medical Examinations






© B. Weber and M. Reichert, 2012

# Two Main Approaches for Capturing Process Variability

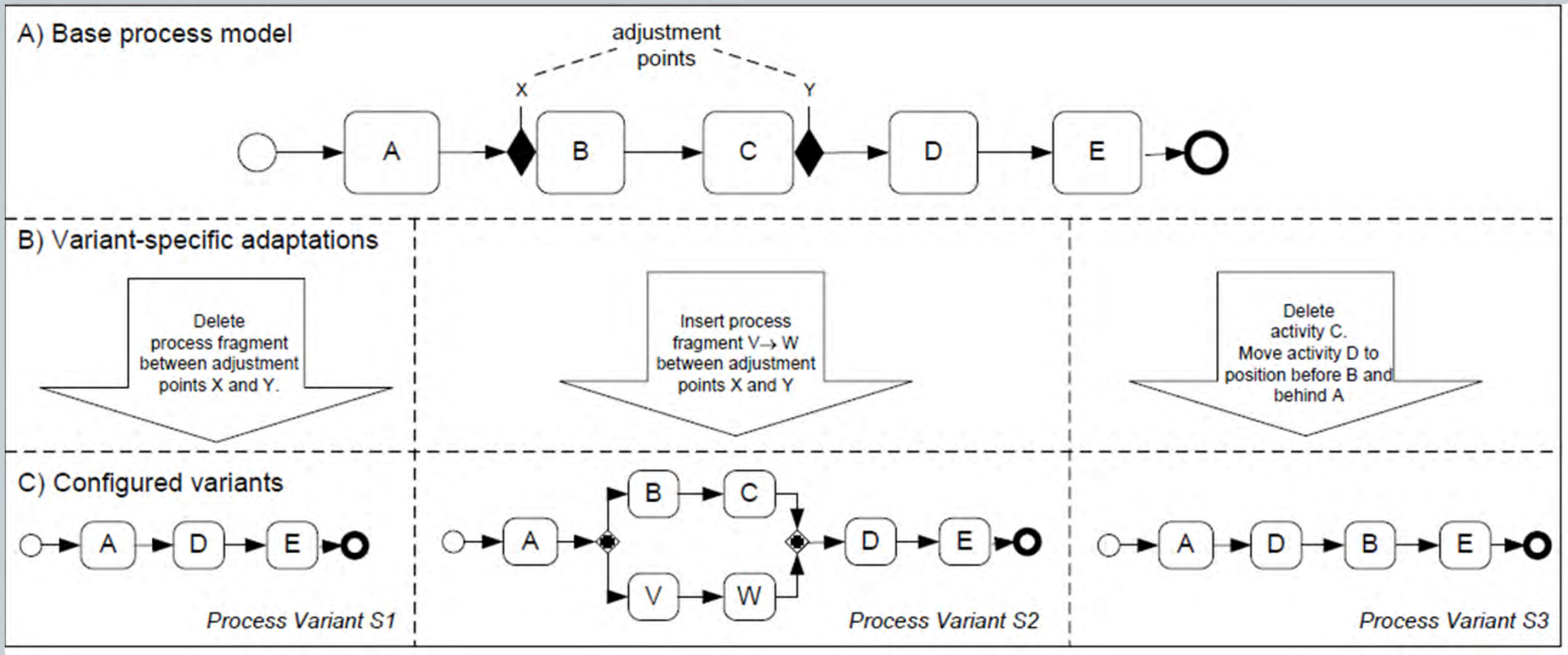


- Behaviour-based Approaches
- Structural Approaches

© B. Weber and M. Reichert, 2012

# Deriving Variants through Structural Changes of a Base Process Model






© B. Weber and M. Reichert, 2012

# Representing a Process Family



- Through a configurable base process model
  - Policy 1: Standard Process
  - Policy 2: Most frequently used process
  - Policy 3: Superset of all process variants
  - Policy 4: Intersection of all process variants
- and a related set of pre-specified changes
  - Adjustment points
  - Change options (i.e., a grouping of change operations)

© B. Weber and M. Reichert, 2012

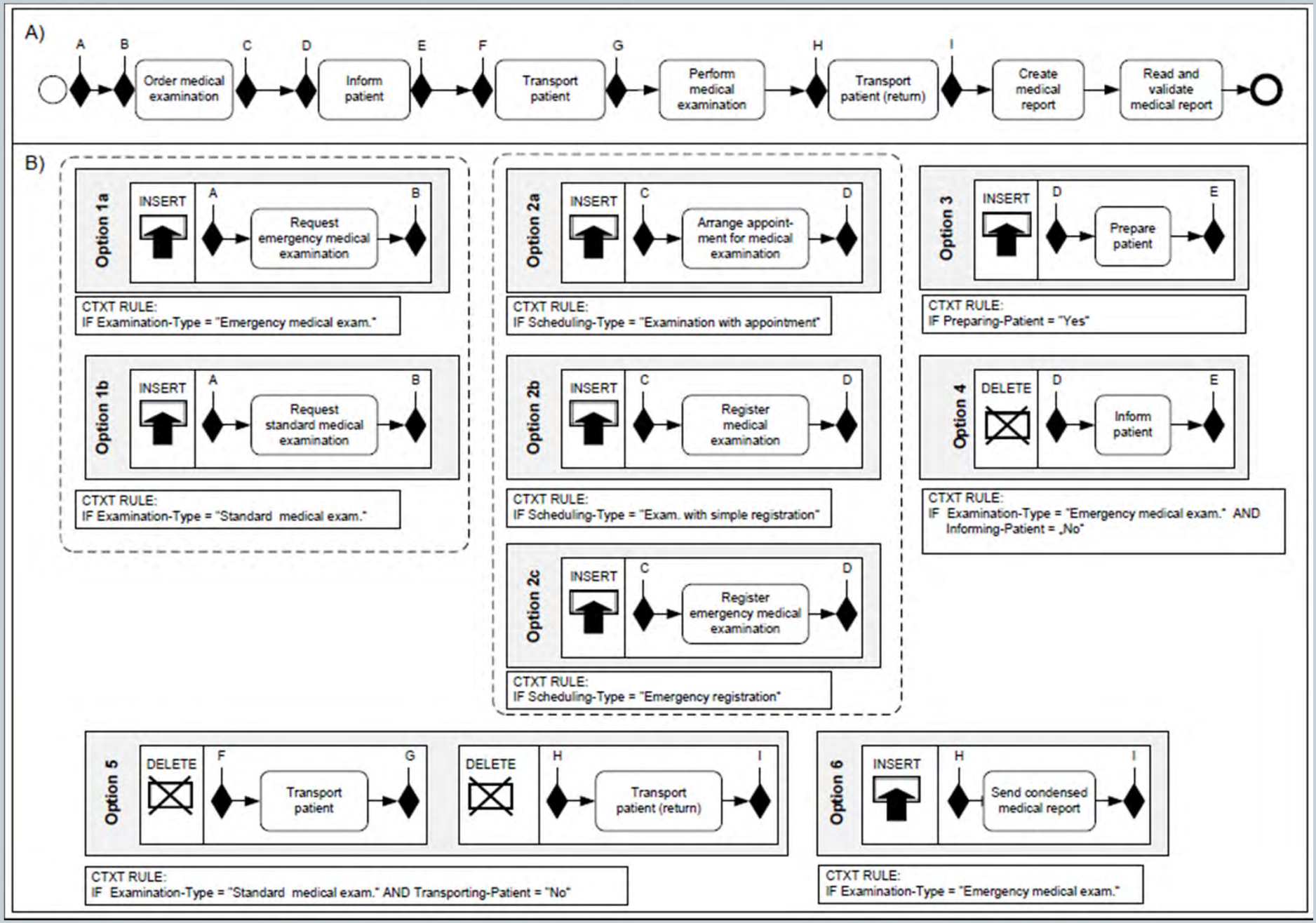
# Examples of Change Operations



| | |
|---|---|
| **1. INSERT-Operation** | |
| Symbol |  |
| Purpose | Adding a *process fragment* (e.g., a single activity or an activity sequence). |
| Parameters | • Process fragment to be added<br>• Target position of the process fragment to be added in the base process, specified in terms of adjustment points |
| **2. DELETE-Operation** | |
| Symbol |  |
| Purpose | Removing a *process fragment* |
| Parameters | • Process fragment to be deleted with entries and exits being marked by adjustment points<br>• Alternatively: deleting single activities by referring to their ID |
| **3. MOVE-Operation** | |
| Symbol |  |
| Purpose | Changing the execution order of activities |
| Parameters | • Process fragment to be moved with entries and exits being marked by adjustment points<br>• Target position of the process fragment to be moved, specified in terms of adjustment points |

© B. Weber and M. Reichert, 2012

# Example of Base Process + Options

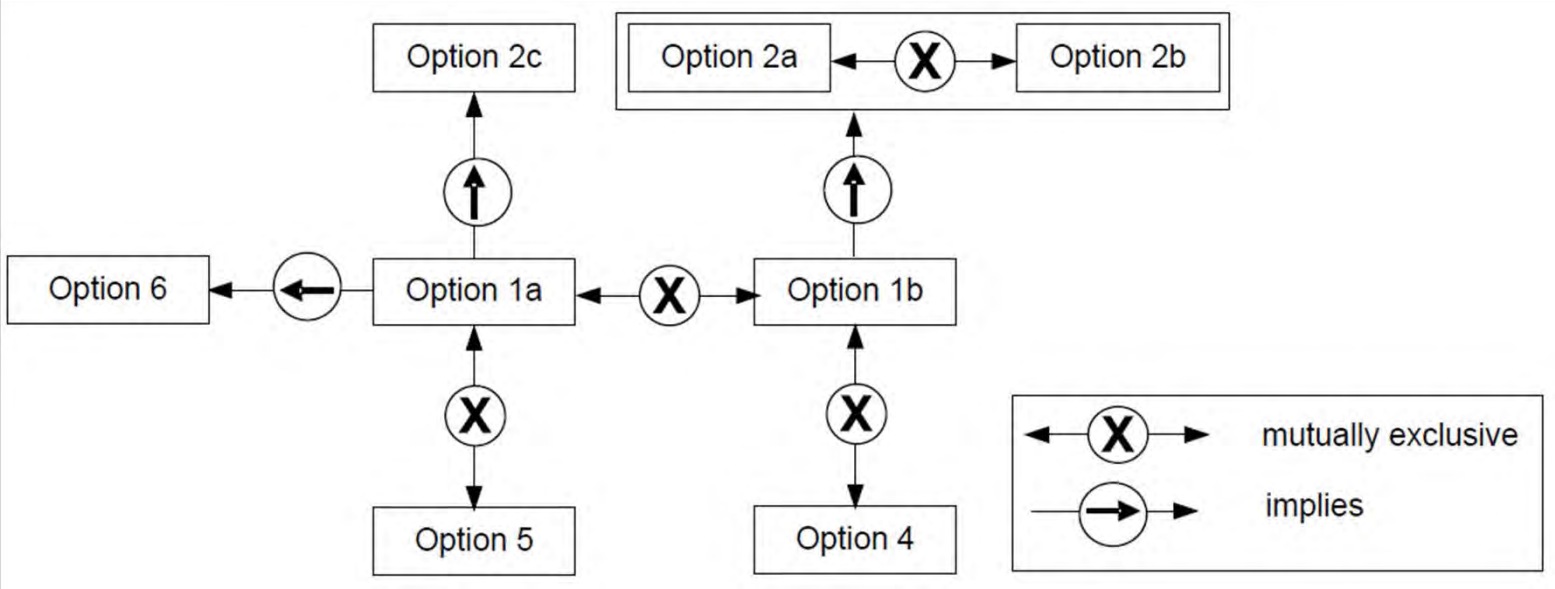



© B. Weber and M. Reichert, 2012

# Constraining Allowed Combinations of Change Options



- Implications
- Mutual exclusion
- Hierarchy




© B. Weber and M. Reichert, 2012

# Context Model



- Context-specific selection of change options
  - Context variables

| Context Variable | Value Range |
|---|---|
| Examination-Type | Standard medical exam., Emergency medical exam} |
| Scheduling-Type | Examination with appointment, Examination with simple registration, Emergency registration |
| Preparing-Patient | Yes, No |
| Informing-Patient | Yes, No |
| Transporting-Patient | Yes, No |

© B. Weber and M. Reichert, 2012

# Bringing all together ...



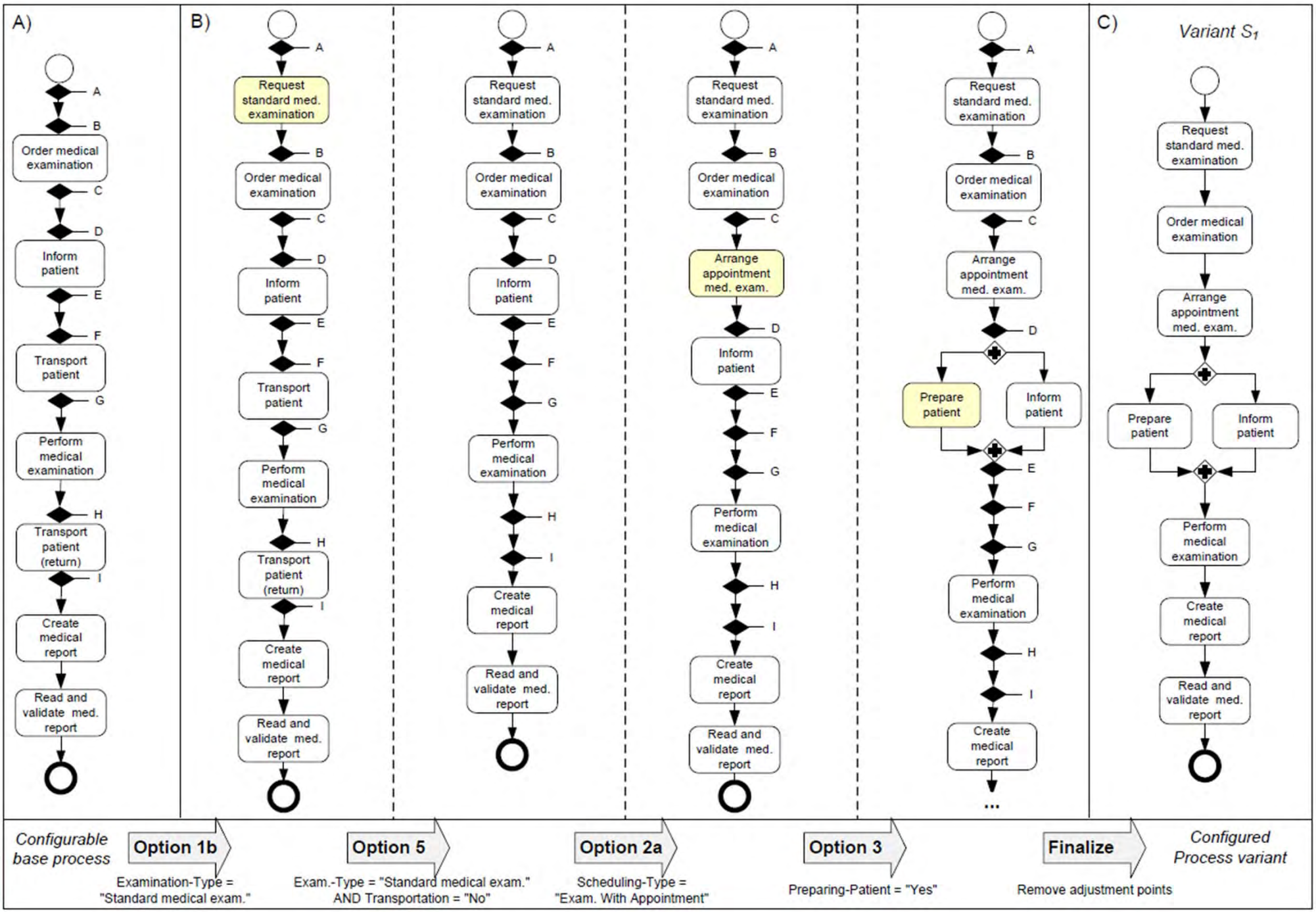
(1) Select relevant changes options ☞ All change options whose context rules evaluate to true are selected

(2) Ensure compliance of the selected options with option constraints ☞ Compliance with option constraints has to be checked

(3) Determine the order in which options shall be applied

(4) Configuring the base process by applying the selected options and their change operations to it

© B. Weber and M. Reichert, 2012


A)
B)
C)
Variant $S_1$
A
B
C
D
E
F
G
H
I
Order medical examination
Inform patient
Transport patient
Perform medical examination
Transport patient (return)
Create medical report
Read and validate med. report
Request standard med. examination
Arrange appointment med. exam.
Prepare patient
...
Configurable base process
Option 1b
Examination-Type = "Standard medical exam."
Option 5
Exam.-Type = "Standard medical exam." AND Transportation = "No"
Option 2a
Scheduling-Type = "Exam. With Appointment"
Option 3
Preparing-Patient = "Yes"
Finalize
Remove adjustment points
Configured Process variant

# Questionnaire-driven Process Configuration



(1) Questionnaire Model

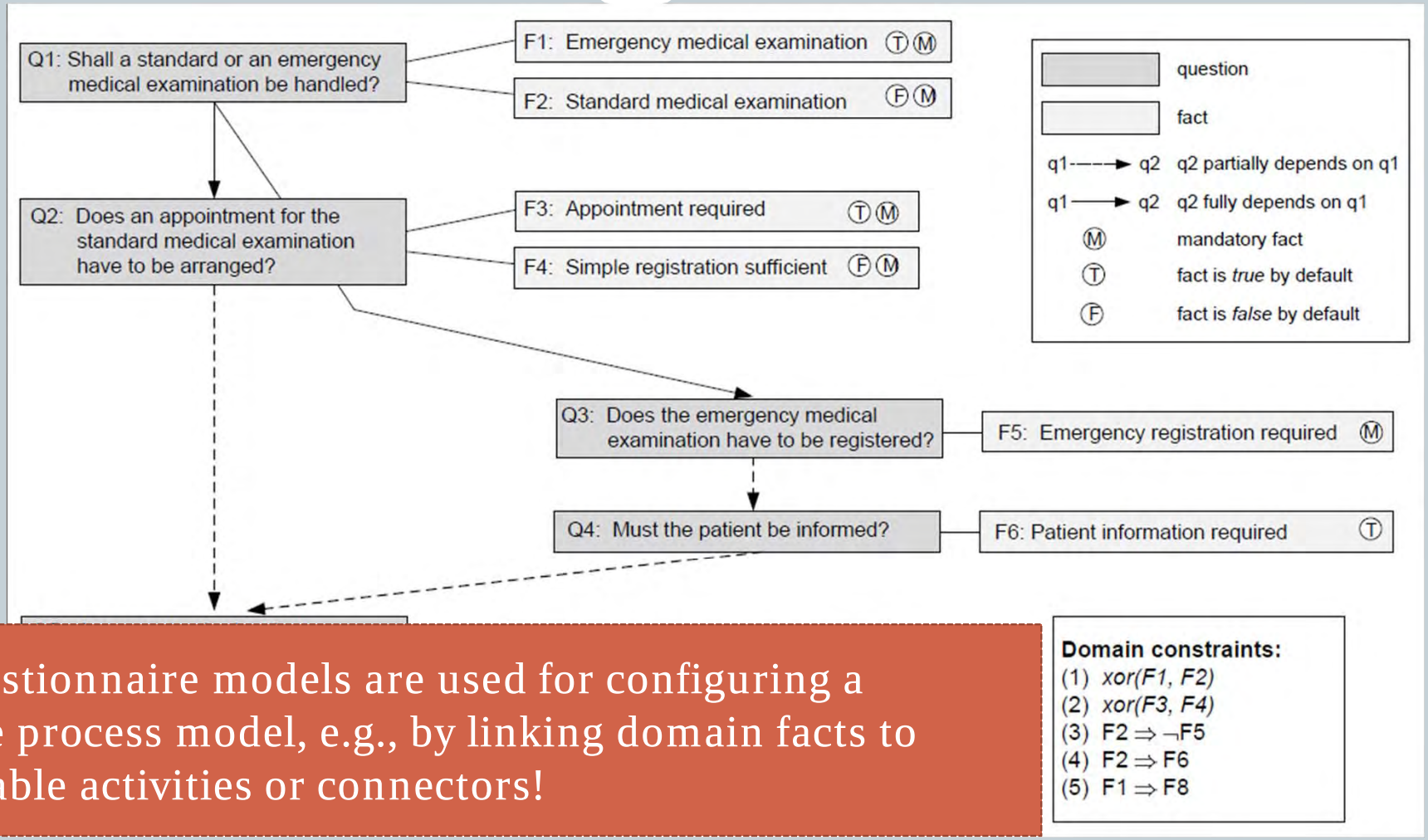
(2) Using Questionnaire Models for Configuring a Reference Process Model

(a) Linking Domain Facts and Configurable Activities

(b) Linking Domain Facts and Configurable Connectors

© B. Weber and M. Reichert, 2012

# Questionnaire-driven Process Configuration




Such questionnaire models are used for configuring a reference process model, e.g., by linking domain facts to configurable activities or connectors!

© B. Weber and M. Reichert, 2012

# Business Processes and Workflows

## Exception and Compensation Handling



BARBARA WEBER
UNIVERSITY OF INNSBRUCK

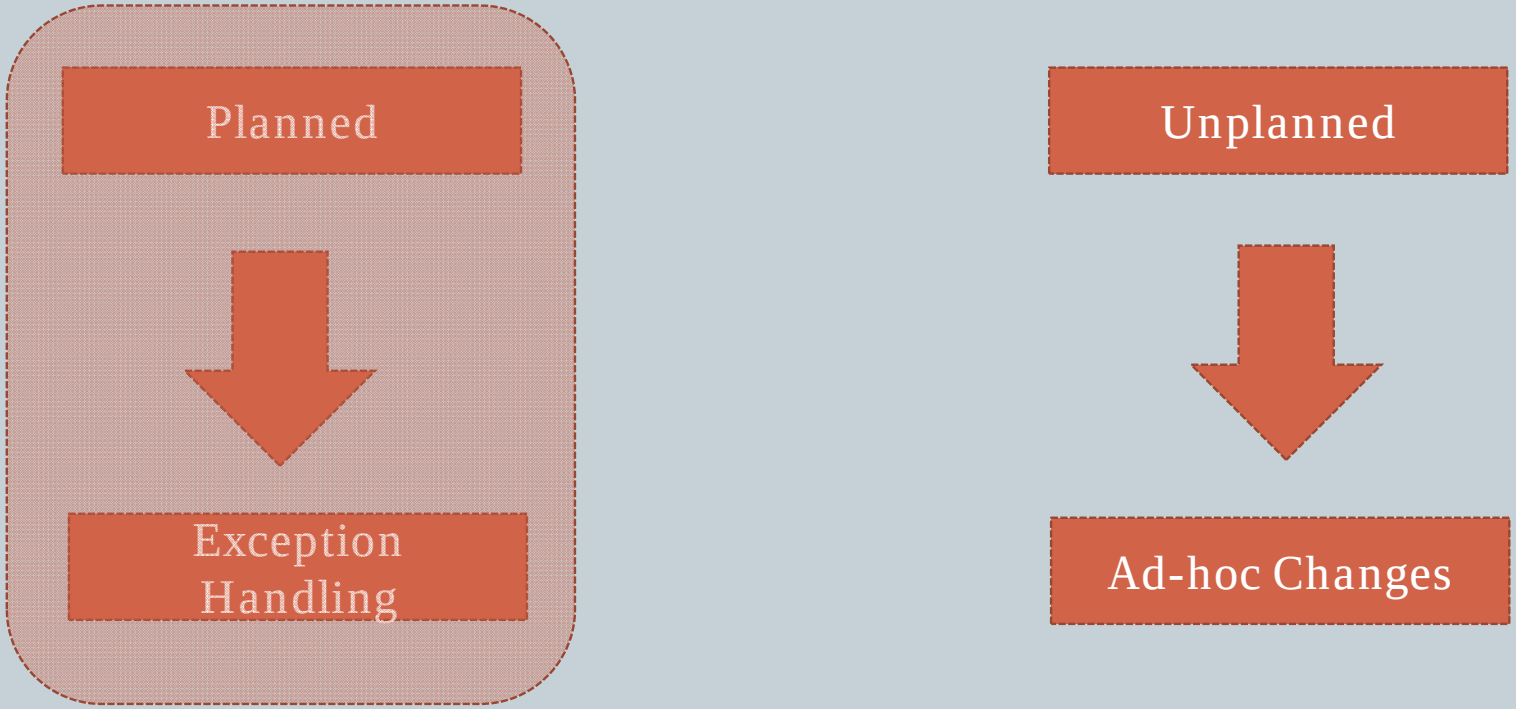
MANFRED REICHERT
ULM UNIVERSITY

ER 2012, FLORENCE

© B. Weber and M. Reichert, 2012

# Process Adaptations






© B. Weber and M. Reichert, 2012

# Exception Handling in PAIS

We will skip respective techniques for handling planned execptions in this tutorial and refer to our text book instead!

Sources

Activity

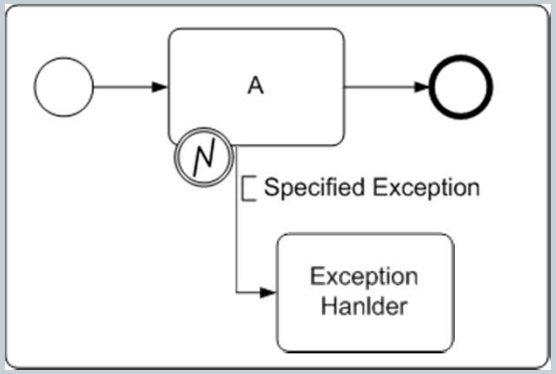
Technical | Semantical

Deadline Expiry

Resource Unavailability

Inconsistence real-world / PAIS

Constraint Violations




a suitable handler is chosen

dler

ves

ordered

Add Behavior

Deferred Fixing | Immediate Fixing

Retry | Exception-driven Rework

Cancelling Behavior

Reject | Compensate

Resource Patterns

Delegate | Escalate | ...

© B. Weber and M. Reichert, 2012

# Business Processes and Workflows

## Handling Unforeseen Exceptions



BARBARA WEBER
UNIVERSITY OF INNSBRUCK

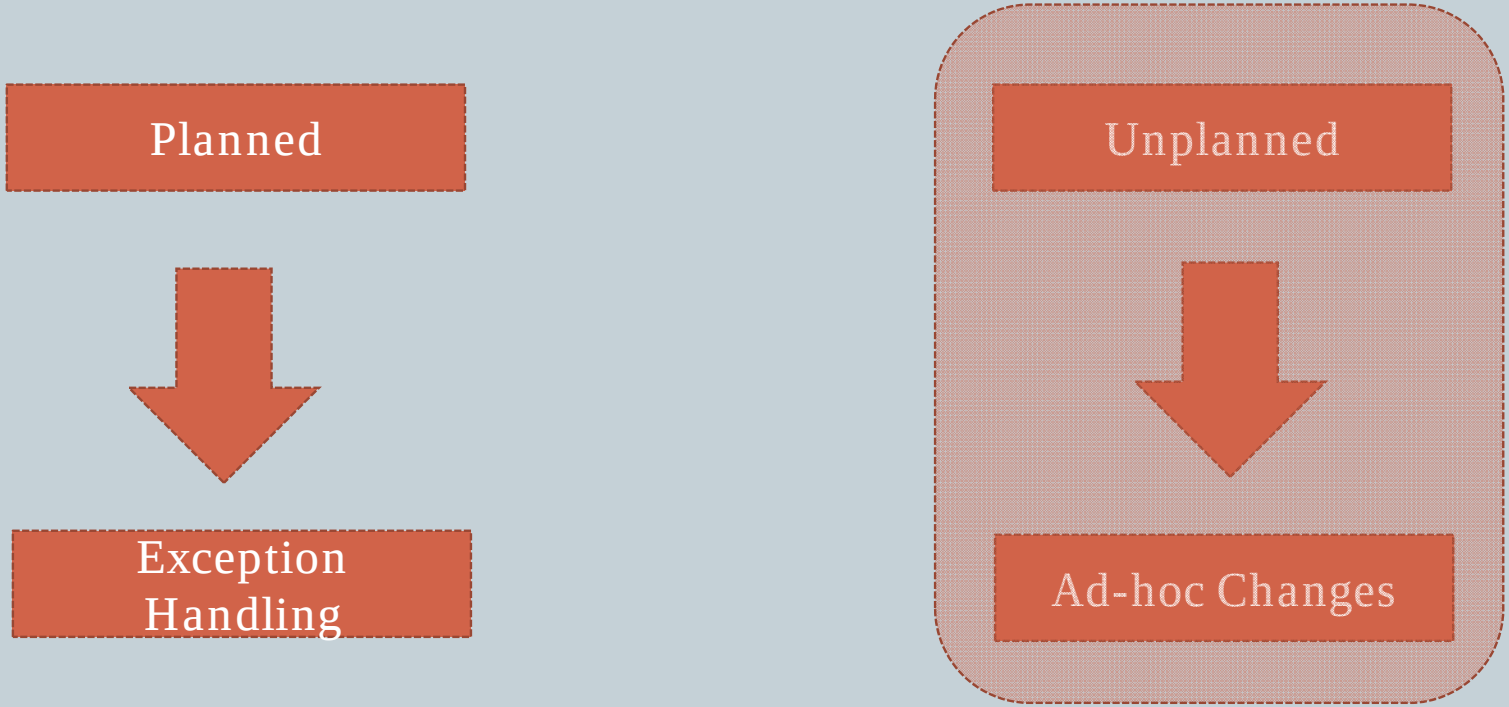
MANFRED REICHERT
ULM UNIVERSITY

ER 2012, FLORENCE

© B. Weber and M. Reichert, 2012

# Process Adaptations

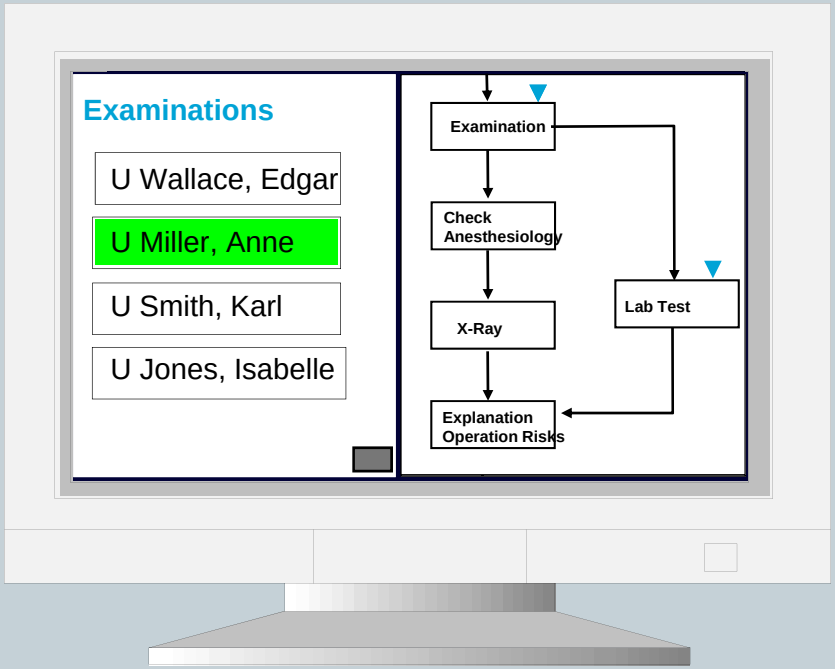





© B. Weber and M. Reichert, 2012

# User View on an Ad-hoc Process Change



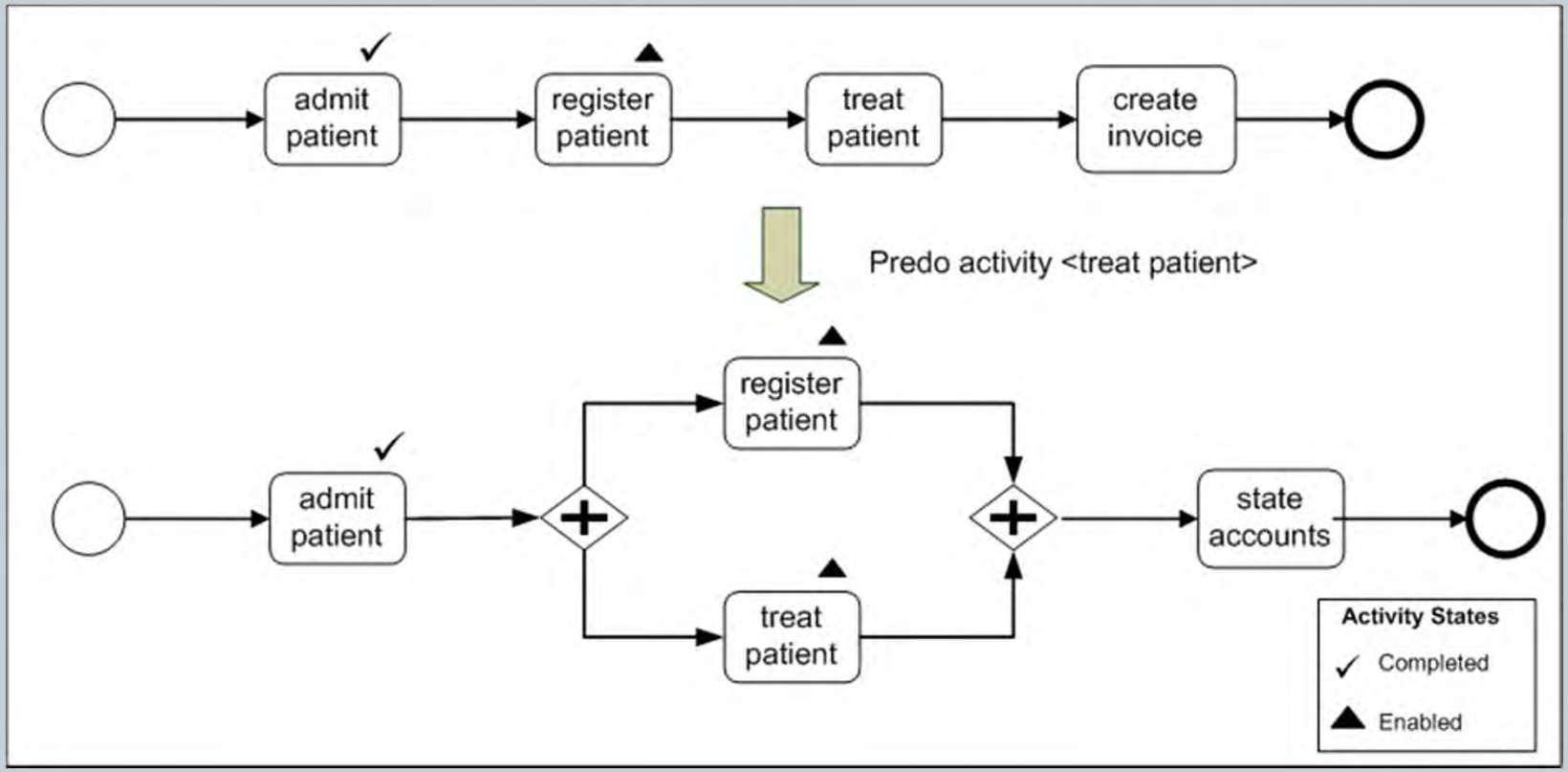




© B. Weber and M. Reichert, 2012

# Behavioral Changes Require Structural Process Model Adaptations



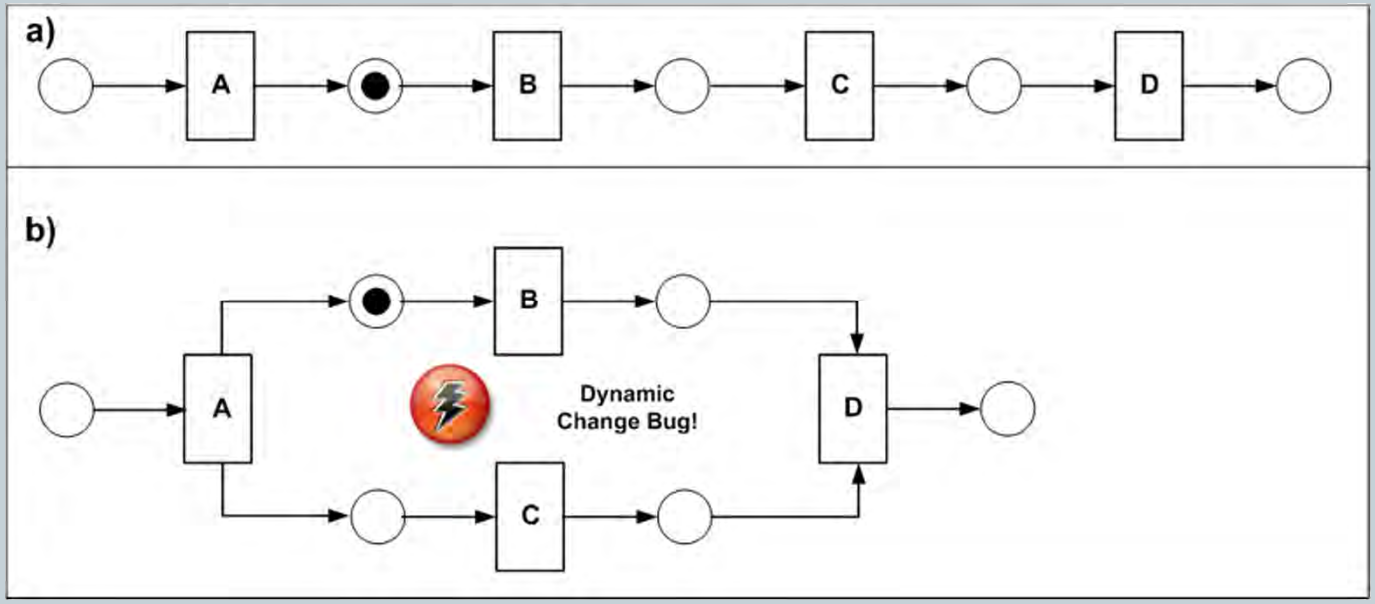



© B. Weber and M. Reichert, 2012

# Behavioral Changes Require Adaptations of the Process Instance State



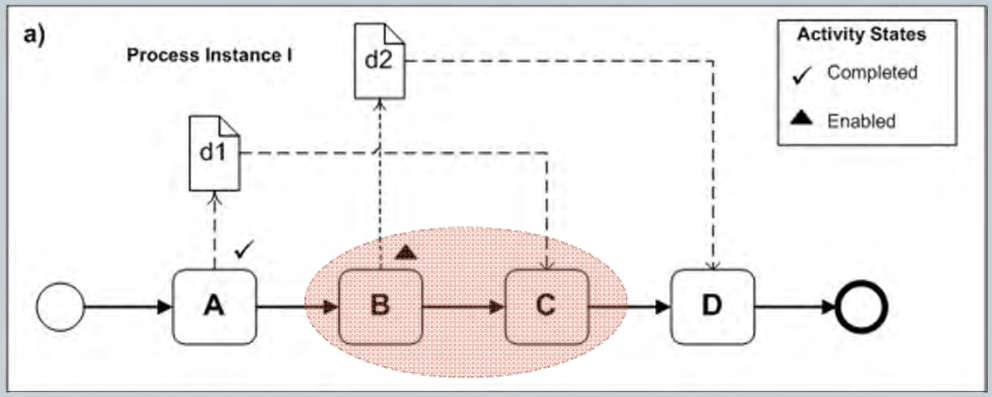
## Dynamic Change Bug

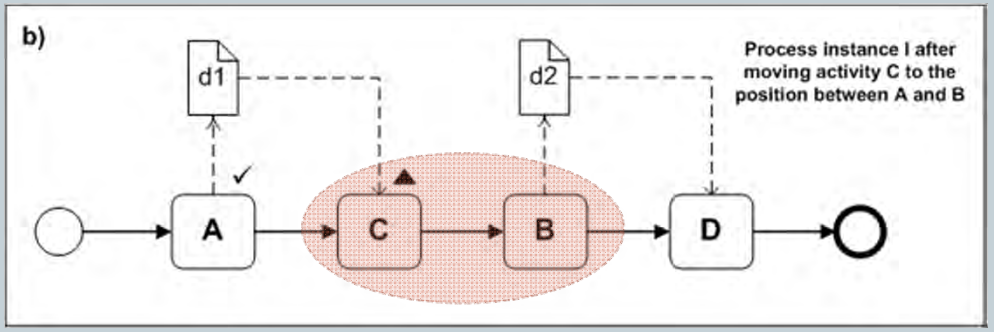



© B. Weber and M. Reichert, 2012

# Behavioral Changes Require Adaptations of the Process Instance State



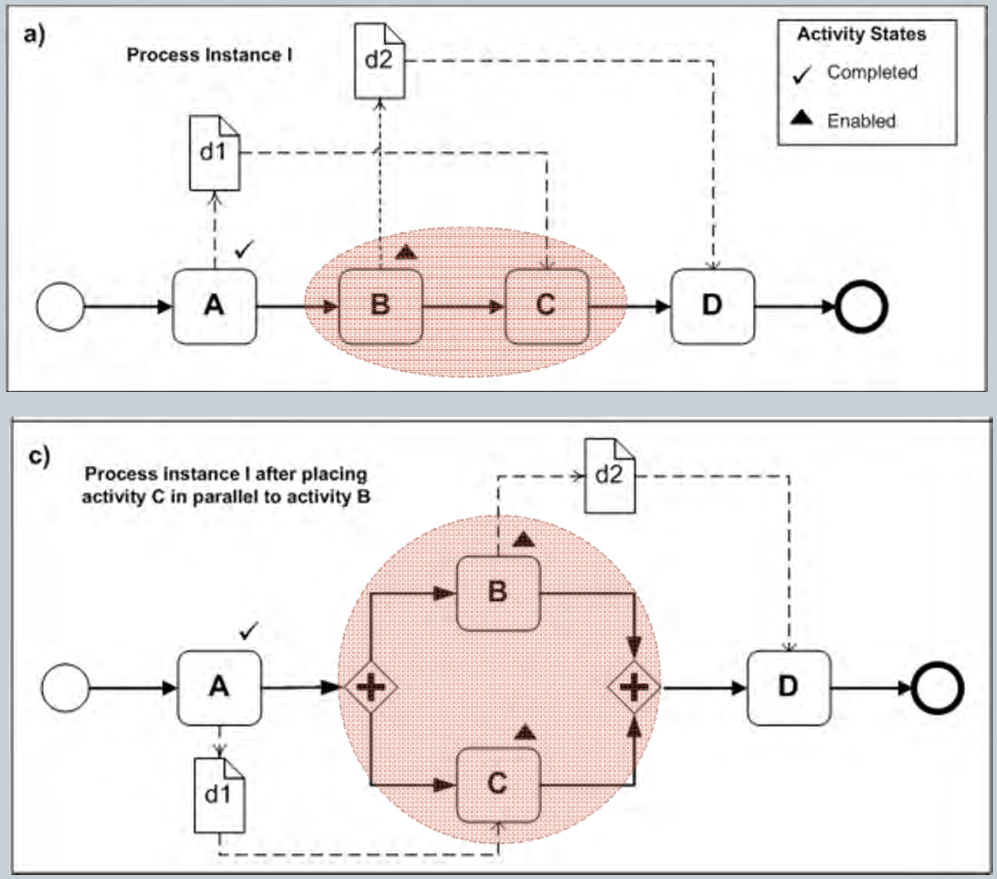



© B. Weber and M. Reichert, 2012

# Behavioral Changes Require Adaptations of the Process Instance State

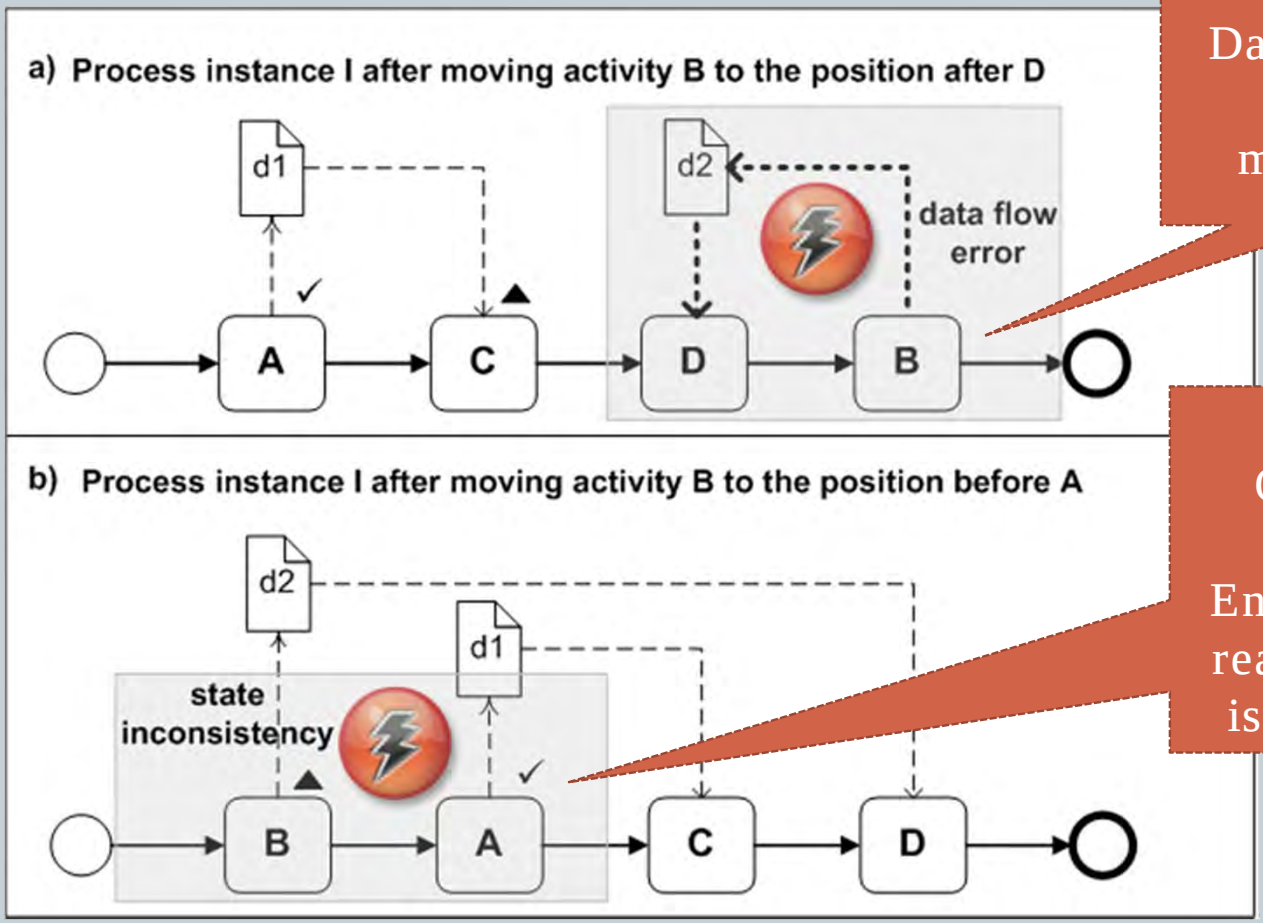





© B. Weber and M. Reichert, 2012

# Behavioral Changes Must not Violate Process Model Soundness and Proper Instance Execution






Data flow error caused by missing data

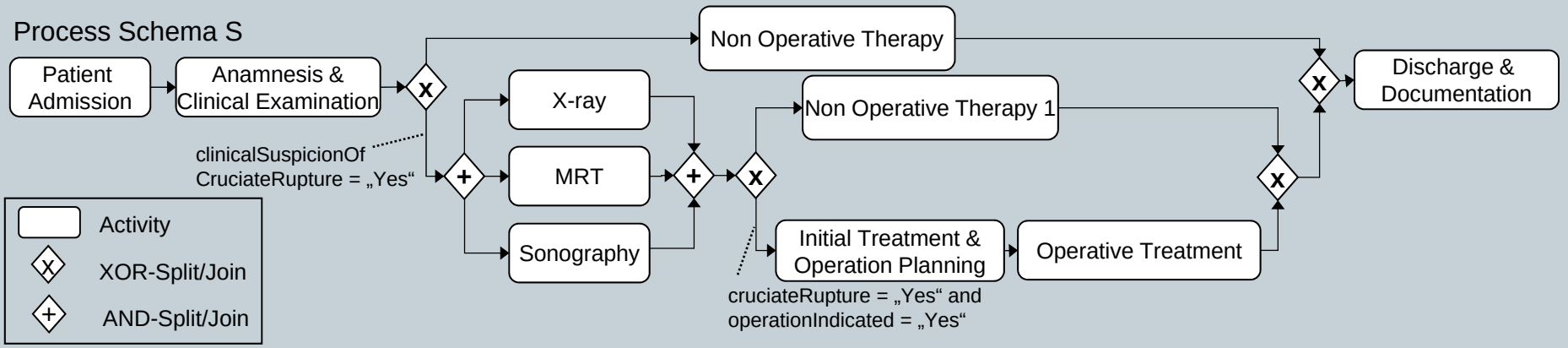
No Proper Completion ensured. End node can be reached while B is still enabled

© B. Weber and M. Reichert, 2012

# Ad-hoc Changes of a Process Instance Must Not Affect any Other Process Instances

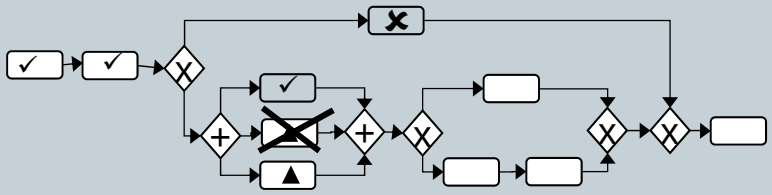


## Process Type Level



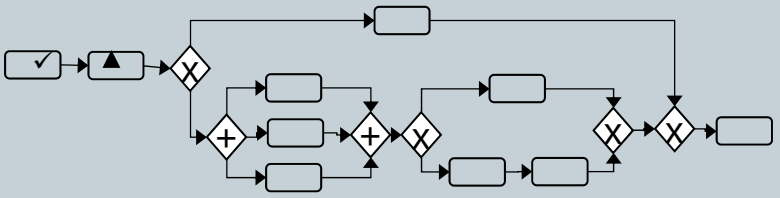

## Process Instance Level

Process Instance I1



**Execution Trace:**
$\sigma_1$ = < „Patient Admission“, „Anamnesis & Clinical Examination“, „X-ray“>

Process Instance I2



**Execution Trace:**
$\sigma_2$ = < „Patient Admission“>

© B. Weber and M. Reichert, 2012

# Structurally Adapting Pre-Specified Process Models



## Change Primitives

- Add node
- Remove node
- Add edge
- Remove edge
- Move edge

## High-Level Change Operations

- Combines a set of change primitives
- Referred to as Adaptation Patterns in the following

© B. Weber and M. Reichert, 2012

# Adaptation Patterns



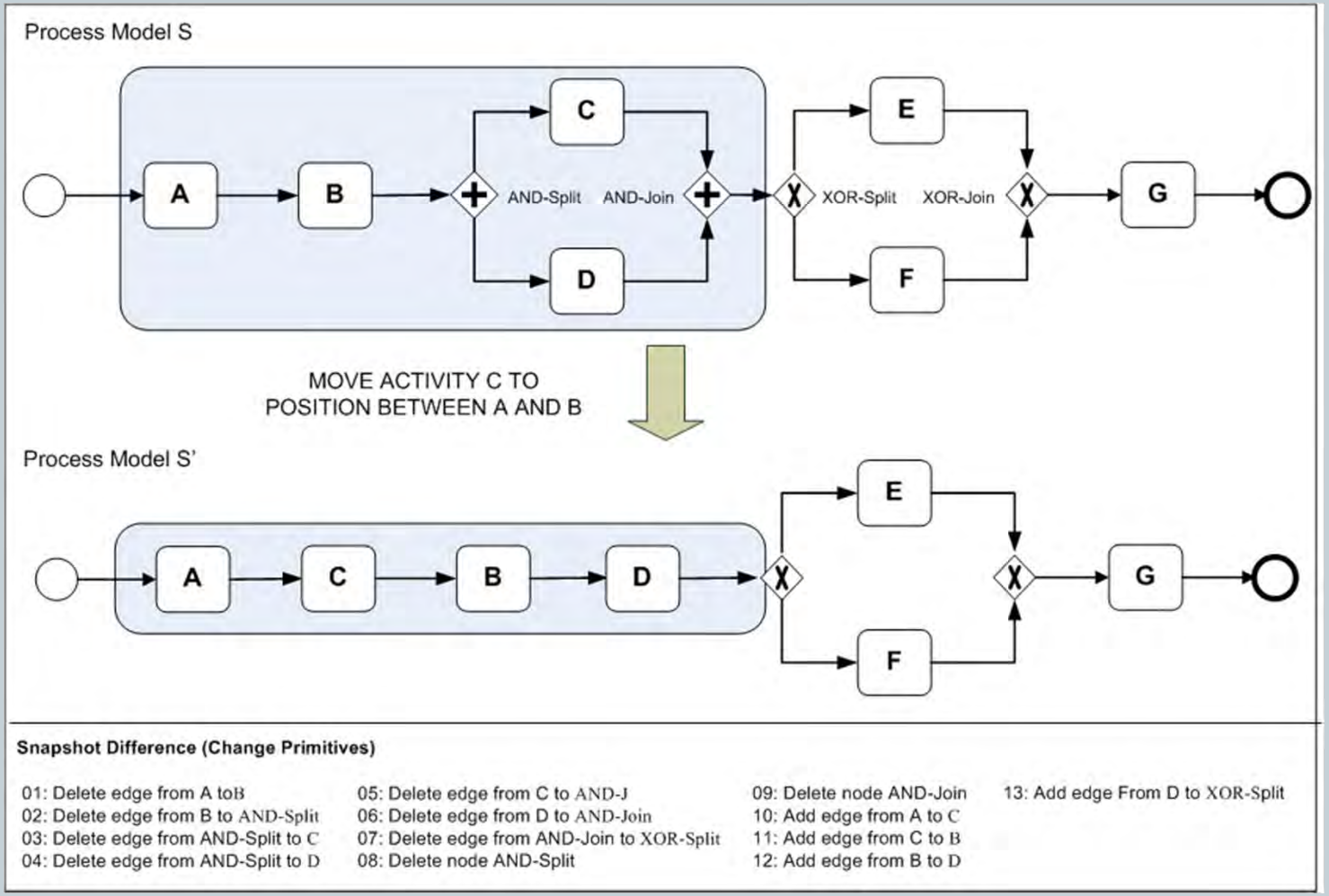
| | | |
|---|---|---|
| Adding / Deleting Process Fragments | AP1: | Insert Process Fragment |
| | AP2: | Delete Prrocess Fragment |
| Moving / Replacing Process Fragments | AP3: | Move Process Fragment |
| | AP4: | Replace Process Fragment |
| | AP5: | Swap Process Fragment |
| | AP14: | Copy Process Fragment |
| Adding / Removing Process Levels | AP6: | Extract Sub Process |
| | AP7: | Inline Sub Process |
| Adapting Control Dependencies | AP8: | Embed Process Fragment in Loop |
| | AP9: | Parallelize Process Fragments |
| | AP10: | Embed Process Fragment in Conditional Branch |
| | AP11: | Add Control Dependency |
| | AP12: | Remove Control Dependency |
| Change Transition Conditions | AP13: | Update Condition |

[WRR08]

© B. Weber and M. Reichert, 2012

# Adaptation Patterns versus Change Primitives






**Snapshot Difference (Change Primitives)**

| | | | |
|---|---|---|---|
| 01: Delete edge from A toB | 05: Delete edge from C to AND-J | 09: Delete node AND-Join | 13: Add edge From D to XOR-Split |
| 02: Delete edge from B to AND-Split | 06: Delete edge from D to AND-Join | 10: Add edge from A to C | |
| 03: Delete edge from AND-Split to C | 07: Delete edge from AND-Join to XOR-Split | 11: Add edge from C to B | |
| 04: Delete edge from AND-Split to D | 08: Delete node AND-Split | 12: Add edge from B to D | |

© B. Weber and M. Reichert, 2012

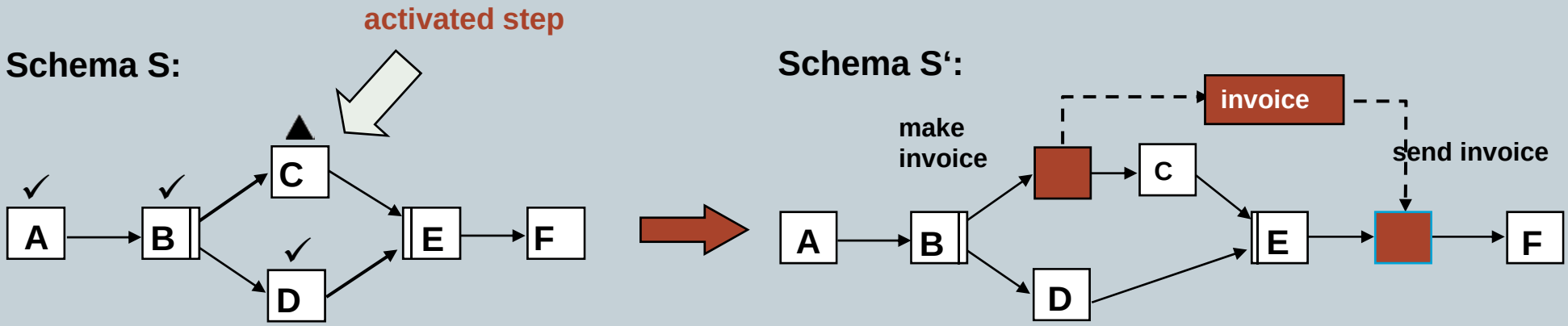
**BW2** Bild tauschen

Barbara Weber; 02.04.2011

# Correctness of Process Instance Changes



## Ensuring Dynamic Correctness




**May the depicted schema change be propagated to the process instance?**

**Need for general correctness criterion**

**⇨State Compliance**

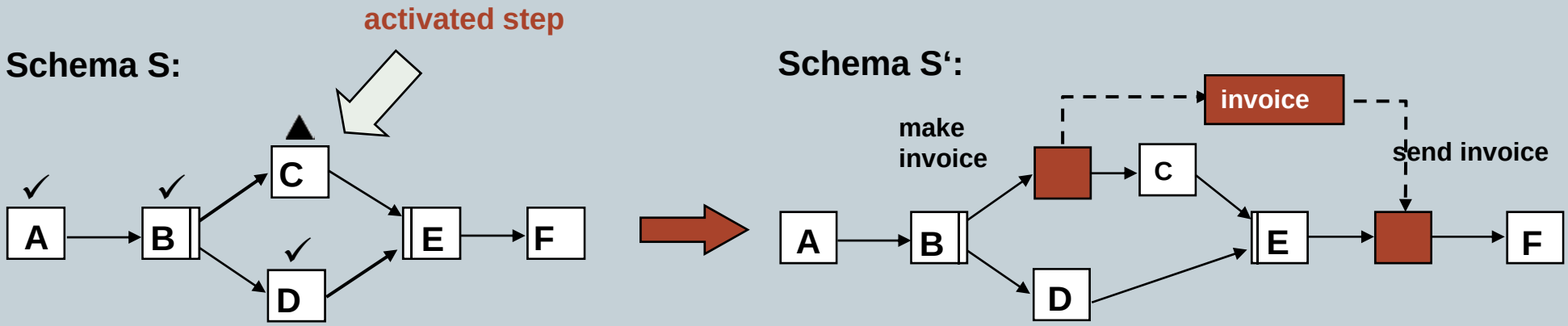
[ReDa98, RRW08a, RRD04a, RRD04b]

© B. Weber and M. Reichert, 2012

# Correctness of Process Instance Changes



## Ensuring Dynamic Correctness




<A> , <B> , <D> ⇨ Trace reproducible on new schema?

More complicated: loop backs

---

Further challenges: - How to efficiently check for compliance?

- How to efficiently migrate process instances?

[RRD04a, RRD04b]

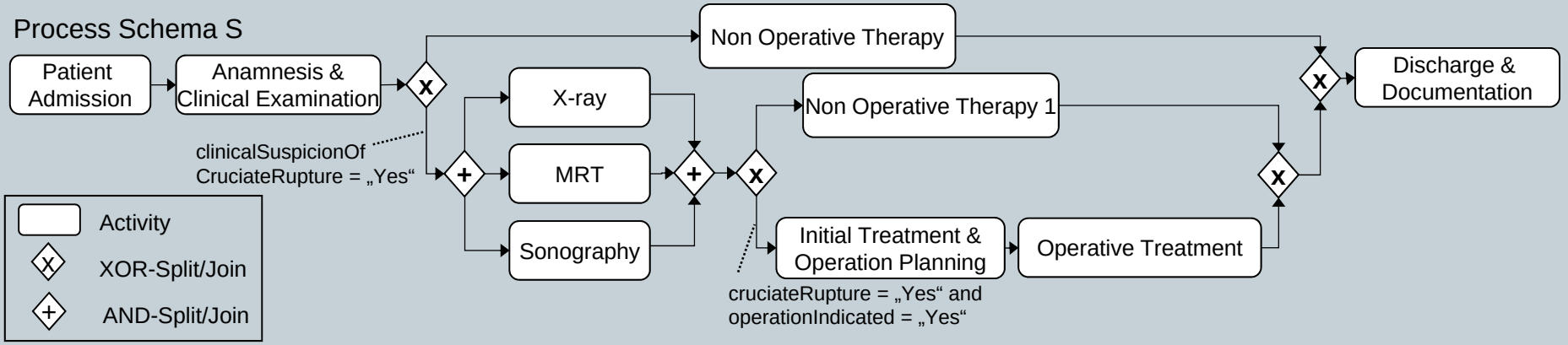
© B. Weber and M. Reichert, 2012

# Correctness of Process Instance Changes



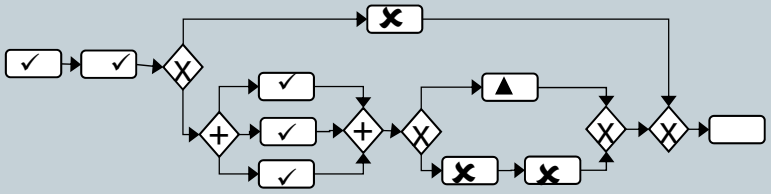
## Process Type Level

Process Schema S




## Process Instance Level

Process Instance I3



**Execution Trace:**
$\sigma_3$ = < „Patient Admission“, „Anamnesis & Clinical Examination“, „MRT“, „X-ray“, „Sonography“>

I3 is not state compliant with change Delete (I3, MRT)

[ReDa98]

© B. Weber and M. Reichert, 2012

# User Assistance & Change Reuse (1)



**The ProCycle (= ADEPT + CBRFlow) Approach for Assisting Users in Defining and Reusing Changes:**

- Annotate ad-hoc changes with information about the reasons for their introduction
- Support users in retrieving past ad-hoc changes applied in similar context
- Assist users in reusing a past ad-hoc change when coping with an exceptional situation

[RWR+05, WRW+09, WRR+05, WRW06, WWB04]

© B. Weber and M. Reichert, 2012

# User Assistance & Change Reuse (2)

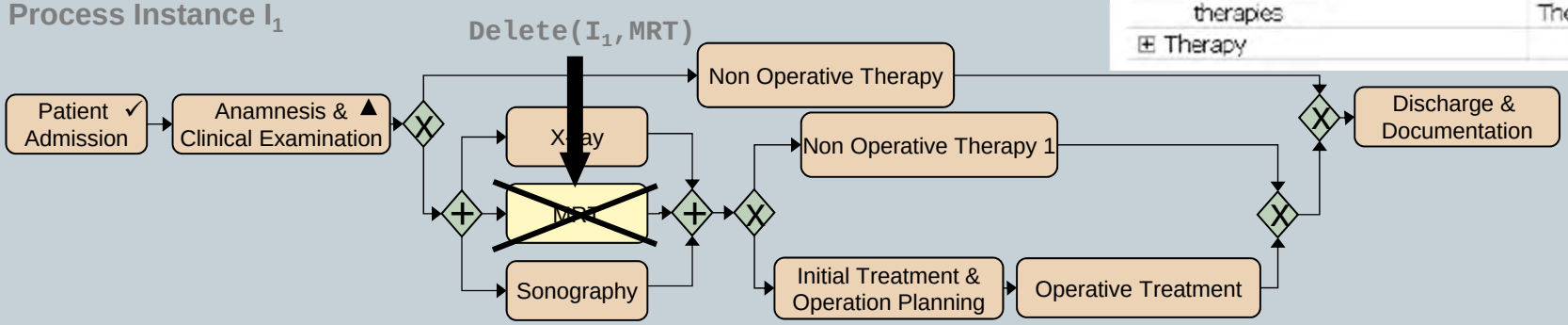


## Memorization of instance deviations including their application context

### Application Context Model

| Object | Type | Min | Max |
|---|---|---|---|
| ⊞ Diagnosis | | | |
| ⊟ Patient | | | |
| age | Integer | 1 | 1 |
| problemList | ProblemList | 1 | 1 |
| weight | Integer | 1 | 1 |
| ⊟ ProblemList | | | |
| diagnoses | Diagnosis | 0 | 1000 |
| hasPacemaker | Boolean | 0 | 1 |
| therapies | Therapy | 0 | 1000 |
| ⊞ Therapy | | | |

Process Instance $I_1$




**Case $c_1$**

$pd_{c_1}$ = The treatment of cruciate ruptures routinely includes a magnetic resonance tomography (MRT), an X-ray and a sonography. However, for a particular patient the MRT may have to be skipped as the respective patient has a cardiac pacemaker.

$sol_{c_1}$ = <Delete($S_I$,MRT)>

$qaSet_{c_1}$ = {(Does the patient have a cardiac pacemaker?, patient.problemList.hasPacemaker = 'Yes')}

$freq_{c_1}$ = 1

# Business Processes and Workflows

## Process Evolution



BARBARA WEBER
UNIVERSITY OF INNSBRUCK

MANFRED REICHERT
ULM UNIVERSITY

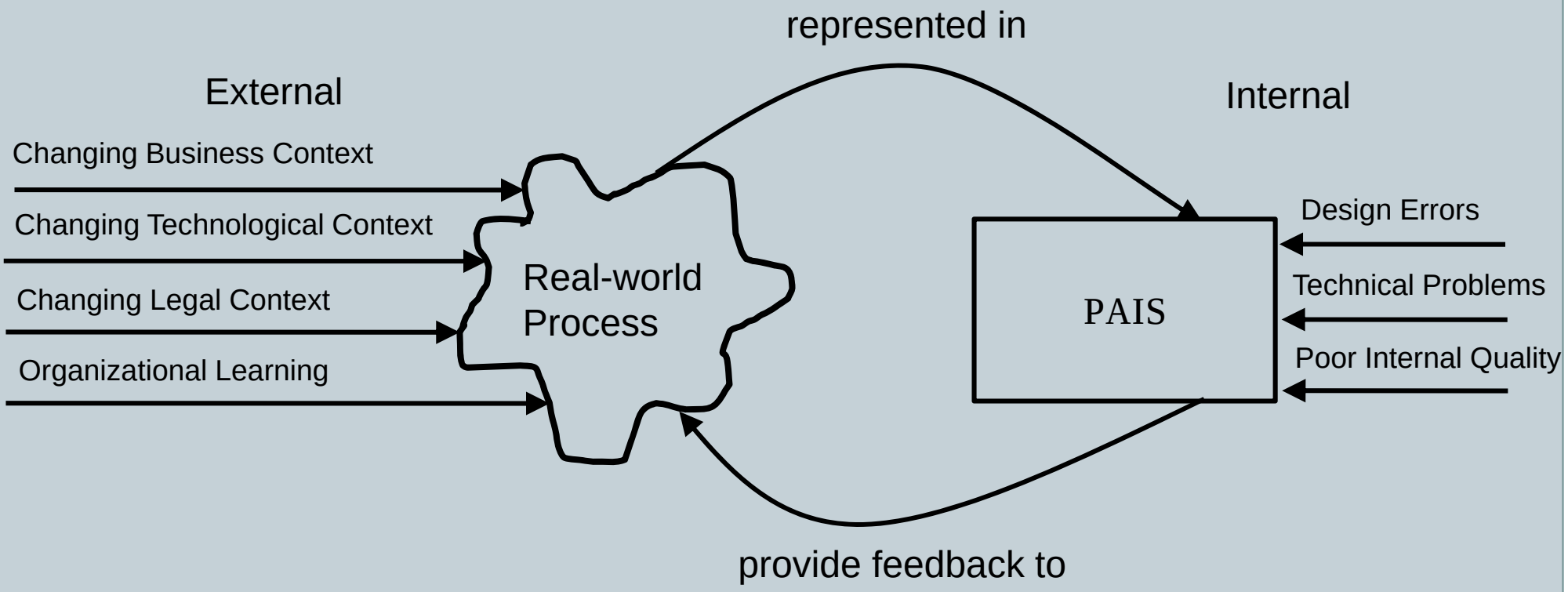
ER 2012, FLORENCE

© B. Weber and M. Reichert, 2012

# Evolution



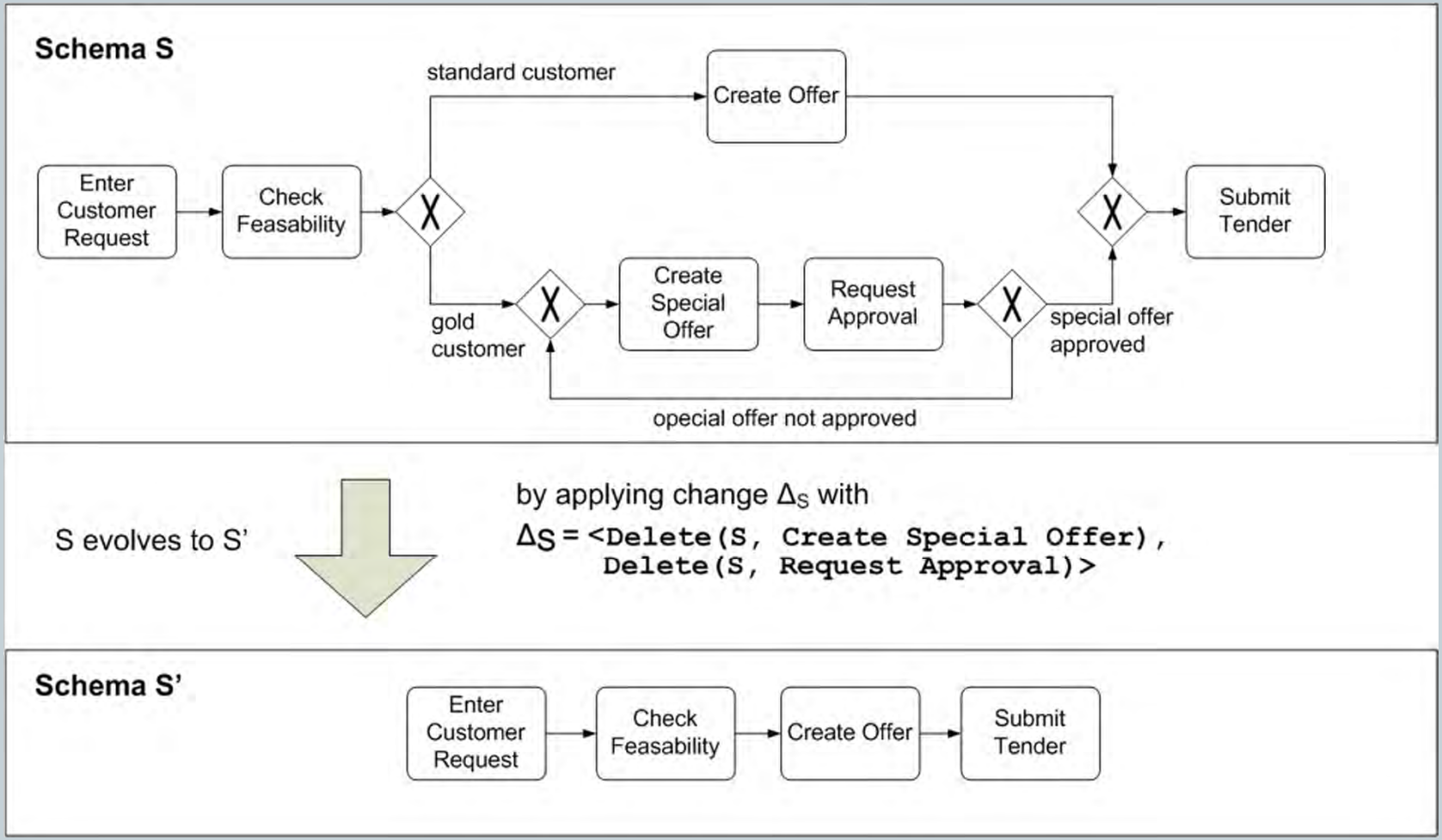
- Drivers




© B. Weber and M. Reichert, 2012

# Schema Evolution






© B. Weber and M. Reichert, 2012

# Change Support Features
## Schema Evolution, Version Control and Instance Migration



- Schema Evolution
  - Changes at the process type level
- How to deal with running instances when adapting the original process schema?
  - Scenario 1: No version control
  - Scenario 2: Co-existence of instances of old / new schema
  - Scenario 3: Change propagation and instance migration

© B. Weber and M. Reichert, 2012

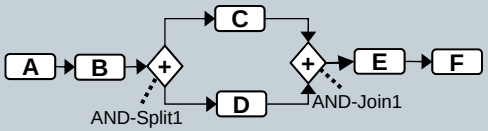
# Scenario 1 - No Version Control



- Schema is overwritten and instances are migrated

***Type change overwrites schema S***

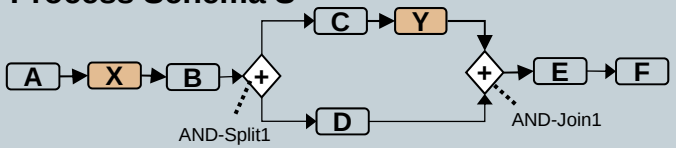
**Process Schema S**



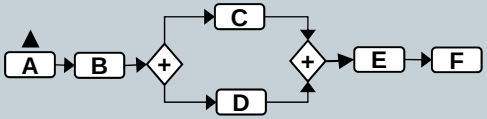

**Insert X between A and B**
**Insert Y between C and AND-Join1**

**Schema Evolution**

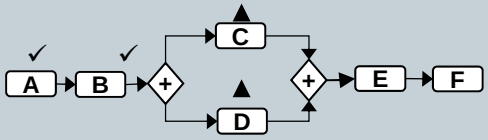
**Process Schema S'**




**Process Instance I1**

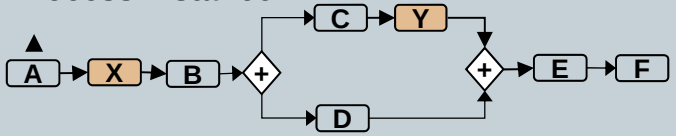



**Process Instance I2**

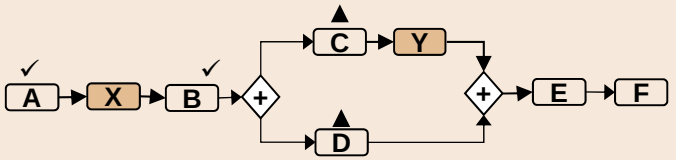



**Change is propagated to all running process instances**

**Process Instance I1**




**Process Instance I2**




**Inconsistent state**

© B. Weber and M. Reichert, 2012

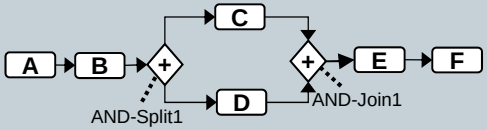
# Scenario 2 - Version Control



- Co-existence of instances of different schema versions

***Type change results into a new version of schema S***

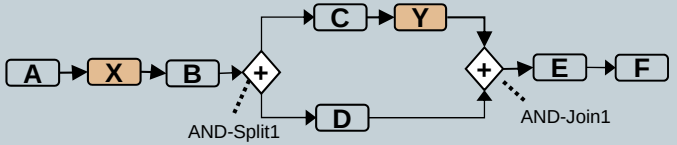
**Process Schema S**




**Insert X between A and B**
**Insert Y between C and AND-Join1**

**Schema Evolution**

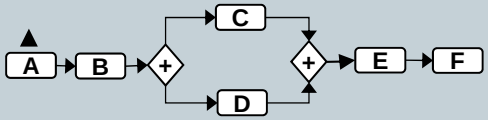
**Process Schema S'**



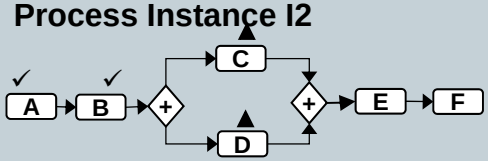

***Old instances remain with schema S***

**Instances created from S (before schema evolution)**

**Process Instance I1**

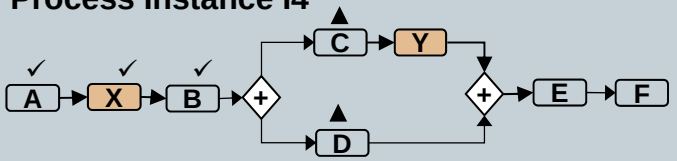



**Process Instance I2**

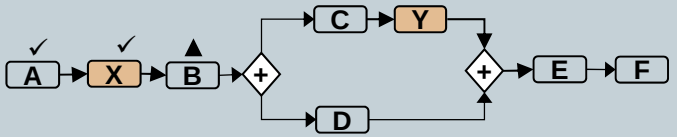



**Instances created from S' (after schema evolution)**

**Process Instance I4**




**Process Instance I5**




© B. Weber and M. Reichert, 2012

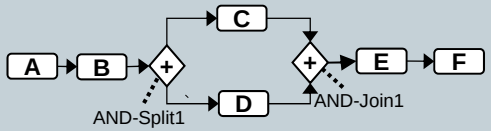
# Scenario 3 – Instance Migration



- Compliant instances are migrated to the new schema

***Type change results into a new version of schema S***

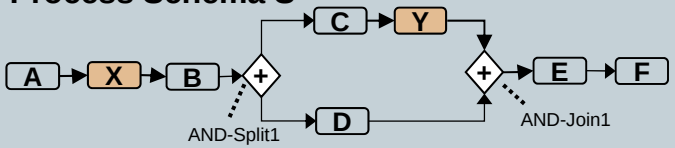
**Process Schema S**




**Insert X between A and B**
**Insert Y between C and AND-Join1**

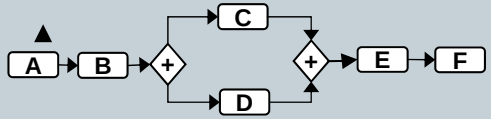
**Schema Evolution**

**Process Schema S'**

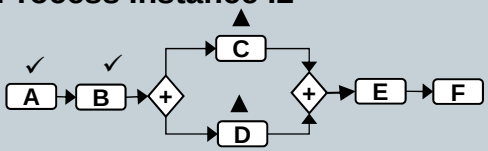



***Migration of compliant process instances to S'***

**Process Instance I1**

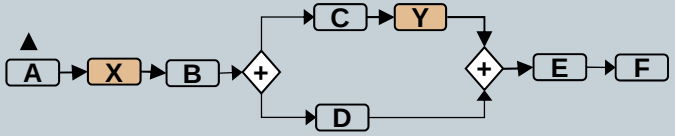



**Process Instance I2**




**Propagation of compliant process instances to schema S' (incl. state adaptations)**

**Process Instance I1**




**Process Instance $I_2$ not compliant with S'**

[RRD04a]

© B. Weber and M. Reichert, 2012

# Process Model Refactoring



(1) Identify refactoring opportunities

(2) Determine which refactoring should be applied

(3) Ensure that the applied refactoring preserves model behavior

(4) Apply the refactoring

(5) Assess the effect of the refactoring on the quality characteristics of the process model repository

© B. Weber and M. Reichert, 2012

# Catalogue of Process Model Smells



PMS1: Non Intension Revealing Naming of Activity / Process Model

PMS2: Contrived Flexibility

PMS4: Large Process Models

PMS3: Redundant Process Fragment

PMS5: Lazy Process Model

PMS6: Unused Branches

PMS7: Frequently Occurring Instance Change

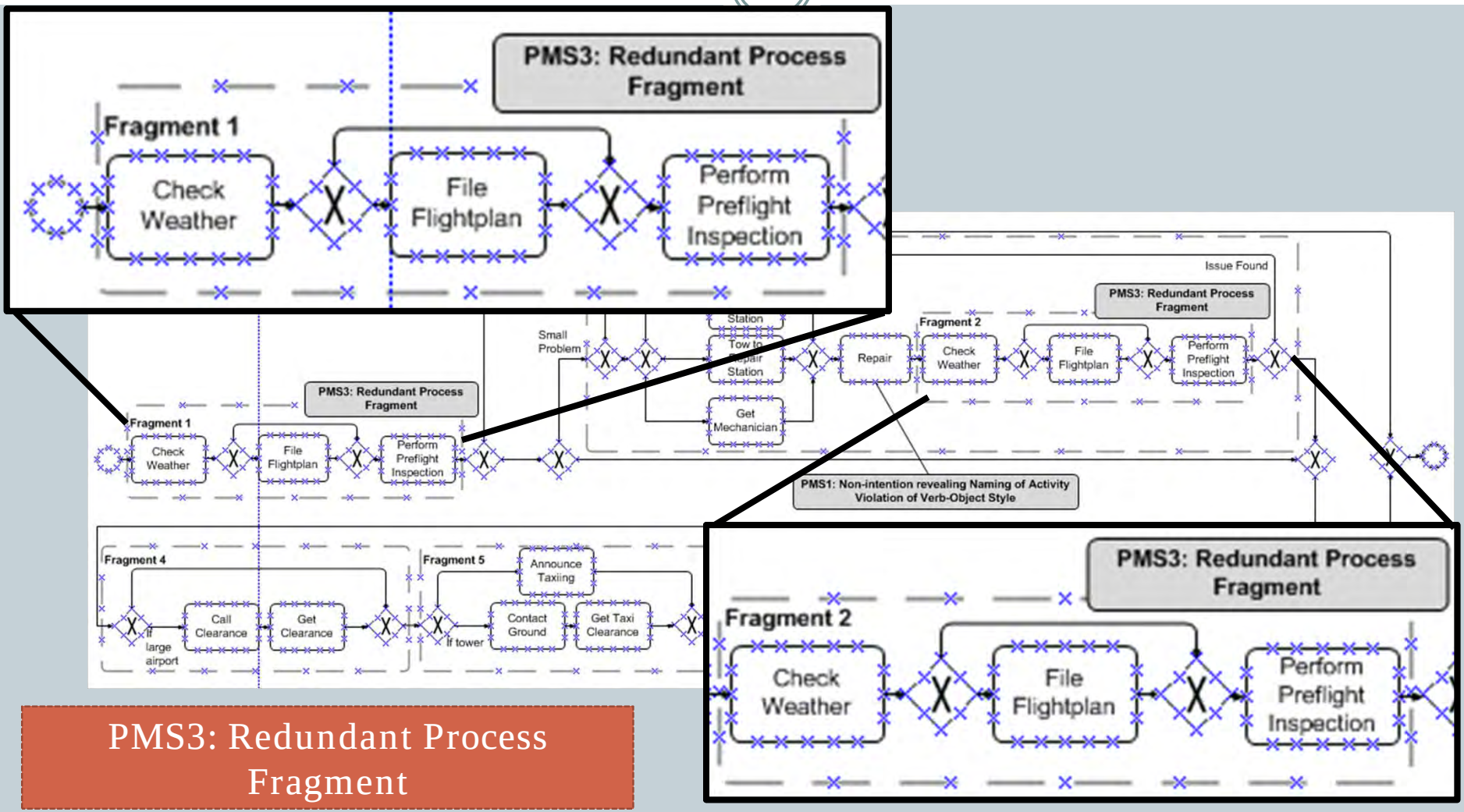
PMS8: Frequently Occurring Variant Change

[WeRe08, WRR+xx]

© B. Weber and M. Reichert, 2012

# Process Model Smells: Example



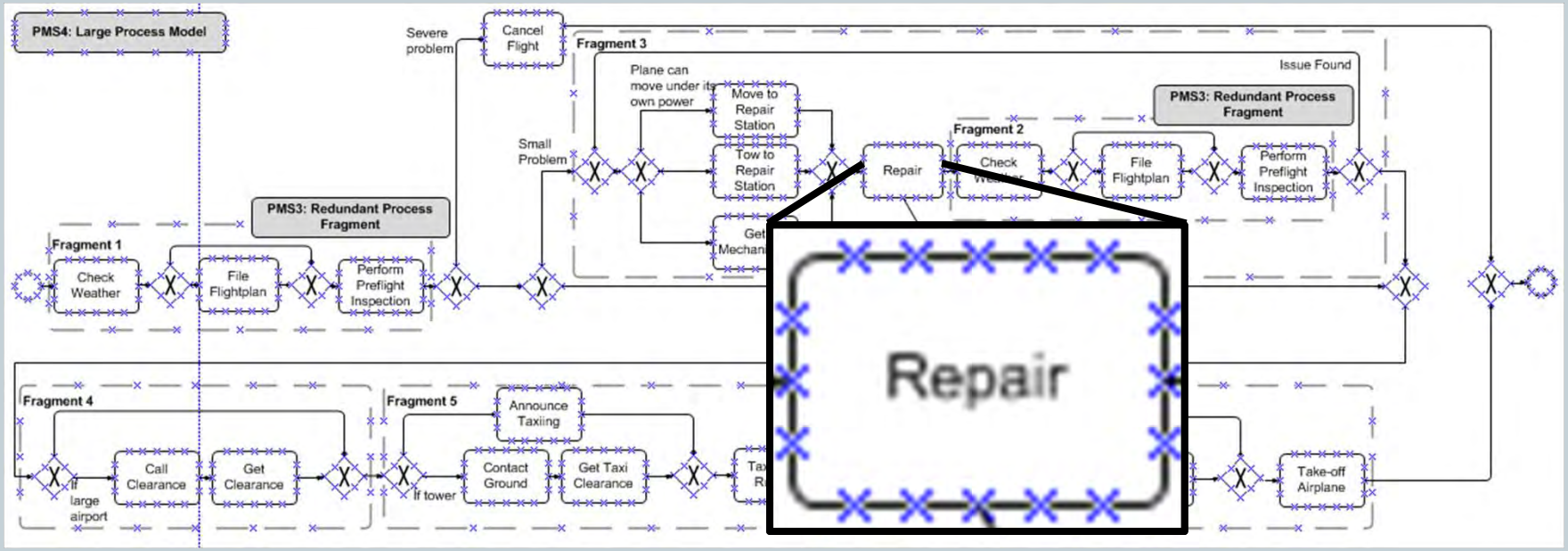



PMS3: Redundant Process Fragment

© B. Weber and M. Reichert, 2012

# Process Model Smells: Example





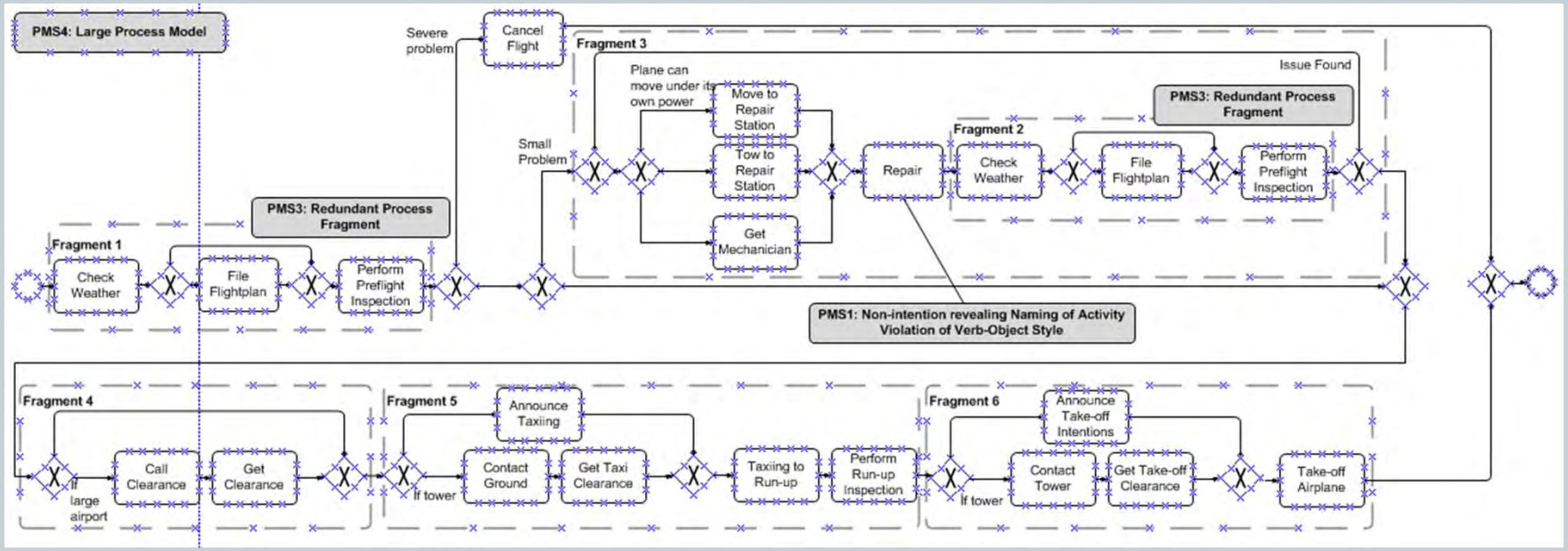

PMS1: : Non Intension Revealing Naming of Activity

© B. Weber and M. Reichert, 2012

# Process Model Smells: Example



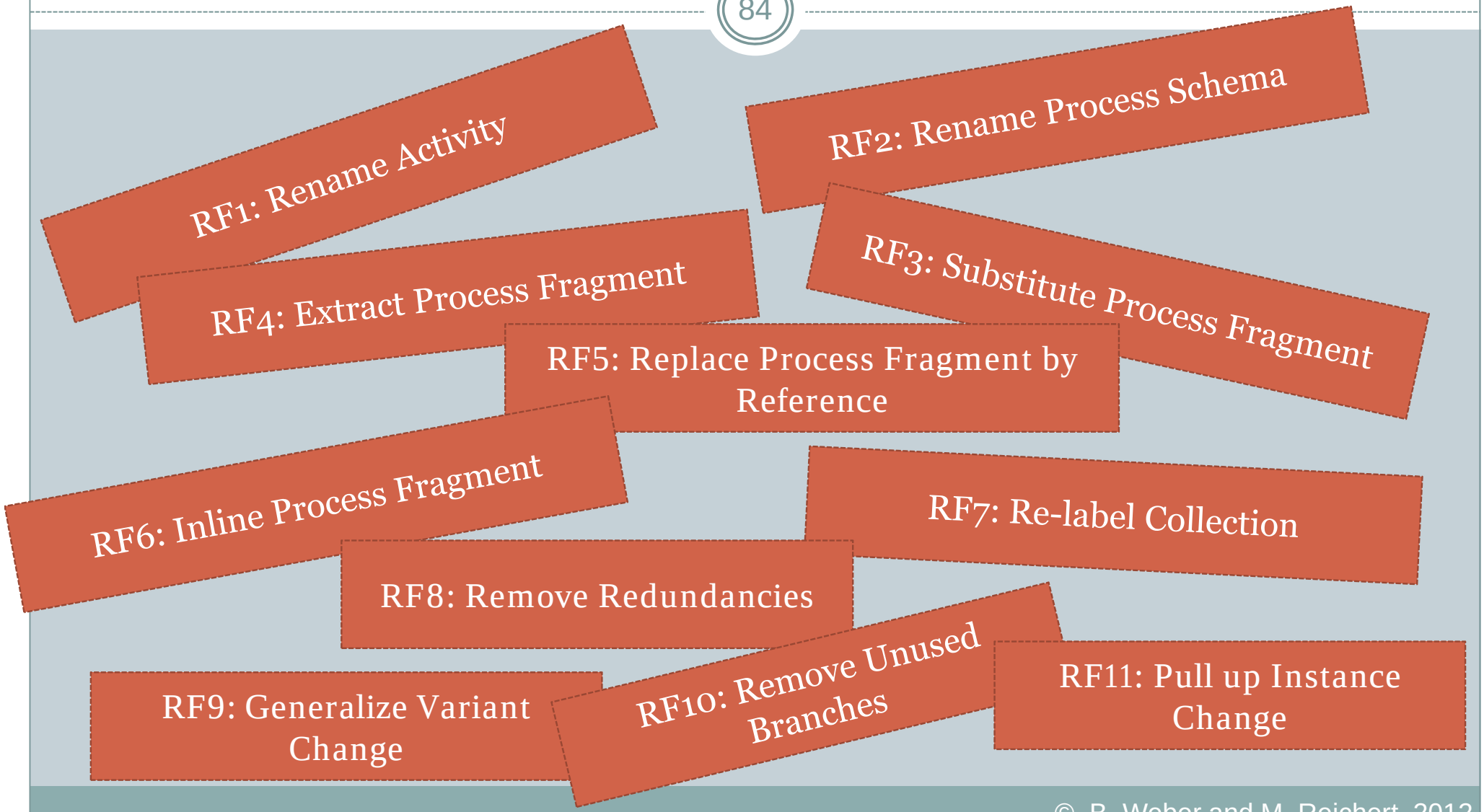
## PMS4: Large Process Model




© B. Weber and M. Reichert, 2012

# Catalogue of Process Model Refactorings

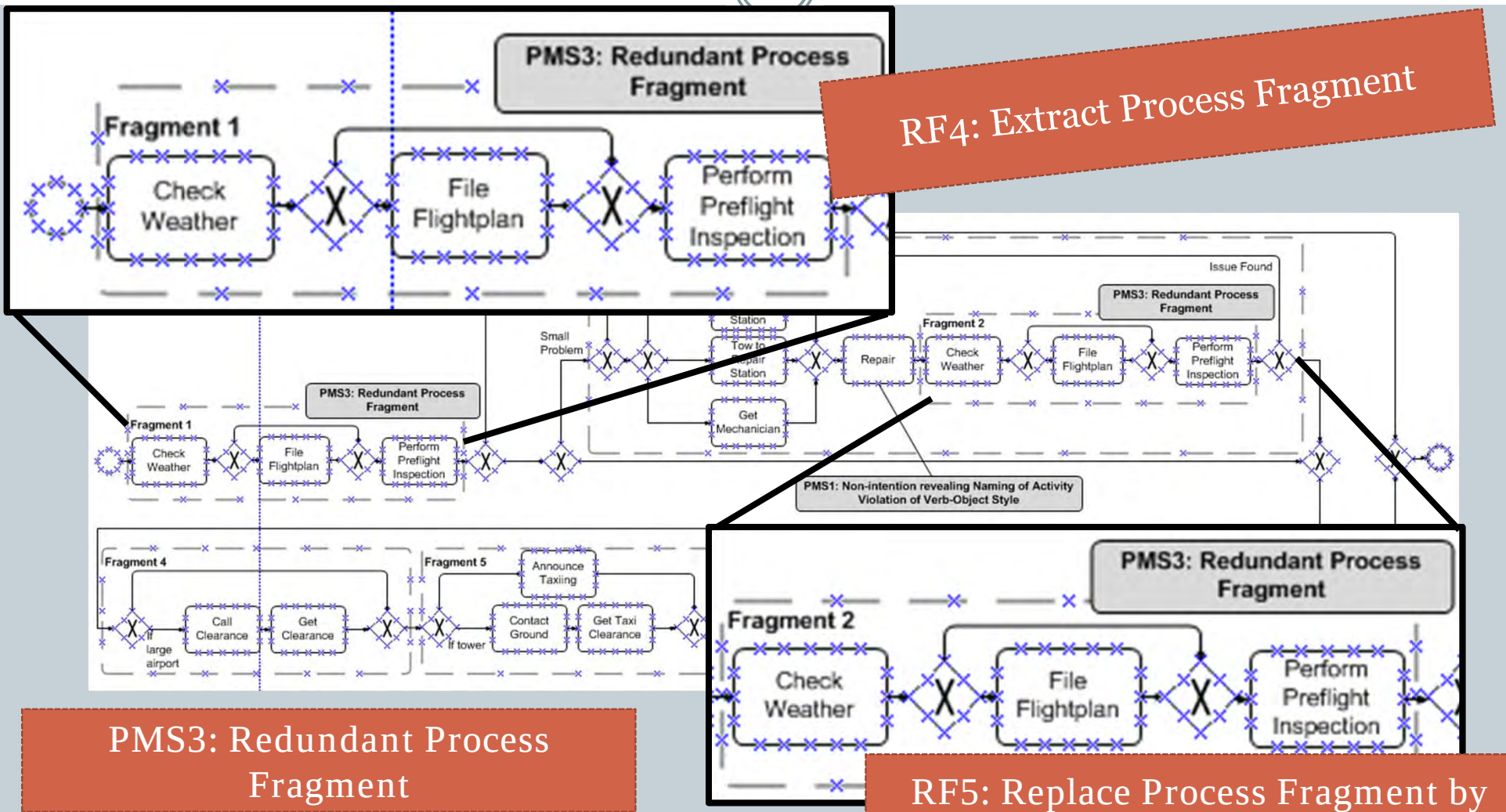





© B. Weber and M. Reichert, 2012

# Process Model Smells: Example

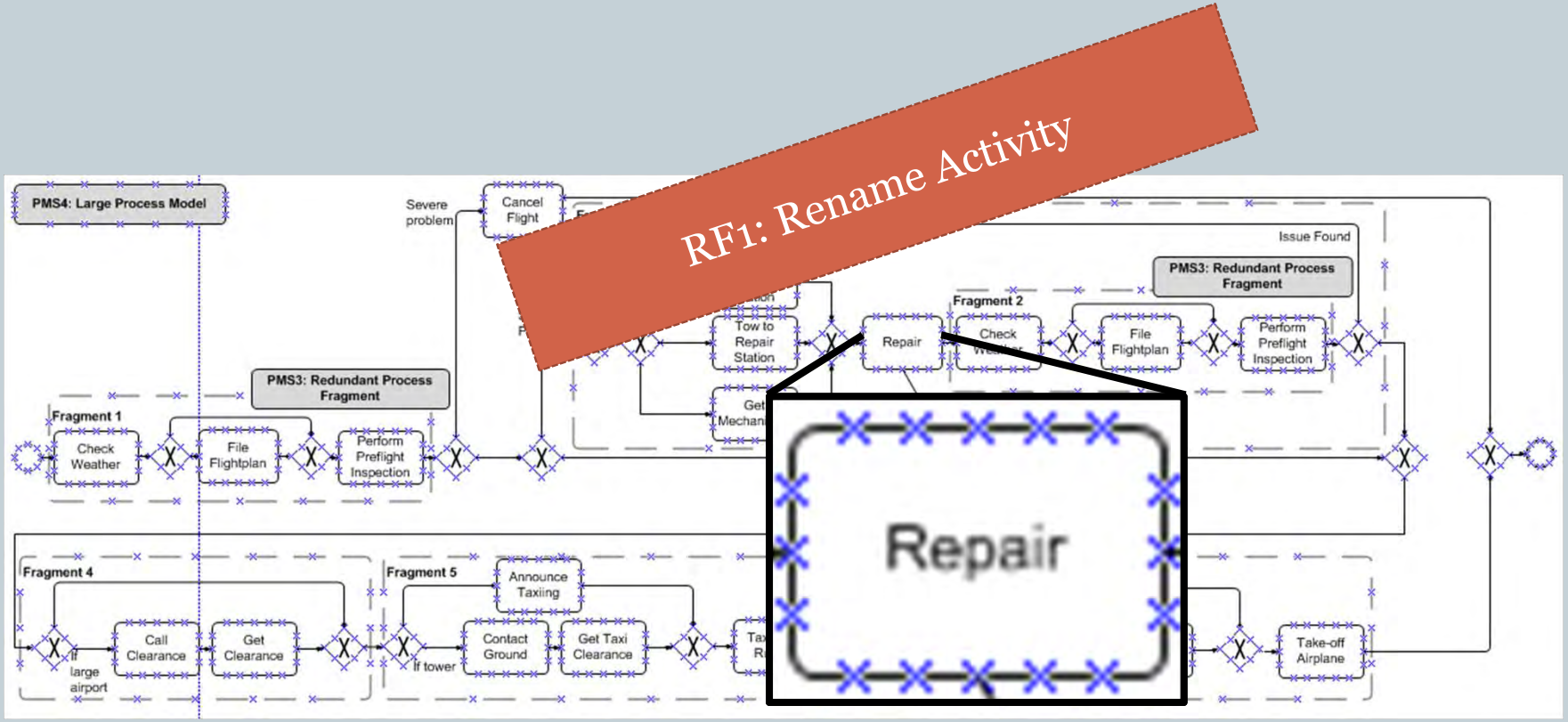





© B. Weber and M. Reichert, 2012

# Process Model Smells: Example





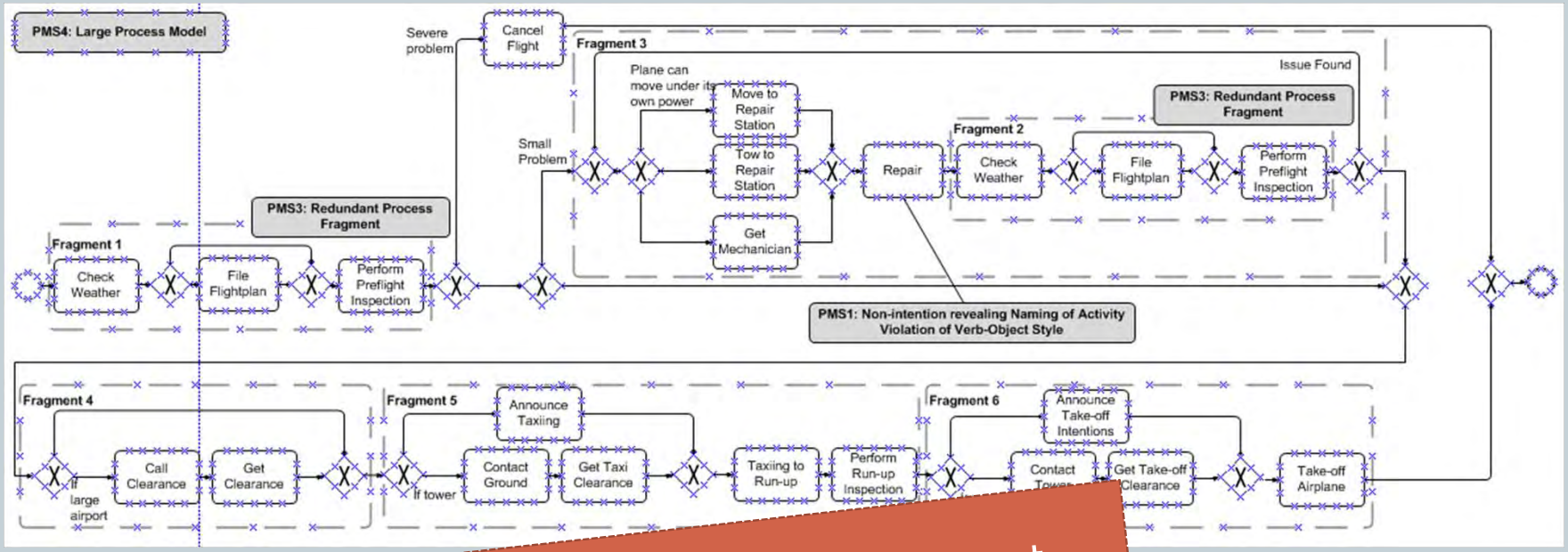

PMS1: : Non Intension Revealing Naming of Activity

© B. Weber and M. Reichert, 2012

# Process Model Smells: Example



PMS4: Large Process Model

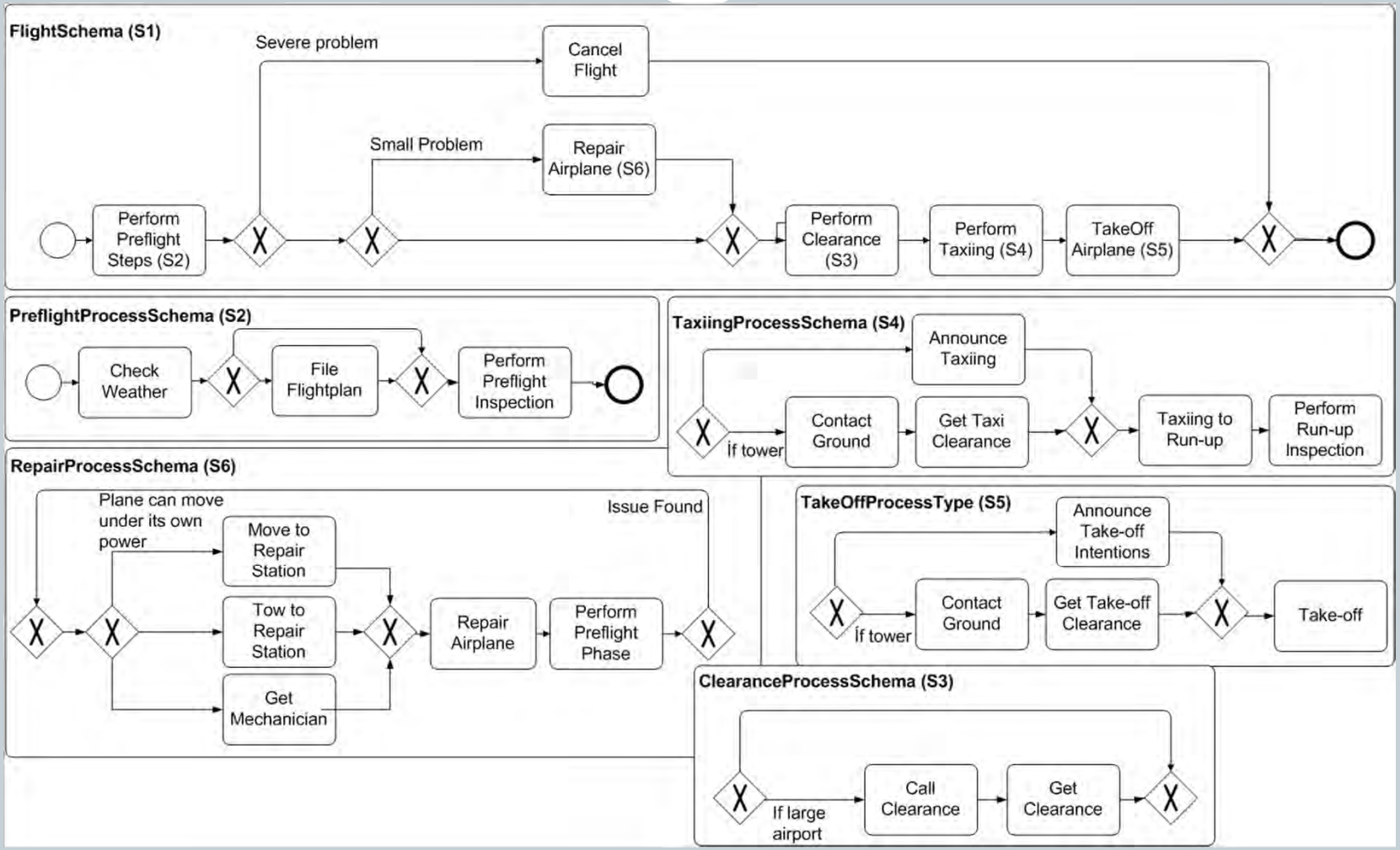



RF4: Extract Process Fragment

© B. Weber and M. Reichert, 2012

# Process Model After Refactoring

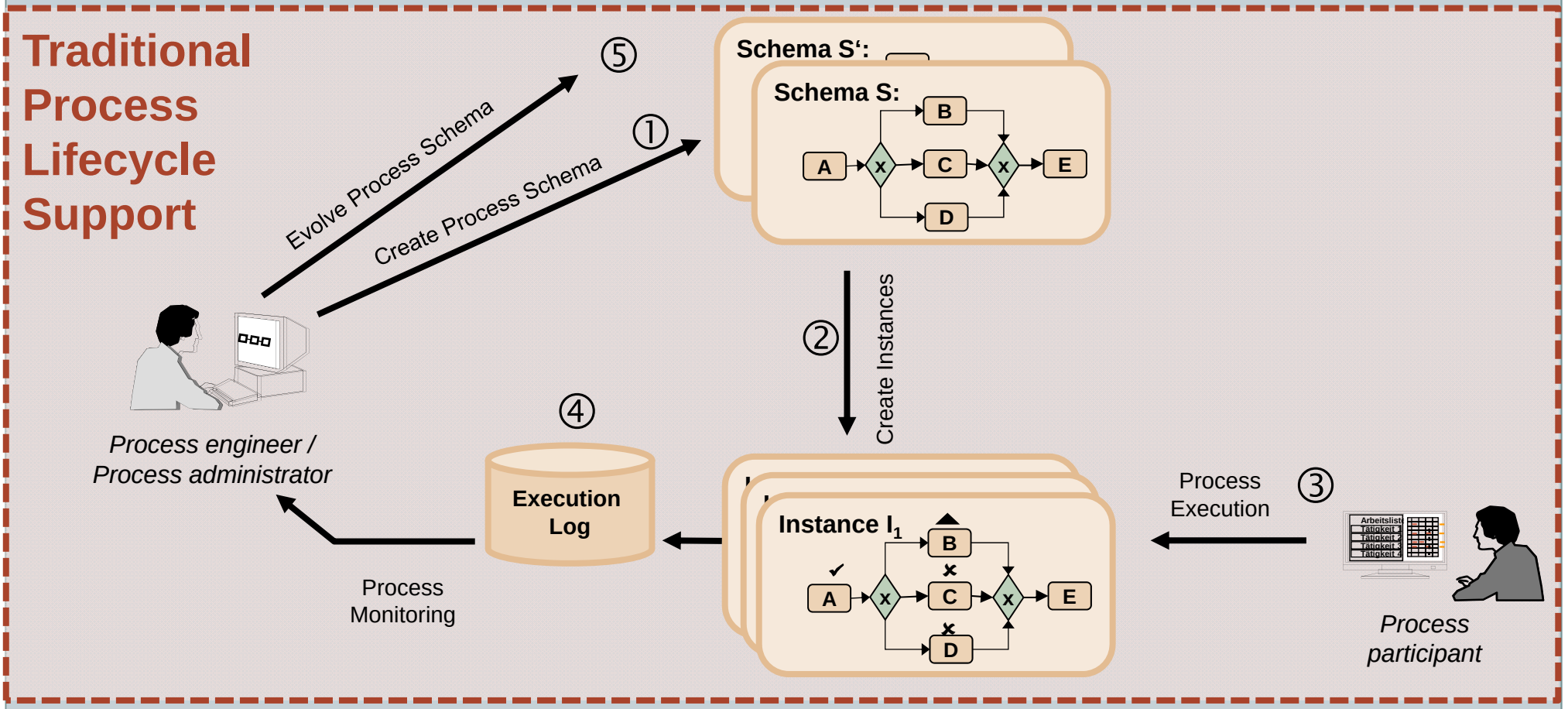





© B. Weber and M. Reichert, 2012

# Integrated Lifecycle Support for Adaptive and Dynamic Processes (1)

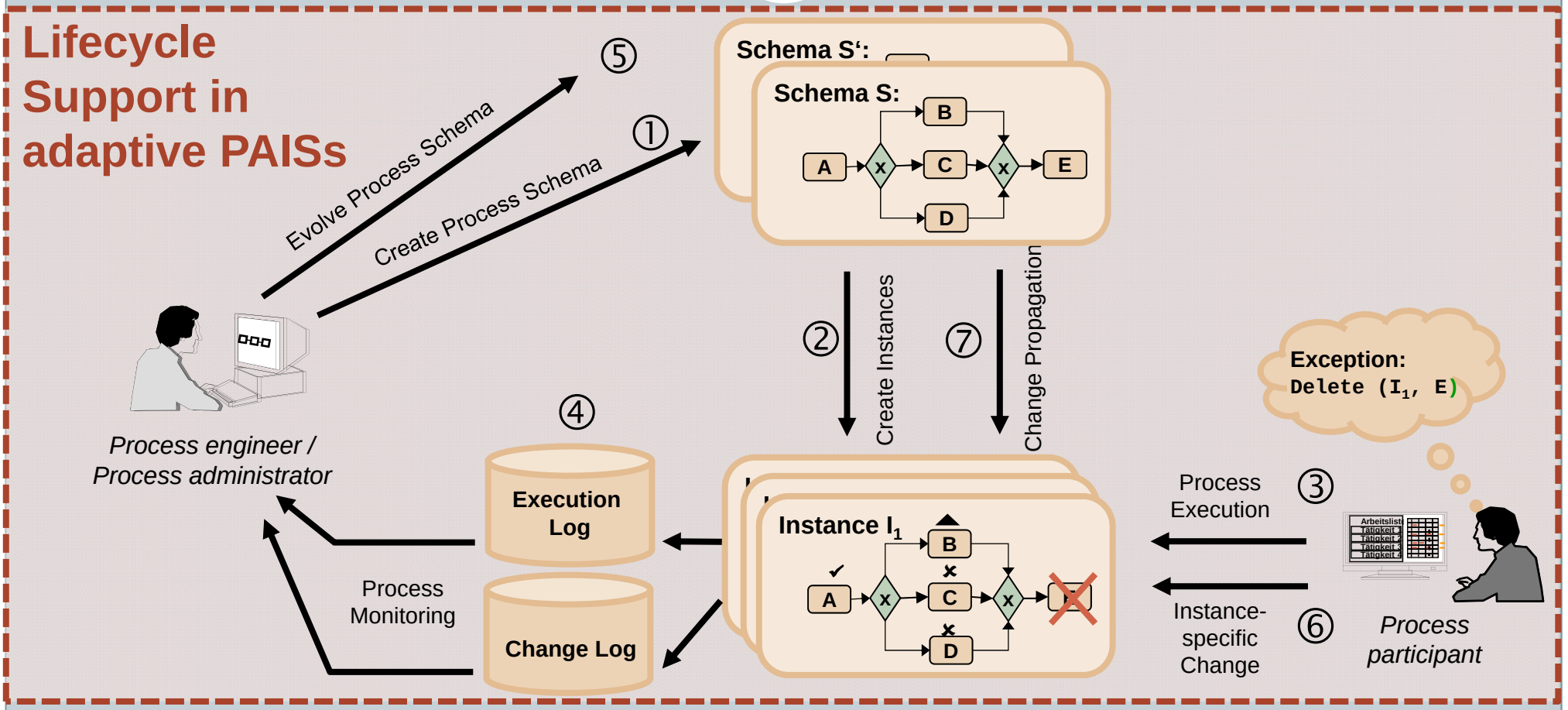





© B. Weber and M. Reichert, 2012

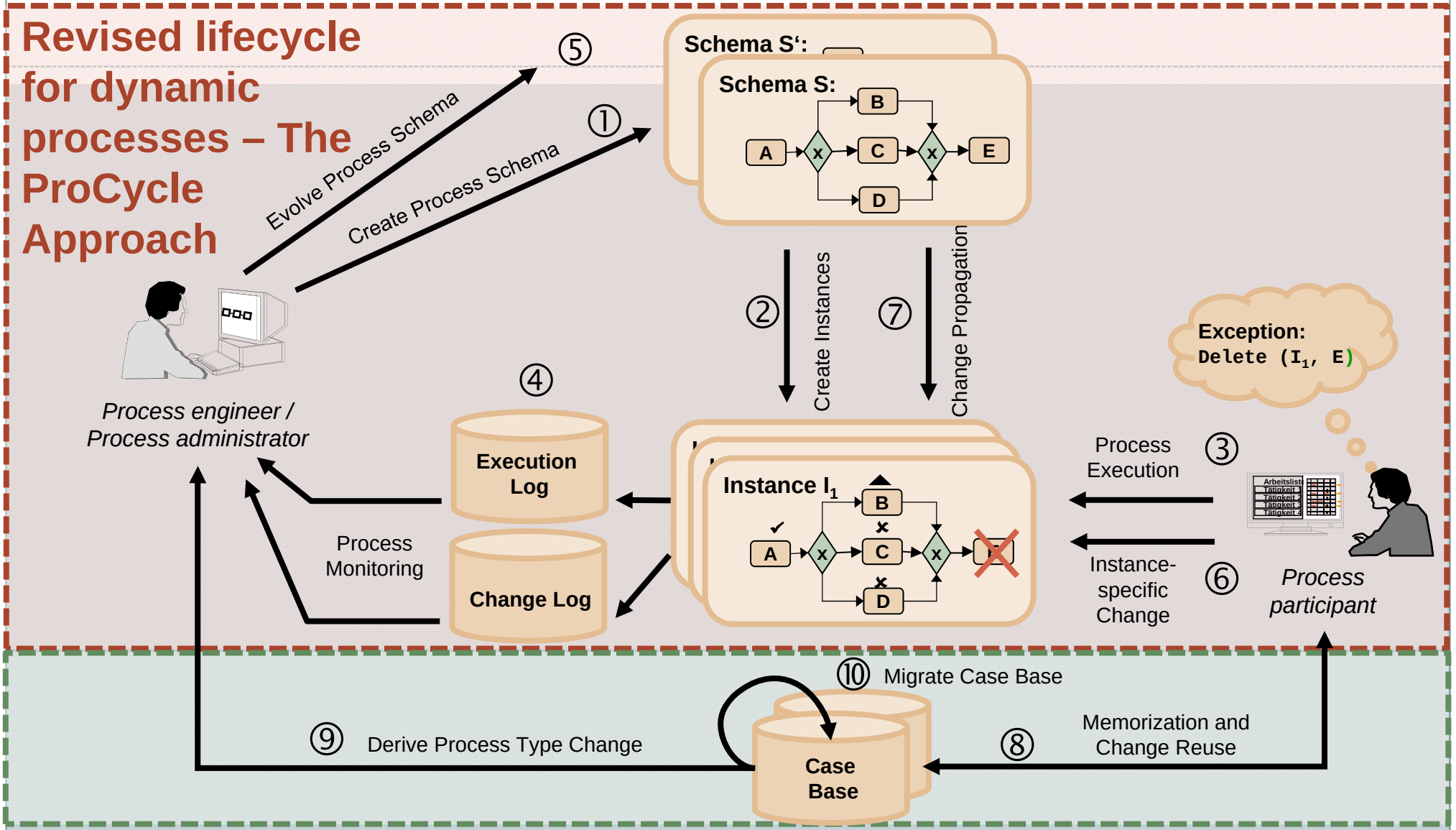
# Integrated Lifecycle Support for Adaptive and Dynamic Processes (2)






© B. Weber and M. Reichert, 2012

# Integrated Lifecycle Support for Adaptive and Dynamic Processes (3)




© B. Weber and M. Reichert, 2012

# Business Processes and Workflows

## Loosely Specified Processes



BARBARA WEBER
UNIVERSITY OF INNSBRUCK

MANFRED REICHERT
ULM UNIVERSITY

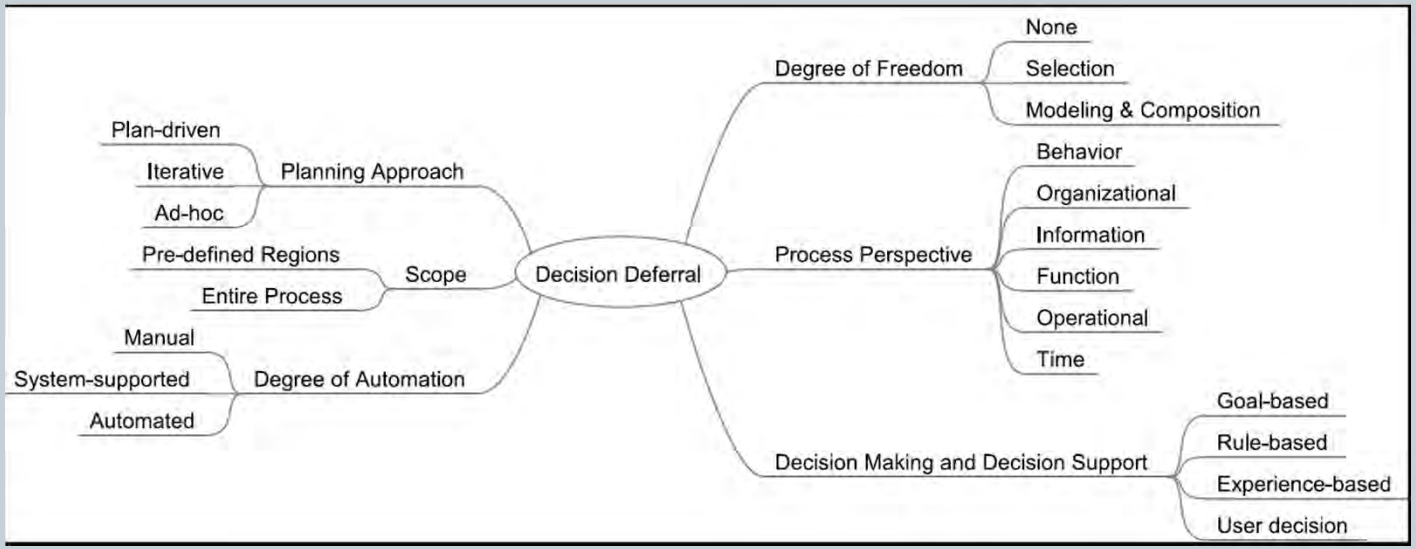
ER 2012, FLORENCE

© B. Weber and M. Reichert, 2012

# Loosely specified Processes



- To deal with unpredictability, non repeatability and emergence loosely specified processes keep (parts) of the process unspecified during build-time
- Loosely specified processes are characterized by *decision deferral ☞ taxonomy of decision deferral*




© B. Weber and M. Reichert, 2012

# Decision Deferral Patterns

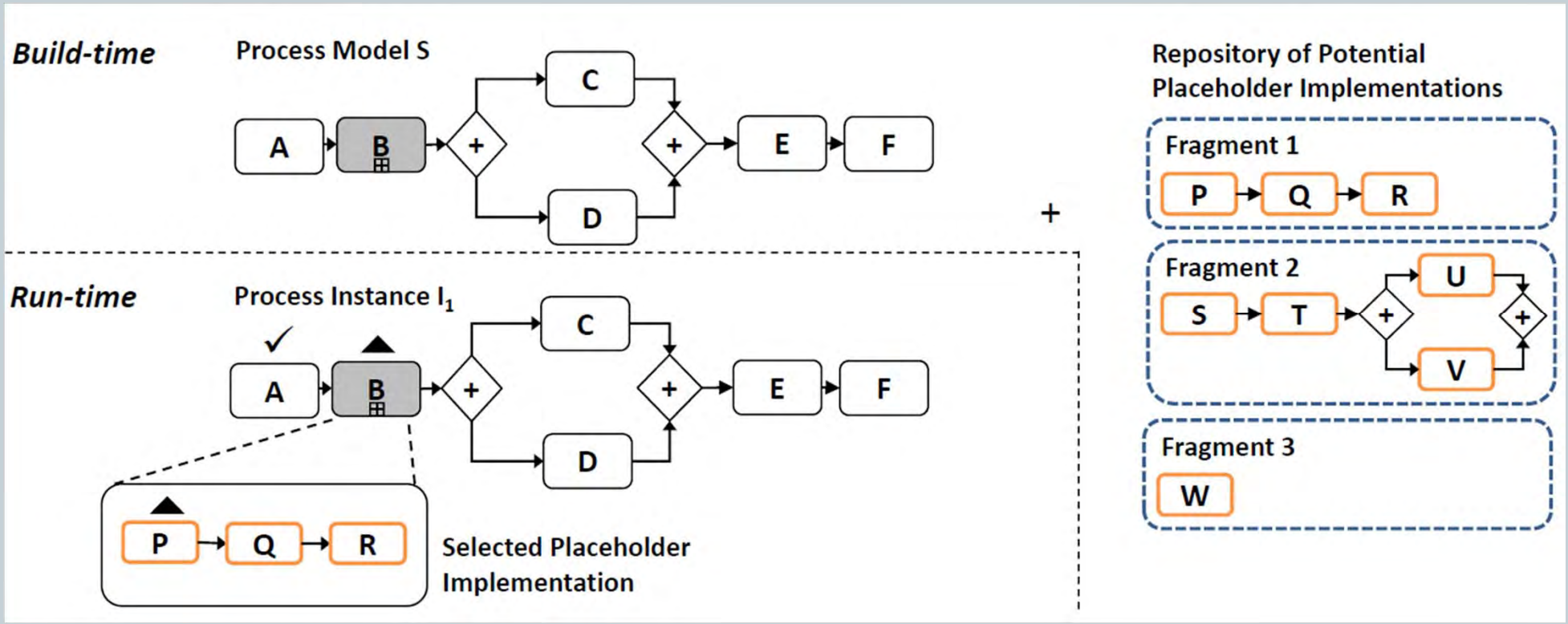


| Pattern | Degree of Freedom | Planning Approach |
|---|---|---|
| Traditional Worflow | None | Plan-driven |
| Late Selection | Selection | Plan-driven |
| Late Modeling and Composition | Modeling / Composition | Plan-driven |
| Iterative Refinement | Modeling / Composition | Iterative |
| Ad-hoc Composition | Modeling / Composition | Ad-hoc |

© B. Weber and M. Reichert, 2012

# Late Selection Pattern

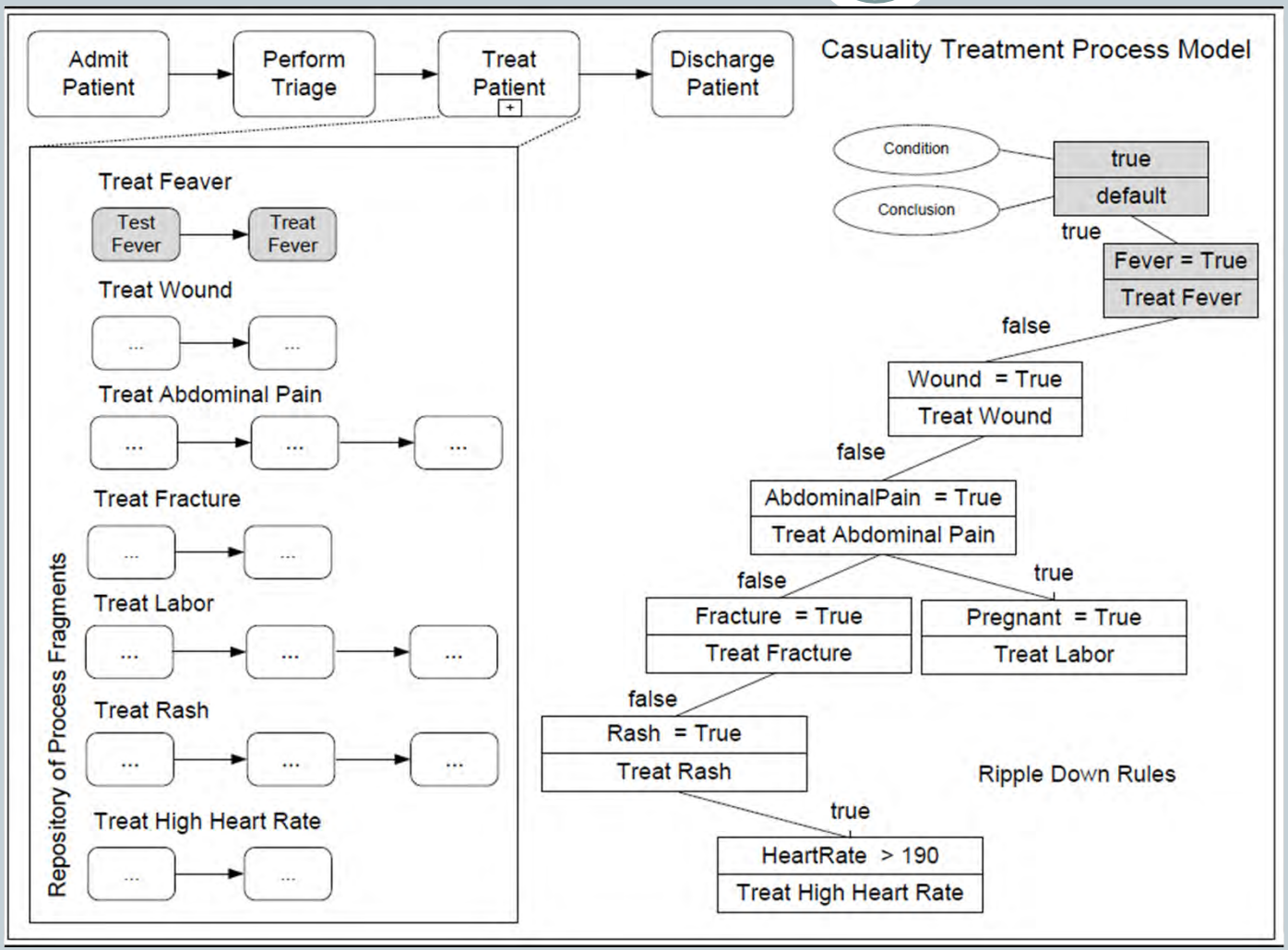





© B. Weber and M. Reichert, 2012

# Late Selection – The Worklets Approach



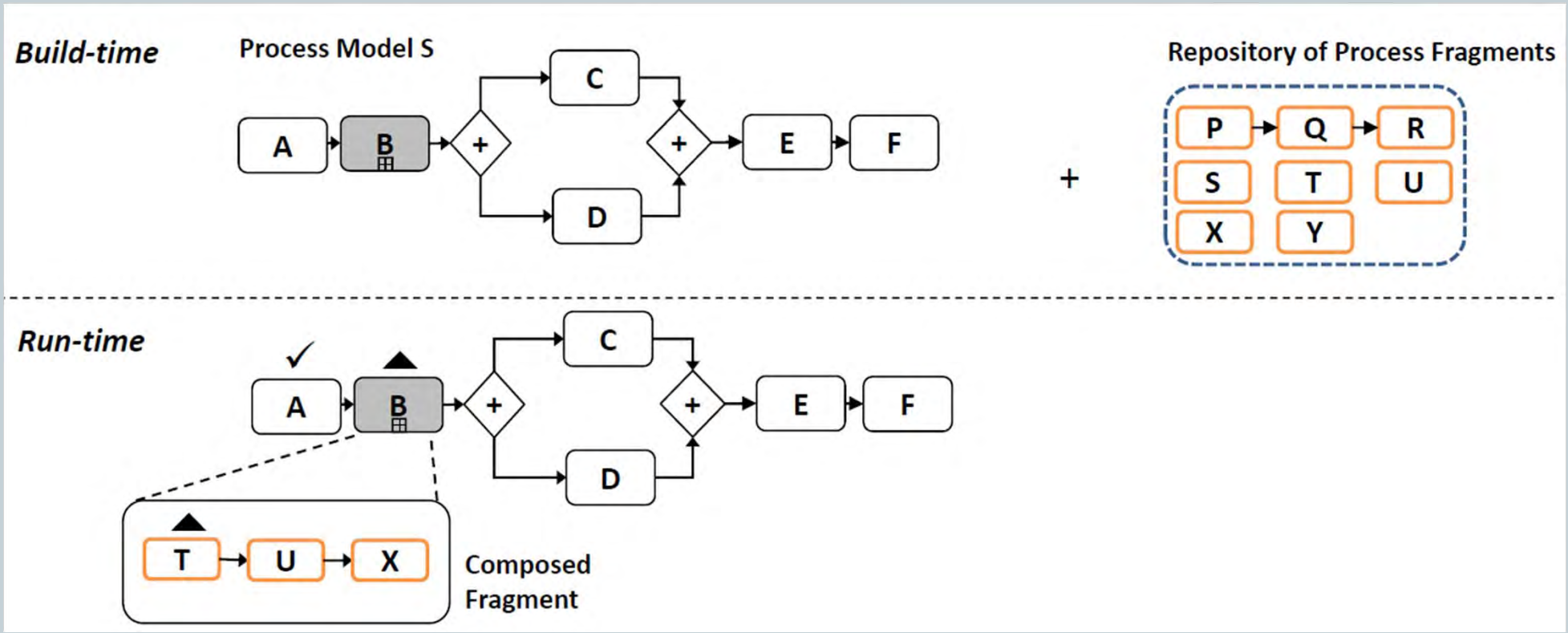



[AHE+06]

© B. Weber and M. Reichert, 2012

# Late Modeling

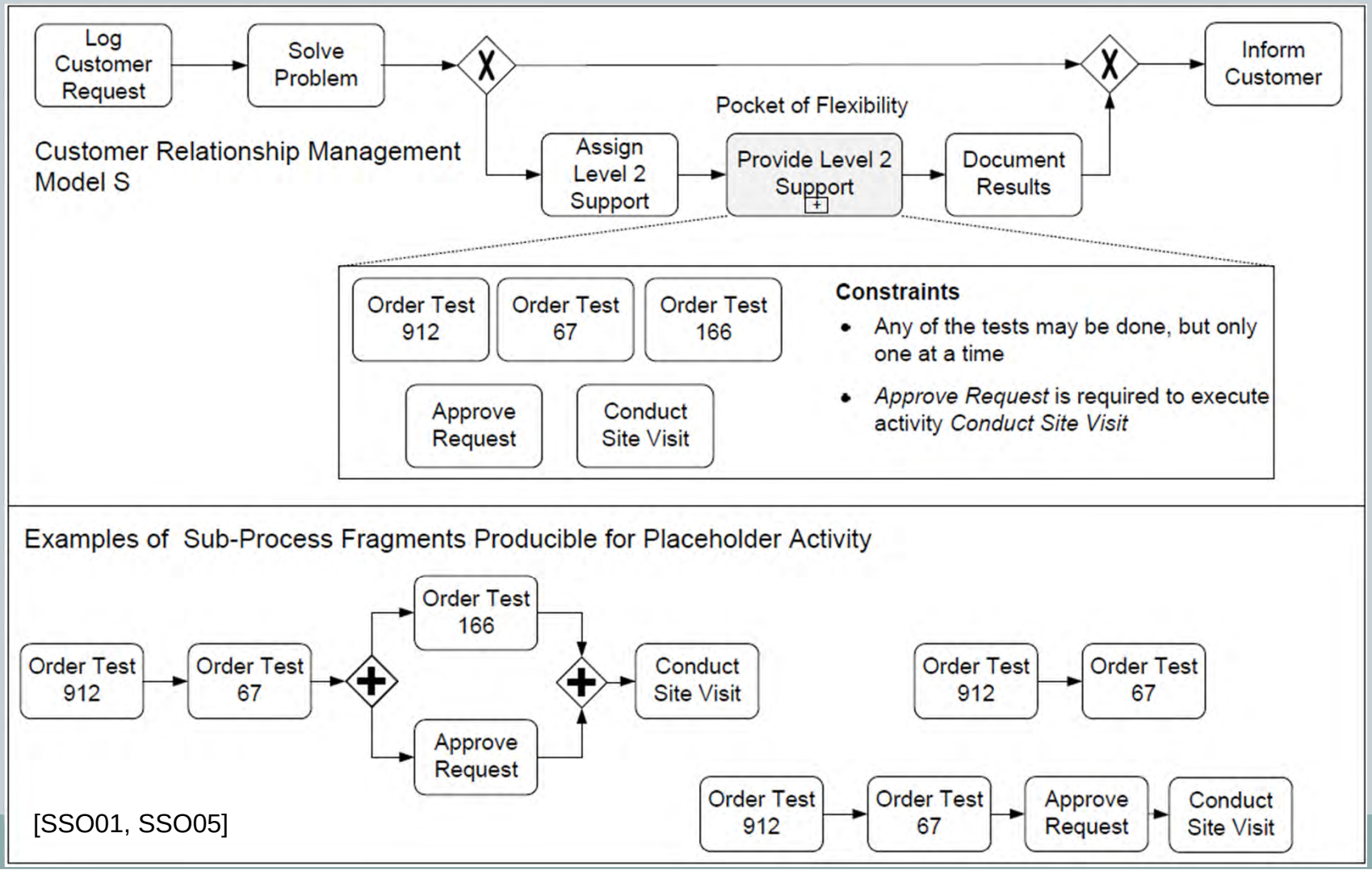





© B. Weber and M. Reichert, 2012

# Late Modeling – Pockets of Flexibility

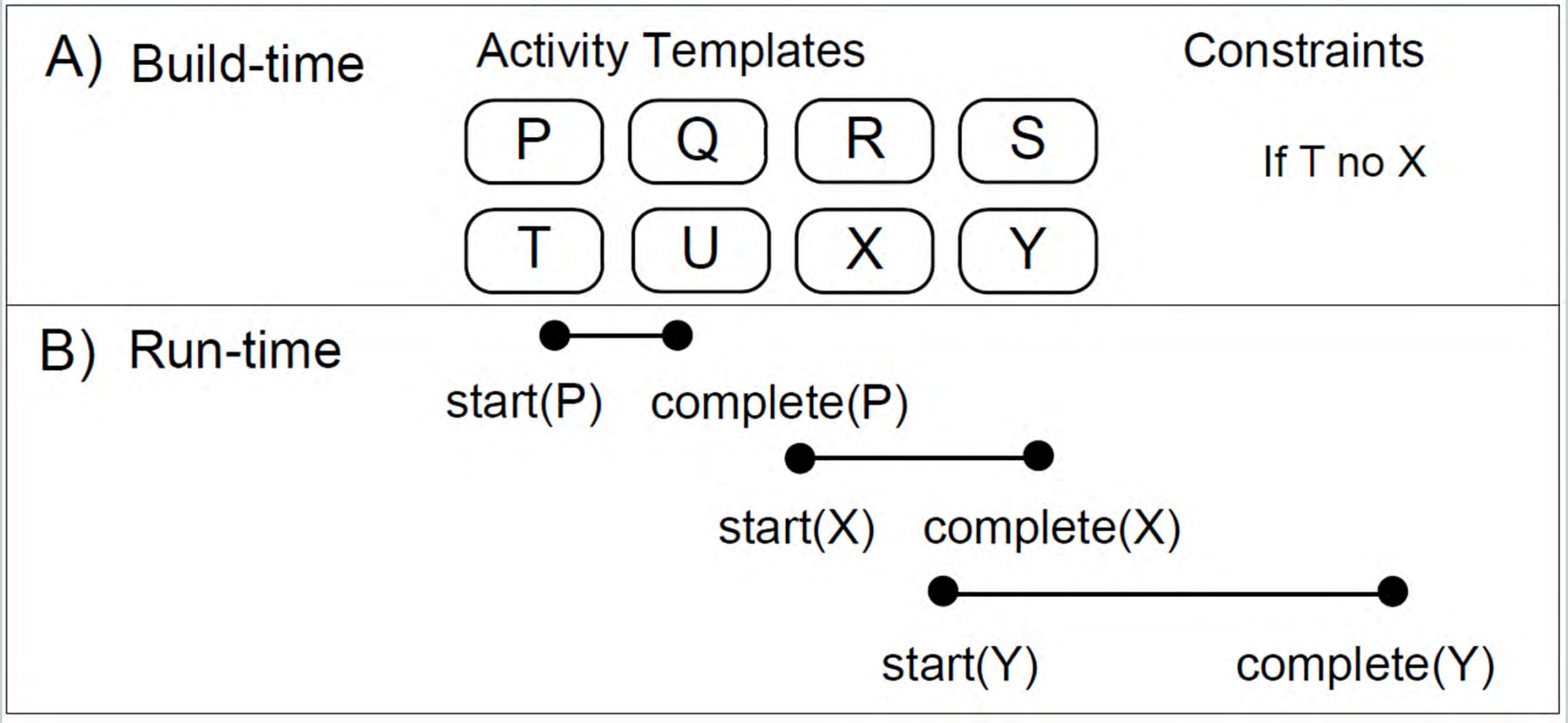





[SSO01, SSO05]

# Ad-hoc Composition - Declare

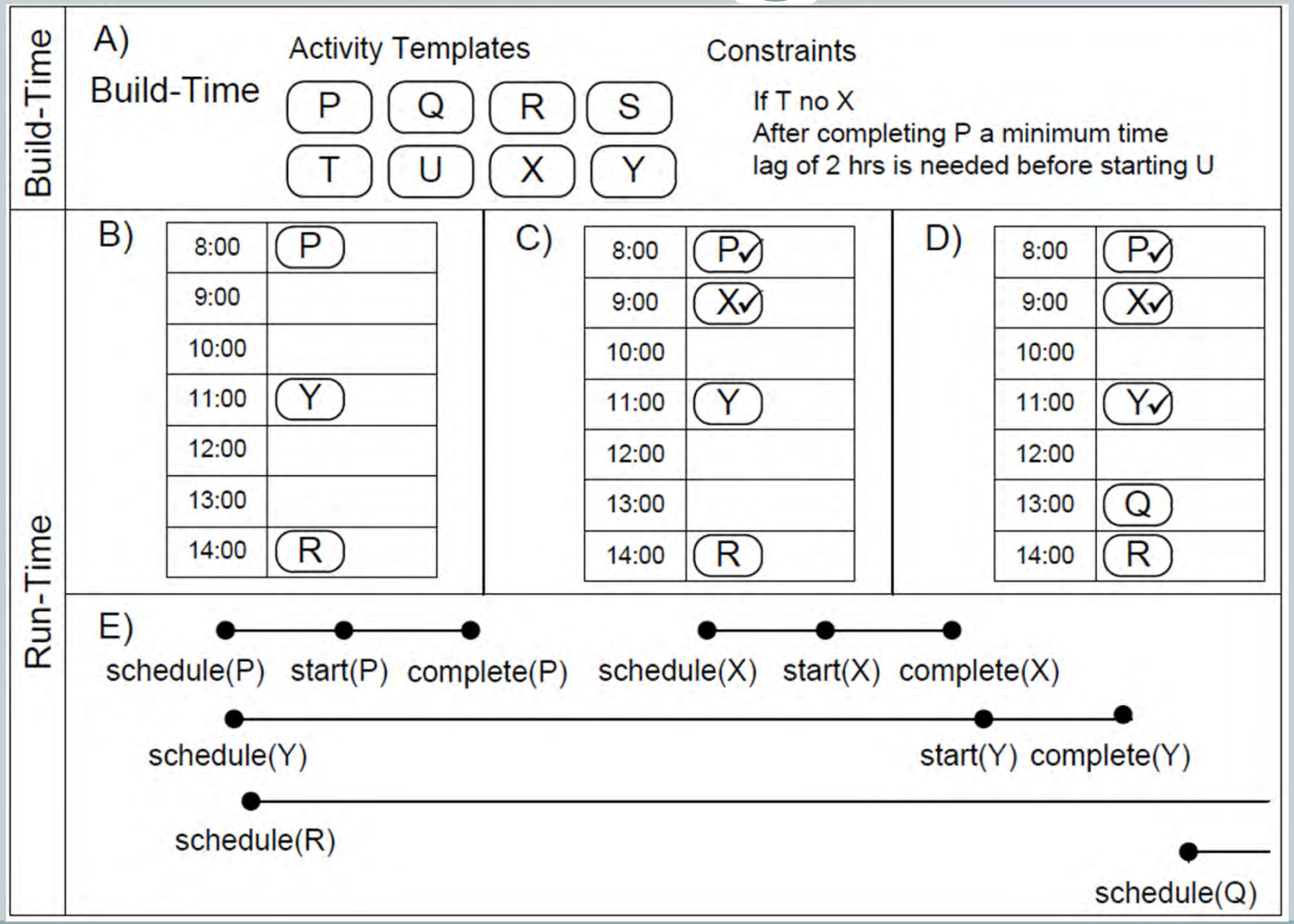





© B. Weber and M. Reichert, 2012

# Iterative Refinement - Alaska






[WZP+09]

# Business Processes and Workflows

## Declarative Processes

101

BARBARA WEBER
UNIVERSITY OF INNSBRUCK

MANFRED REICHERT
ULM UNIVERSITY

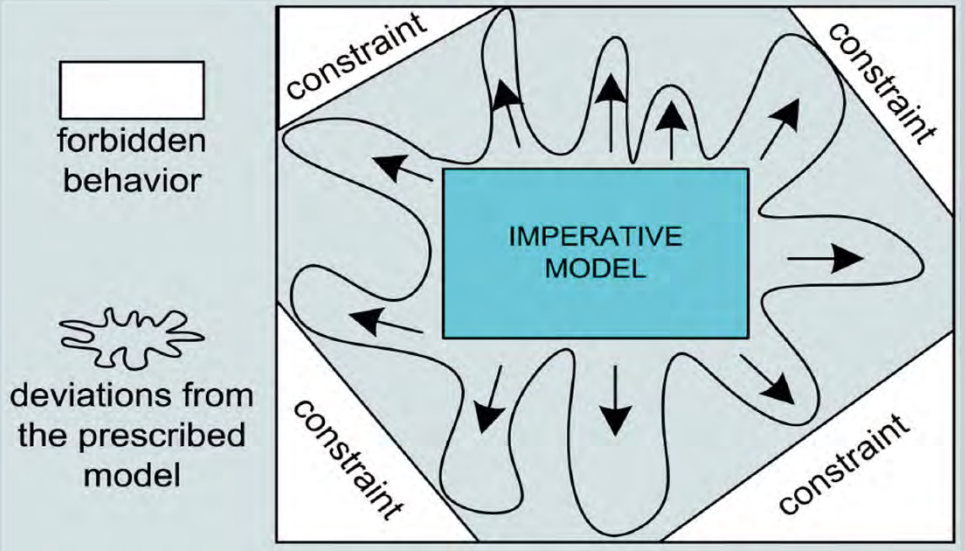
ER 2012, FLORENCE

© B. Weber and M. Reichert, 2012

# Declarative Processes



- Instead of describing exactly how a business process should be executed, declarative processes
  - describe the activities to be executed and
  - constraints prohibiting undesired behavior (e.g., selection constraints, ordering constraints, resource constraints)




[PSSA07]

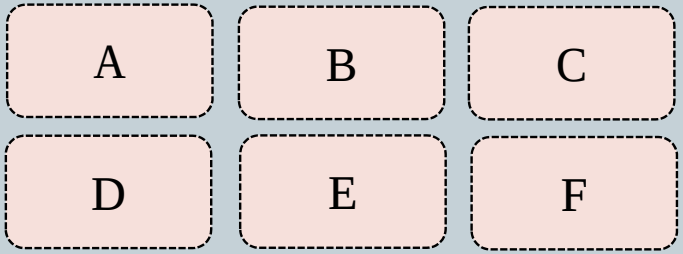
© B. Weber and M. Reichert, 2012

# Modeling Declarative Processes

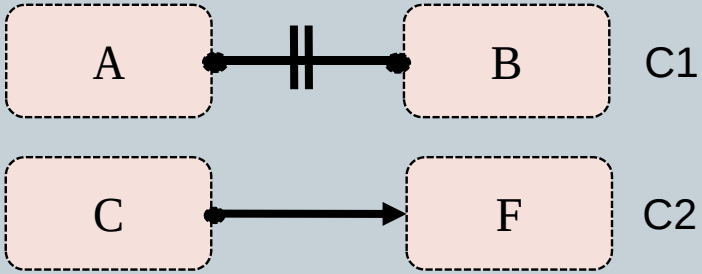


Declarative Process Model S

Activities A




Constraints C




Legend


NOT CO-EXISTENCE
A and B are mutually exclusive


RESPONSE
If A is executed, B needs to executed afterwards

**Execution trace producible on S:**

$\sigma_1$ = < A, A, D, E, A>
$\sigma_2$ = < B, C, F, E, B>
$\sigma_3$ = < B, E, F>

**Execution trace not producible on S:**

$\sigma_4$ = < A, C, E, A> C2 ×
$\sigma_5$ = < B, D, C> C2 ×
$\sigma_6$ = < A, D, B, F, E> C1 ×

© B. Weber and M. Reichert, 2012

# Modeling Declarative Processes



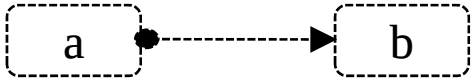
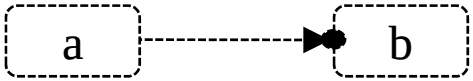
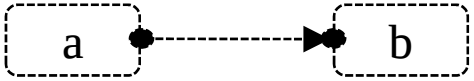
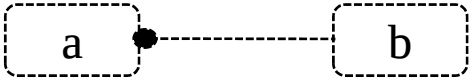
- 5 Major Categories
  - Selection Constraints
  - Relation Constraints
  - Branching Constraints
  - Negation Constraints
  - Choice Constraints

© B. Weber and M. Reichert, 2012

EXAMPLE: SELECTION CONSTRAINTS

| | | |
|---|---|---|
| `existence(a, n)` | n..*<br>a | Activity *a* must occur at least *n* times in every trace |
| | Example: `existence(A,1)`<br>Supported traces, e.g.: <A>, <A, A, A><br>Unsupported trace, e.g.: <> | |
| `at_most(a, n)` | 0..n<br>a | Activity *a* must occur at most *n* times in every trace |
| | Example: `at_most(A,3)`<br>Supported traces, e.g.: <>, <A>, <A, A>, <A, A, A><br>Unsupported trace, e.g.: <A, A, A, A> | |
| `exactly(a, n)` | n<br>a | Activity *a* must occur exactly *n* times in every trace |
| | Example: `exactly(A,2)`<br>Supported trace, e.g.: <A, A><br>Unsupported traces, e.g.,: <A>, <A, A, A> | |
| `init(a)` | init<br>a | Activity *a* must be the first executed activity in every trace |
| | Example: `init(A)`<br>Supported trace, e.g.: <A, C, D, B><br>Unsupported trace, e.g.: <D, C, B, A> | |

## EXAMPLE: RELATION CONSTRAINTS

| | | |
|---|---|---|
| `response(a, b)` |  | If *a* is executed, *b* needs to be executed afterwards (but not necessarily directly after) |
| | Example: `response(A,B)`<br>Supported traces, e.g.: <A,B>, <A,A,A,B>, <B><br>Unsupported trace, e.g.: <A> | |
| `precedence(a, b)` |  | Activity *b* needs to be preceded by activity *a* |
| | Example: `precedence(A,B)`<br>Supported traces, e.g.: <A,B>, <A,B,B,B>, <A><br>Unsupported trace, e.g.: <B> | |
| `succession(a, b)` |  | If *a* is executed, *b* needs to be executed afterwards (but not necessarily directly after); activity *b* needs to be preceded by activity *a* |
| | Example: `succession(A,B)`<br>Supported traces, e.g.: <A,B>, <A,A,A,B>, <A,B,B,B><br>Unsupported traces, e.g.: <A>, <B> | |
| `respondedExistence(a, b)` |  | If activity *a* is executed, activity *b* needs to be executed either before or after *a* |
| | Example: `respondedExistence(A,B)`<br>Supported traces, e.g.: <A,B>, <B,A>, <A,B,A>, <B><br>Unsupported trace, e.g.: <A> | |

# Executing Declarative Processes



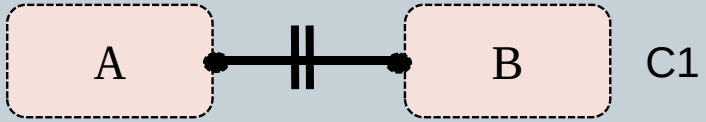
Declarative Process Model S

Activities A

A B C

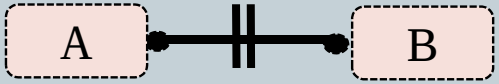
D E F

Constraints C




C1


C2




NOT CO-EXISTENCE
A and B are mutually exclusive

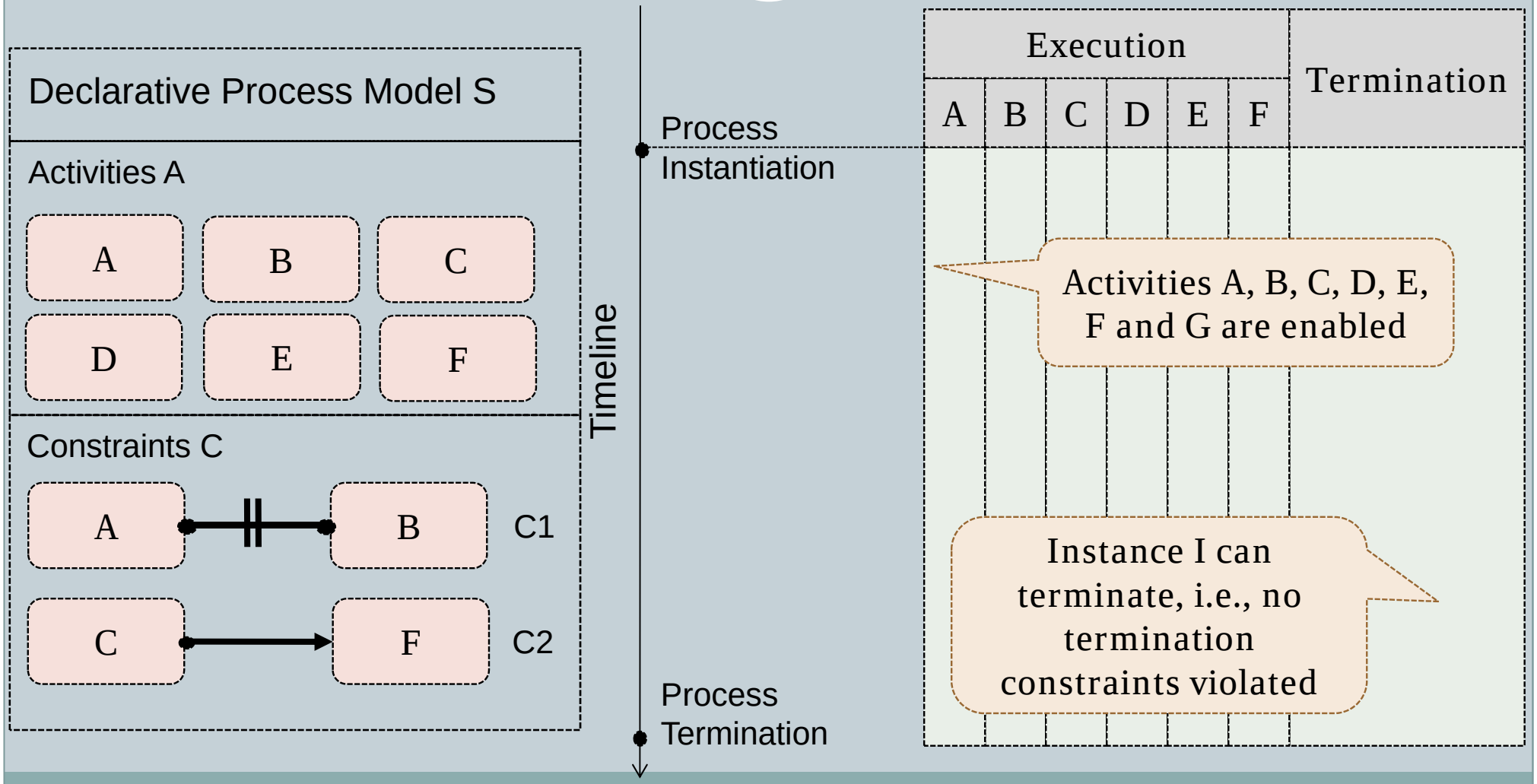

RESPONSE
If A is executed, B needs to executed afterwards

| Partial Trace | Set of Enabled Activities |
|---|---|
| < > | {A, B, C, D, E, F} |
| <A> | {A, C, D, E, F}<br>B is not included since partial trace <A, B> violates constraint C1 |
| <A, C> | {A, C, D, E, F}<br>B is not included since partial trace <A, B> violates constraint C1 |

© B. Weber and M. Reichert, 2012

# Executing Declarative Processes

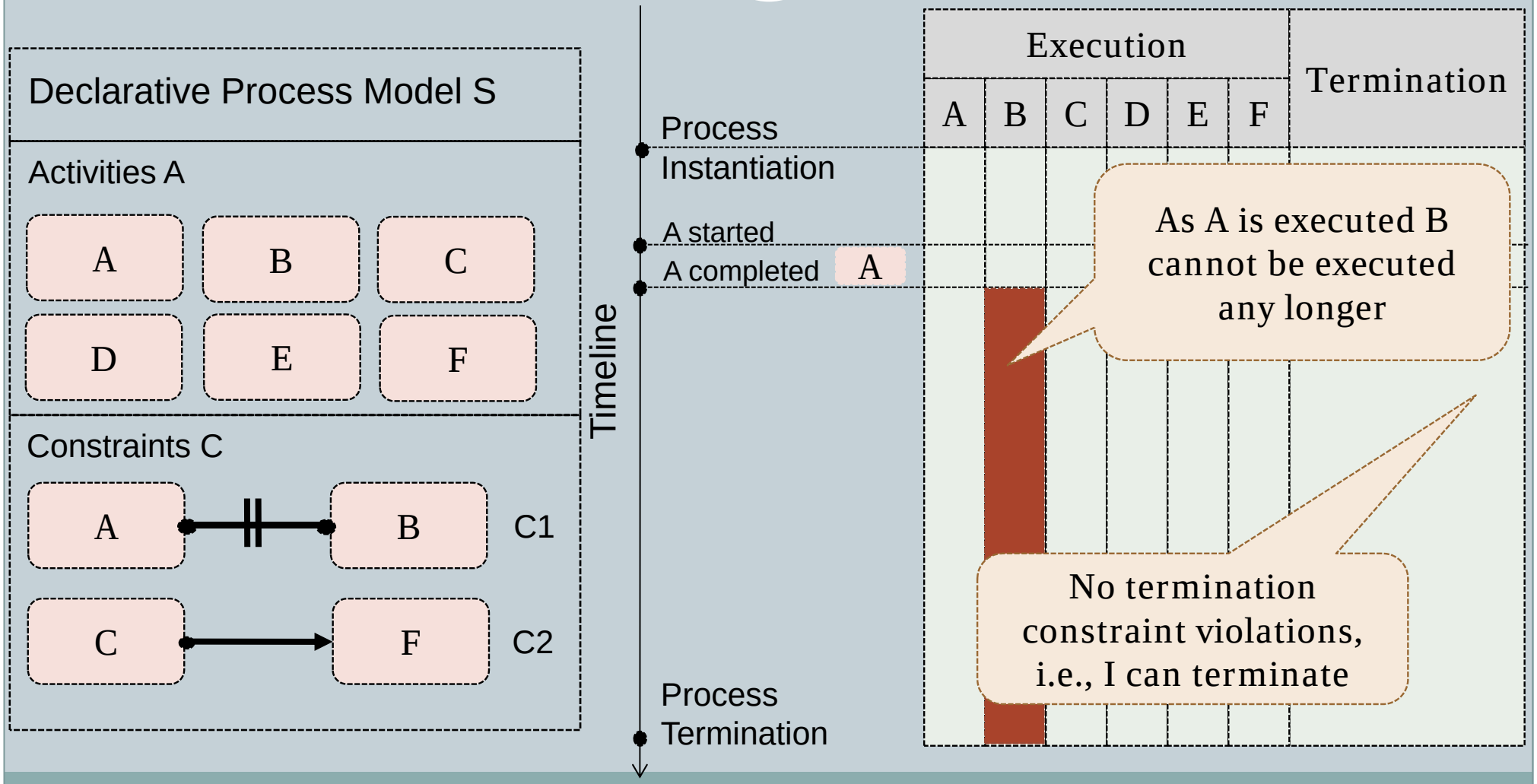





© B. Weber and M. Reichert, 2012

# Executing Declarative Processes

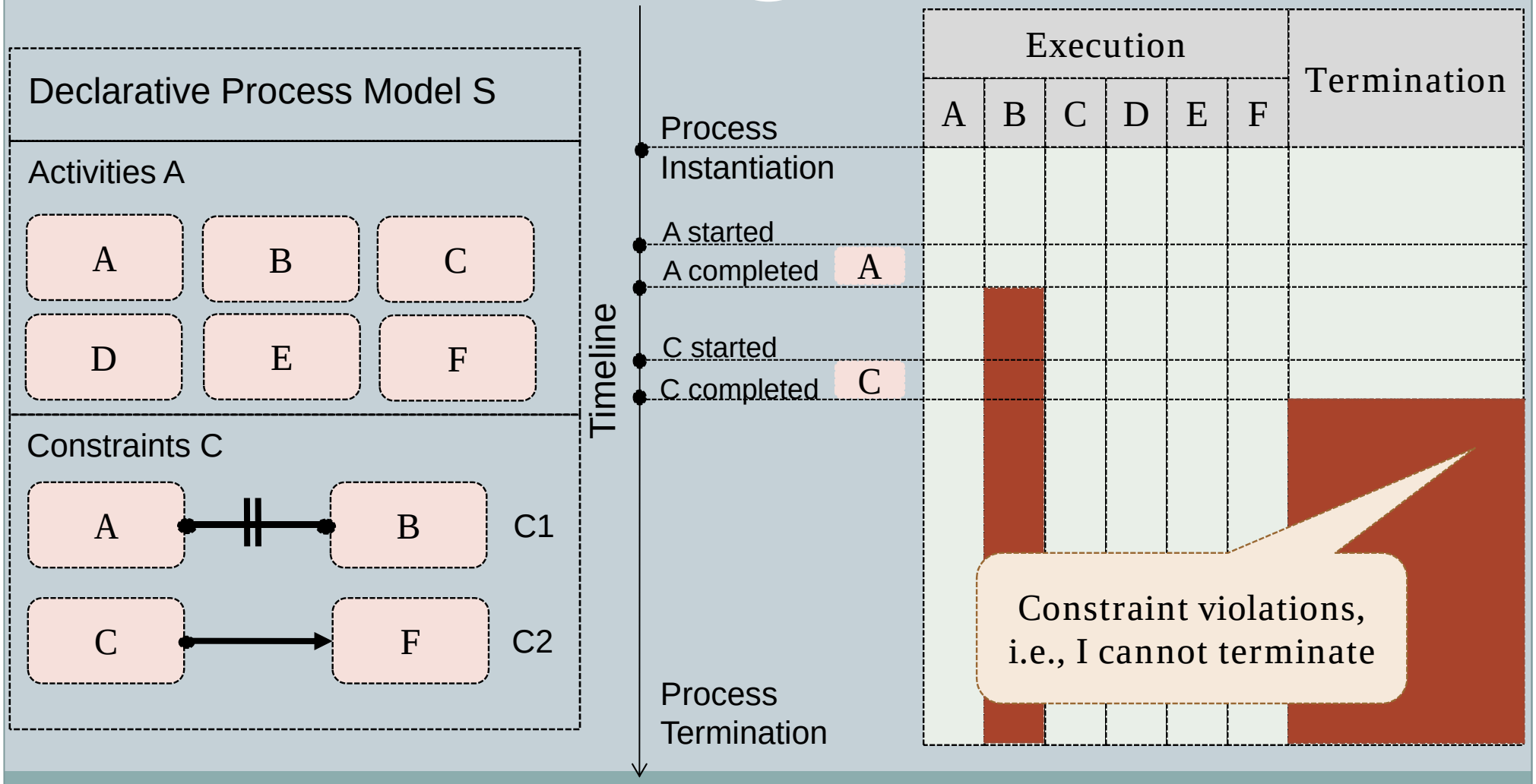





© B. Weber and M. Reichert, 2012

# Executing Declarative Processes

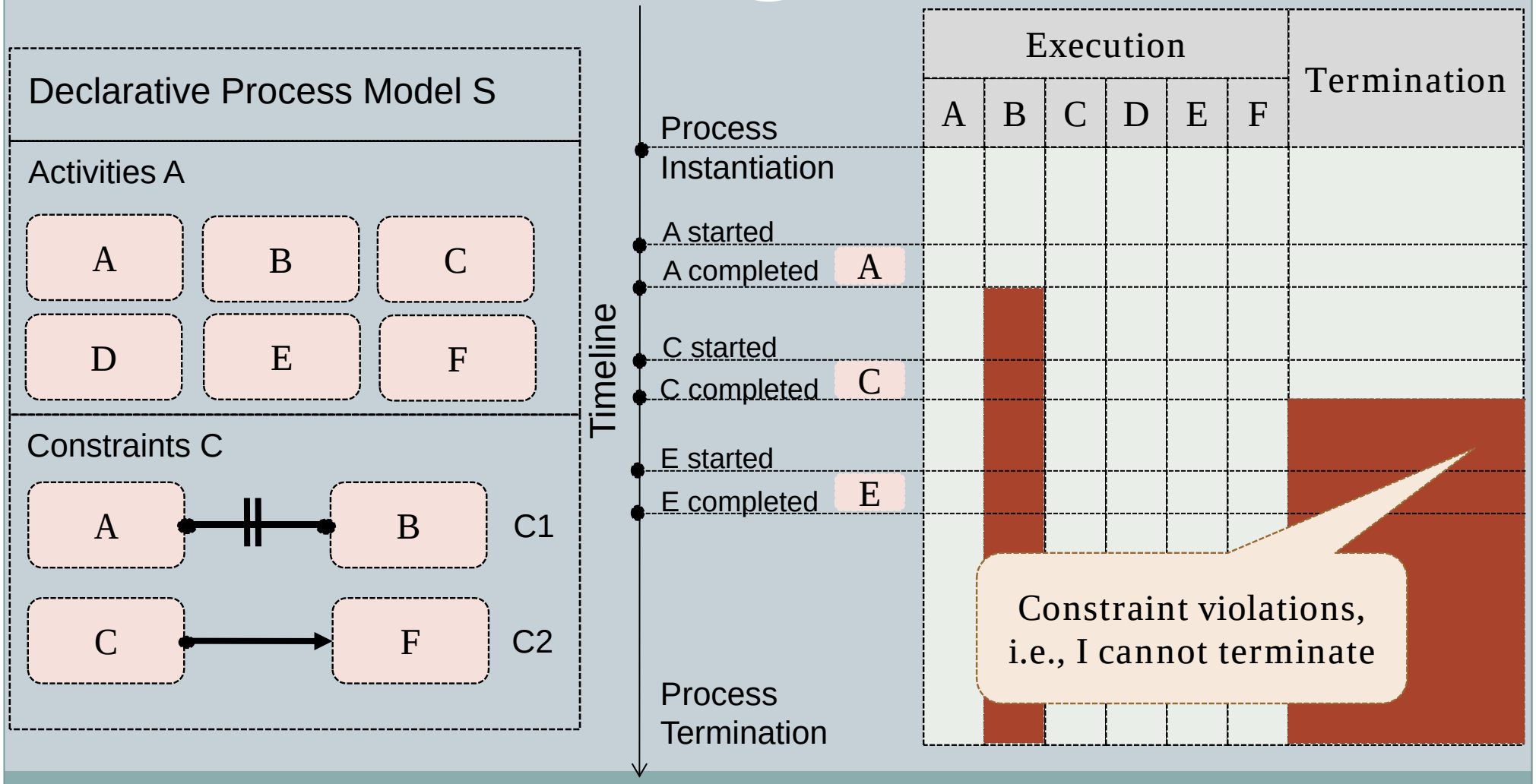





© B. Weber and M. Reichert, 2012

# Executing Declarative Processes

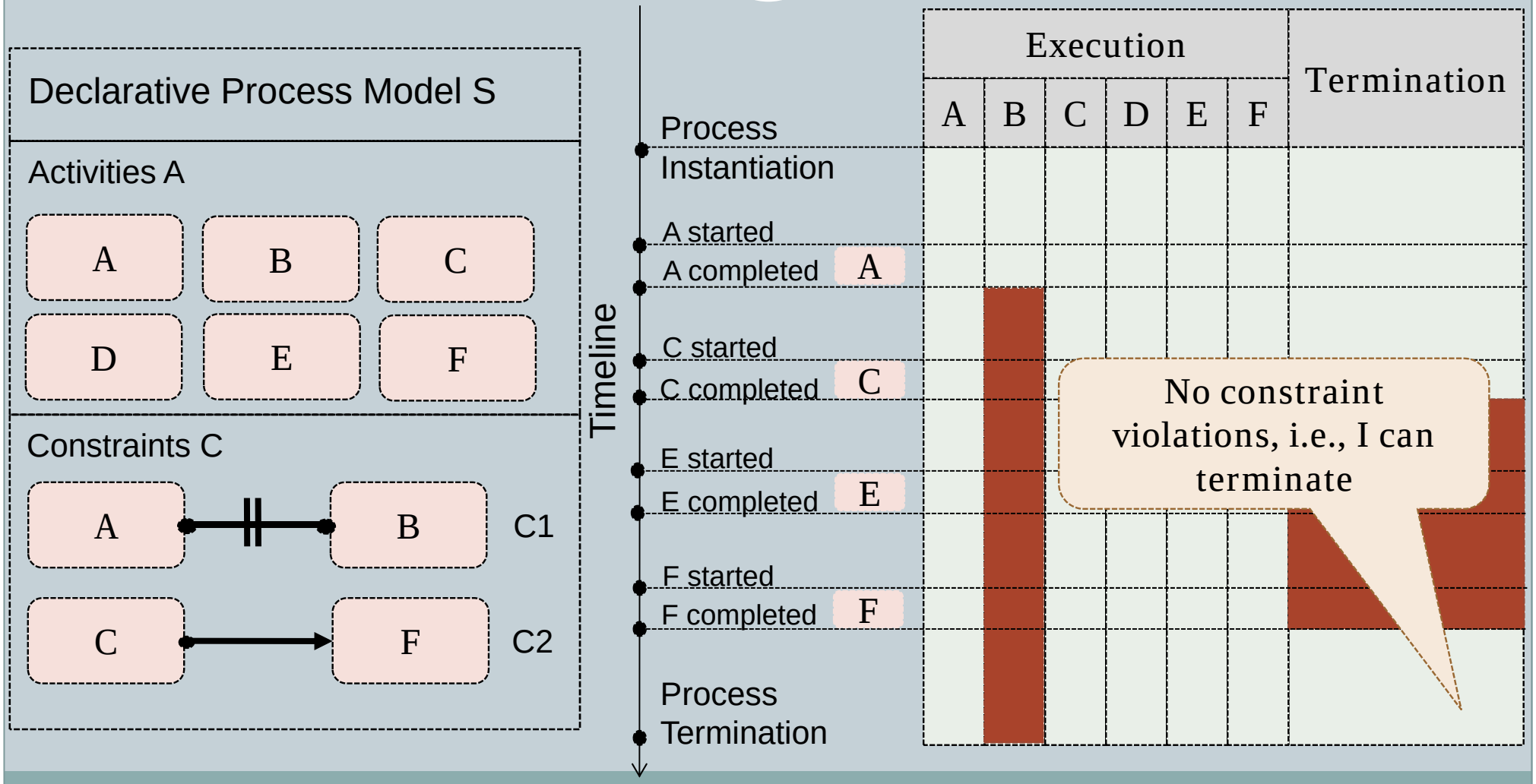





© B. Weber and M. Reichert, 2012

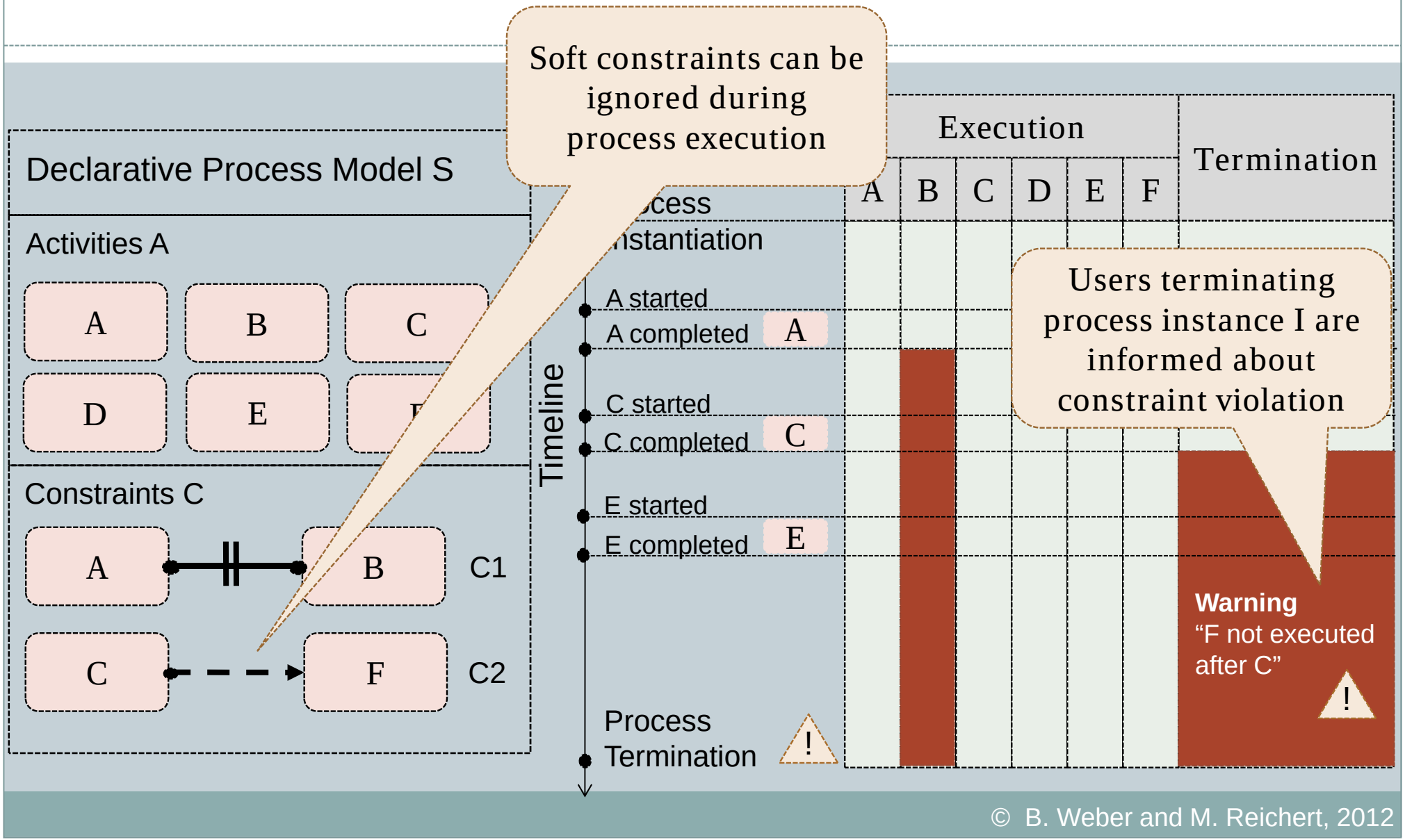
# Executing Declarative Processes






© B. Weber and M. Reichert, 2012

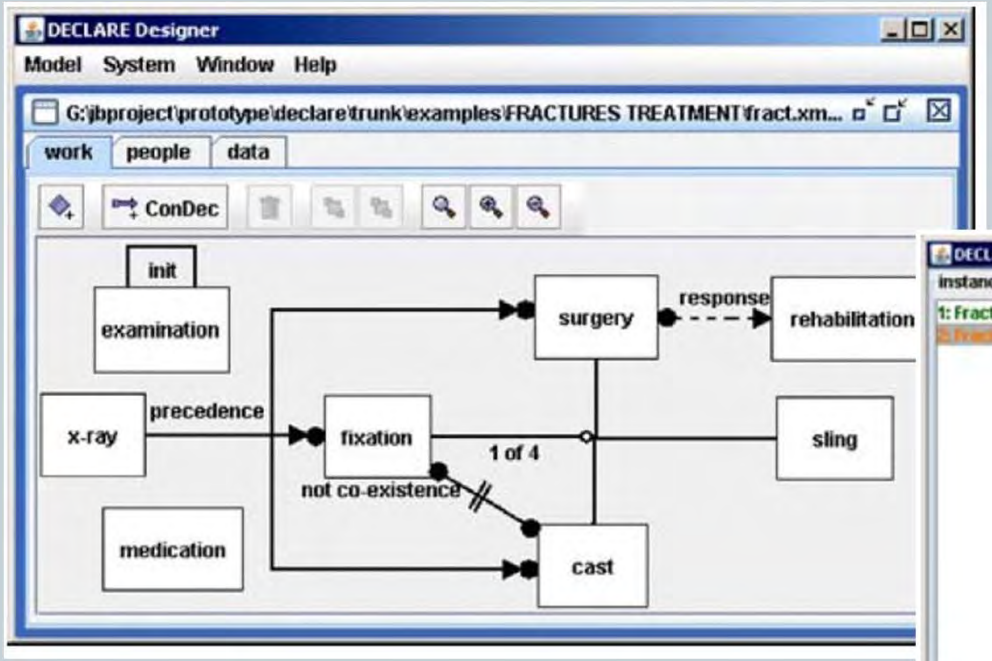
# Overriding Constraints



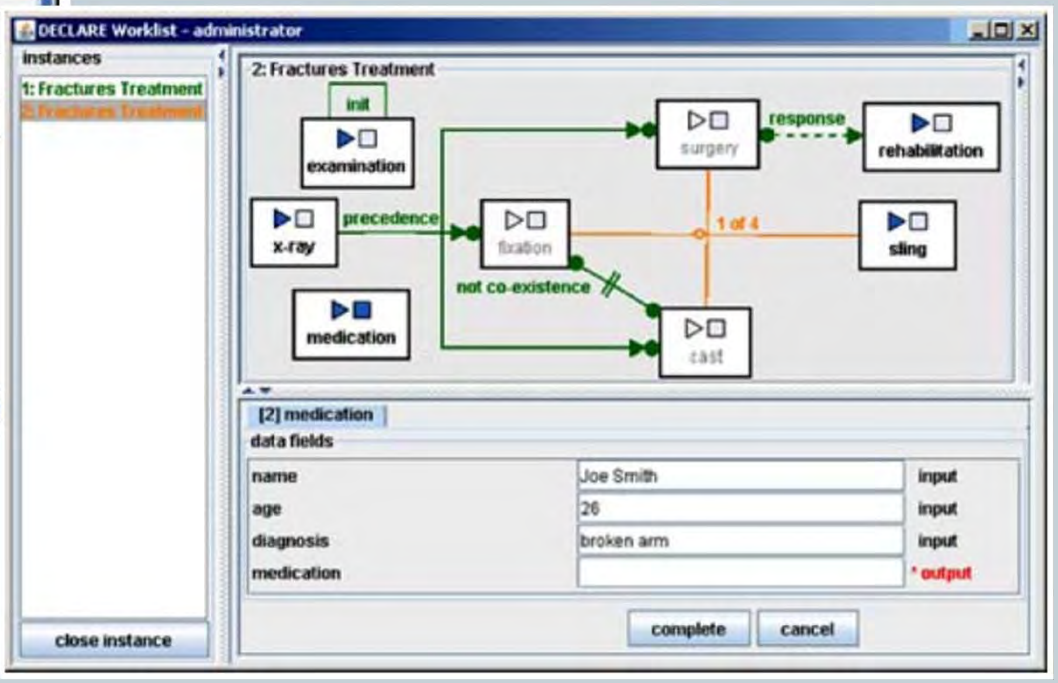

© B. Weber and M. Reichert, 2012

# The Declare System






Composing Declarative Processes with Declare



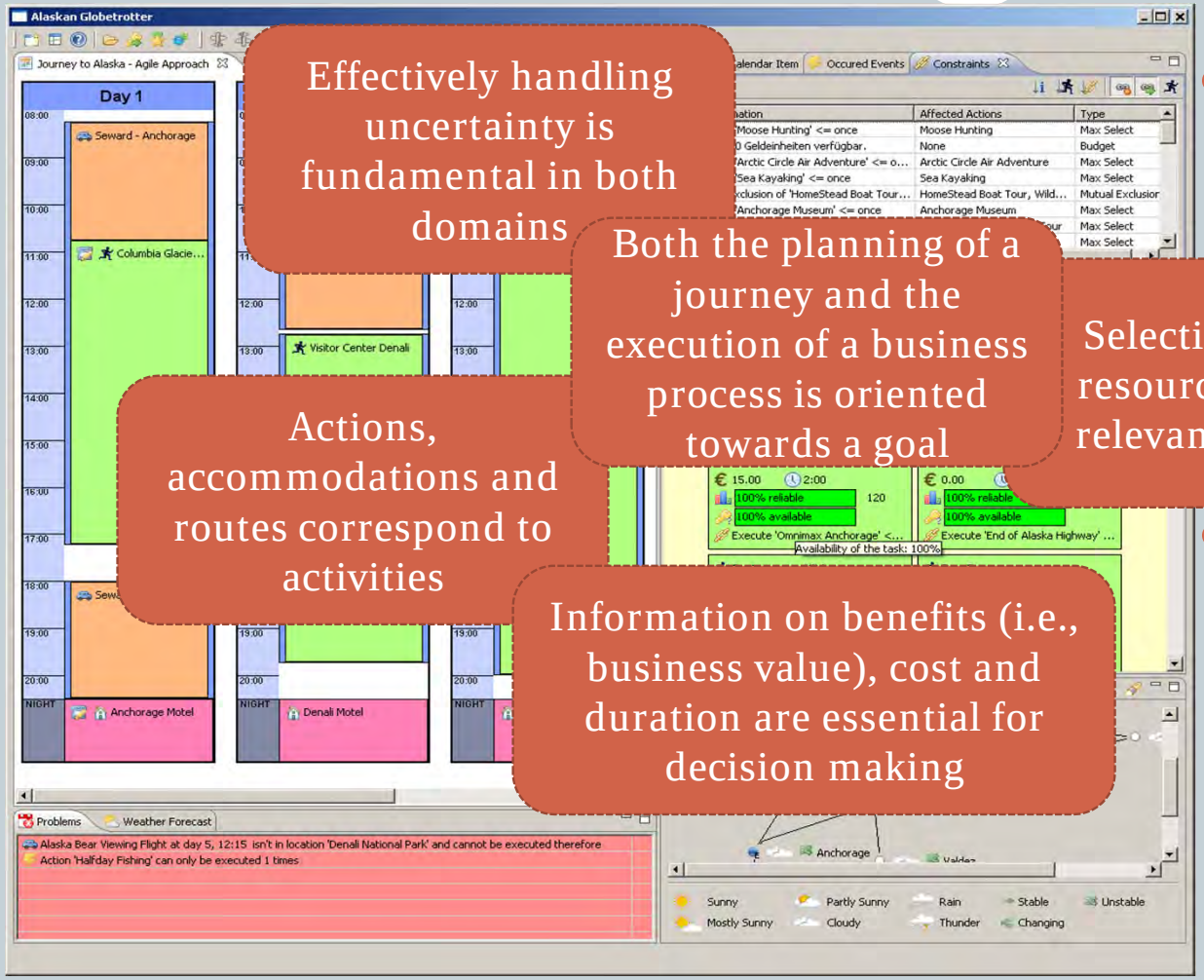

Executing Declarative Processes with Declare

van der Aalst, Pesic and Schonenberg 2009 [APS09]

© B. Weber and M. Reichert, 2012

# The Alaska Simulator






Effectively handling uncertainty is fundamental in both domains

Both the planning of a journey and the execution of a business process is oriented towards a goal

Selection, ordering and resource constraints are relevant in both settings

Actions, accommodations and routes correspond to activities

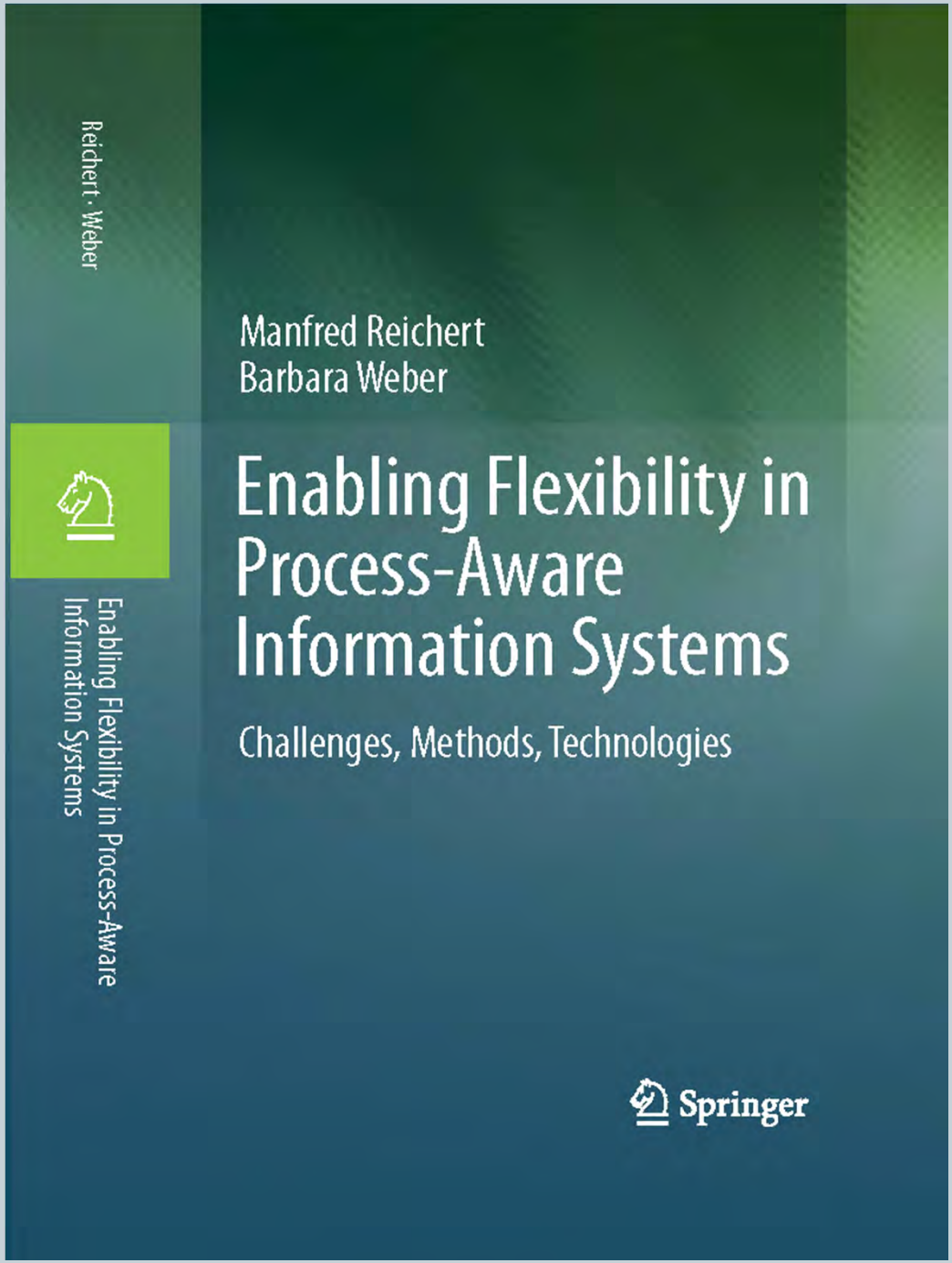
Information on benefits (i.e., business value), cost and duration are essential for decision making

- Is an interactive planning tool providing support [illegible]nposition
- Uses journey as a metaphor for business processes

http:\\alaskasimulator.org

Weber, Zugal, Pinggera and Wild 2009 [WZP+09]

© B. Weber and M. Reichert, 2012


Reichert · Weber
Manfred Reichert
Barbara Weber
Enabling Flexibility in Process-Aware Information Systems
Challenges, Methods, Technologies
Springer
Enabling Flexibility in Process-Aware Information Systems

# References (Flexibility-by-Design)



[AHK+03] W.M.P van der Aalst, A.H.M. ter Hofstede, B. Kiepuszewski, and A.P. Barros. **Workflow Patterns.** *Distributed and Parallel Databases*, 14(3), pages 5-51, July 2003.

[RHA+06] N. Russell, A.H.M. ter Hofstede, W.M.P. van der Aalst, and N. Mulyar. **Workflow Control-Flow Patterns: A Revised View**. *BPM Center Report BPM-06-22* , BPMcenter.org, 2006.

[TAS09] Nikola Trcka, Wil M. P. van der Aalst, Natalia Sidorova: **Data-Flow Anti-patterns: Discovering Data-Flow Errors in Workflows.** CAiSE 2009: 425-439

[VVK09] Jussi Vanhatalo, Hagen Völzer, Jana Koehler: **The refined process structure tree.** Data Knowl. Eng. 68(9): 793-818 (2009)

[Wes07] Mathias Weske: **Business Process Management: Concepts, Languages, Architectures,** Springer 2007.

[WVA+09] Wynn, M.T., Verbeek, H.M.W., Aalst, W.M.P. van der, Hofstede, A.H.M. ter and Edmond, D. (2009). **Business process verification : finally a reality!** *Business Process Management Journal*, *15*(1), 74-92.

© B. Weber and M. Reichert, 2012

# References (Configurable Process Models)



[GAV+08] Florian Gottschalk, Wil M. P. van der Aalst, Monique H. Jansen-Vullers, Marcello La Rosa: **Configurable Workflow Models**. Int. J. Cooperative Inf. Syst. 17(2): 177-221 (2008)

[HBR10] Alena Hallerbach, Thomas Bauer, Manfred Reichert: **Capturing variability in business process models: the Provop approach.** Journal of Software Maintenance 22(6-7): 519-546 (2010)

[HBR09] Alena Hallerbach, Thomas Bauer, Manfred Reichert: **Guaranteeing Soundness of Configurable Process Variants in Provop.** CEC 2009: 98-105

[HBR08] Alena Hallerbach, Thomas Bauer, Manfred Reichert: **Managing Process Variants in the Process Life Cycle.** ICEIS (3-2) 2008: 154-161

[Ros09] M. La Rosa: **Managing Variability in Process-Aware Information Systems,** *PhD Thesis*, Queensland University of Technology, Brisbane, Australia. April 2009.

[RDH09] M. La Rosa, M. Dumas, A.H.M. ter Hofstede: **Modelling Business Process Variability for Design-Time Configuration.** In J. Cardoso, W.M.P. van der Aalst (editors), *Handbook of Research on Business Process Modeling*, IDEA Group – Information Science Reference, 2009.

[RLS+07] M. La Rosa, J. Lux, S. Seidel, M. Dumas and A.H.M. ter Hofstede: **Questionnaire-driven Configuration of Reference Process Models.** In *Proc. CAiSE 2007*, Trondheim, Norway. LNCS Vol. 4495, pp. 424–438, Springer, 2007.

[RoAa07] Michael Rosemann, Wil M. P. van der Aalst: **A configurable reference modelling language**. Inf. Syst. 32 (1): 1-23 (2007)

© B. Weber and M. Reichert, 2012

# References (Exception Handling)



[AHA+07] Michael Adams, Arthur H. M. ter Hofstede, Wil M. P. van der Aalst, David Edmond: **Dynamic, Extensible and Context-Aware Exception Handling for Workflows.** OTM Conferences (1) 2007: 95-112

[LCO+10]Barbara Staudt Lerner, Stefan Christov, Leon J. Osterweil, Reda Bendraou, Udo Kannengiesser, Alexander E. Wise: **Exception Handling Patterns for Process Modeling.** IEEE Trans. Software Eng. 36(2): 162-183 (2010)

[MoSa87] Hector Garcia-Molina, Kenneth Salem: Sagas. SIGMOD Conference 1987: 249-259

[NAH06] N. Russell, W.M.P. van der Aalst, and A.H.M. ter Hofstede. **Exception Handling Patterns in Process-Aware Information Systems.** BPM Center Report BPM-06-04 , BPMcenter.org, 2006.

[RHE+04] N. Russell, A.H.M. ter Hofstede, D. Edmond, and W.M.P. van der Aalst. **Workflow Resource Patterns.** BETA Working Paper Series, WP 127, Eindhoven University of Technology, Eindhoven, 2004.

© B. Weber and M. Reichert, 2012

# References (Handling Unforeseen Exceptions)



[ReDa98] Manfred Reichert, Peter Dadam: **ADEPT$_{flex}$-Supporting Dynamic Changes of Workflows Without Losing Control**. J. Intell. Inf. Syst. 10(2): 93-129 (1998)

[RRD04a] Stefanie Rinderle, Manfred Reichert, Peter Dadam: **Correctness criteria for dynamic changes in workflow systems - a survey**. Data Knowl. Eng. 50(1): 9-34 (2004)

[RRD04b] Stefanie Rinderle, Manfred Reichert, Peter Dadam: **Flexible Support of Team Processes by Adaptive Workflow Systems.** Distributed and Parallel Databases 16(1): 91-116 (2004)

[RRW08] Stefanie Rinderle-Ma, Manfred Reichert, Barbara Weber: **Relaxed Compliance Notions in Adaptive Process Management Systems.** ER 2008: 232-247

[RWR+05] Stefanie Rinderle, Barbara Weber, Manfred Reichert, Werner Wild: Integrating Process Learning and Process Evolution - A Semantics Based Approach. Business Process Management 2005: 252-267

[WRR05] Barbara Weber, Stefanie Rinderle, Werner Wild, Manfred Reichert: **CCBR-Driven Business Process Evolution.** ICCBR 2005: 610-624

[WRR08] Barbara Weber, Manfred Reichert, Stefanie Rinderle-Ma: **Change patterns and change support features - Enhancing flexibility in process-aware information systems.** Data Knowl. Eng. 66(3): 438-466 (2008)

[WRW+09] Barbara Weber, Manfred Reichert, Werner Wild and Stefanie Rinderle-Ma: **Providing Integrated Life Cycle Support in Process-Aware Information Systems.** In: Int J Coop Inf Sys 18 (2009) 1, pp. 115-165.

[WRW06] Barbara Weber, Manfred Reichert, Werner Wild: **Case-Base Maintenance for CCBR-Based Process Evolution.** ECCBR 2006: 106-120

[WWB04] Barbara Weber, Werner Wild, Ruth Breu: CBRFlow: **Enabling Adaptive Workflow Management Through Conversational Case-Based Reasoning**. ECCBR 2004: 434-448

© B. Weber and M. Reichert, 2012

# References (Process and Variant Mining)



[ABD07] Wil M. P. van der Aalst, H. T. de Beer, Boudewijn F. van Dongen: **Process Mining and Verification of Properties: An Approach Based on Temporal Logic**. OTM Conferences (1) 2005: 130-147

[ADH+03] W.M.P. van der Aalst, B.F. van Dongen, J. Herbst, L. Maruster, G. Schimm, and A.J.M.M. Weijters. **Workflow Mining: A Survey of Issues and Approaches.** Data and Knowledge Engineering , 47(2):237-267, 2003.

[ARS05] Wil M. P. van der Aalst, Hajo A. Reijers, Minseok Song: **Discovering Social Networks from Event Logs.** Computer Supported Cooperative Work 14(6): 549-593 (2005)

[ARV+10] Wil M. P. van der Aalst, Vladimir Rubin, H. M. W. Verbeek, Boudewijn F. van Dongen, Ekkart Kindler, Christian W. Günther: **Process mining: a two-step approach to balance between underfitting and overfitting**. Software and System Modeling 9(1): 87-111 (2010)

[AWM04] W.M.P. van der Aalst, A.J.M.M. Weijters, and L. Maruster. **Workflow Mining: Discovering Process Models from Event Logs.** IEEE Transactions on Knowledge and Data Engineering 16(9):1128-1142, 2004.

© B. Weber and M. Reichert, 2012

# References (Process and Variant Mining)



[GGM+07] Gianluigi Greco, Antonella Guzzo, Giuseppe Manco, Domenico Saccà: **Mining unconnected patterns in workflows.** Inf. Syst. 32(5): 685-712 (2007)

[GGP06] Gianluigi Greco, Antonella Guzzo, Luigi Pontieri, Domenico Saccà: **Discovering Expressive Process Models by Clustering Log Traces**. IEEE Trans. Knowl. Data Eng. 18(8): 1010-1027 (2006)

[HeKa04] Joachim Herbst and Dimitris Karagiannis: **Workflow Mining with InWoLvE.** Computers and Industry 53(3): 245-264 (2004).

[LRW10] Chen Li, Manfred Reichert, Andreas Wombacher: **The Minadept Clustering Approach for Discovering Reference Process Models Out of Process Variants.** Int. J. Cooperative Inf. Syst. 19(3-4): 159-203 (2010)

[MWA07] Ana Karla A. de Medeiros, A. J. M. M. Weijters, Wil M. P. van der Aalst: **Genetic process mining: an experimental evaluation**. Data Min. Knowl. Discov. 14(2): 245-304 (2007)

[PiGo04] S. Pinter and M. Golani: **Discovering workflow models from activities' lifespans.** Computers and Industry 53(3): 283-296 (2004).

[RoAa08] Anne Rozinat, Wil M. P. van der Aalst: **Conformance checking of processes based on monitoring real behavior.** Inf. Syst. 33(1): 64-95 (2008)

[Sch04] Guido Schimm: **Mining exact models of concurrent workflows.** Computers and Industry 53(3): 265-281 (2004).

[WeAa03] A.J.M.M. Weijters and W.M.P. van der Aalst. **Rediscovering Workflow Models from Event-Based Data using Little Thumb**. Integrated Computer-Aided Engineering, 10(2):151-162, 2003.

[WWA+10] Lijie Wen, Jianmin Wang, Wil M. P. van der Aalst, Biqing Huang, Jiaguang Sun: **Mining process models with prime invisible tasks**. Data Knowl. Eng. 69(10): 999-1021 (2010)

© B. Weber and M. Reichert, 2012

# Referencesn (Process Evolution)



[RRD04a] Stefanie Rinderle, Manfred Reichert, Peter Dadam: **Correctness criteria for dynamic changes in workflow systems - a survey**. Data Knowl. Eng. 50(1): 9-34 (2004)

[WeRe08] B. Weber and M. Reichert: **Refactoring Process Models in Large Process Repositories** In Proc. CAiSE'08 (2008), pp. 124-139

[WRR+11] B. Weber and M. Reichert and H. Reijers and J. Mendling: **Refactoring Large Process Model Repositories** Computers and Industry 62(2011) 5, pp. 467-486.

[WRW+09] Weber B., Reichert M., Wild W. and Rinderle-Ma S.: **Providing Integrated Life Cycle Support in Process-Aware Information Systems.** In: International Journal of Cooperative Information Systems 18 (2009) 1, pp. 115-165.

© B. Weber and M. Reichert, 2012

# References (Loosely-specified Processes)



[AaPe06] van der Aalst, W., Pesic, M.: **DecSerFlow: Towards a Truly Declarative Service Flow Language**. Tech. Rep., BPMcenter.org (2006)

[AHE+06] Adams, M., ter Hofstede, A., Edmond, D., van der Aalst, W.: **A Service-Oriented Implementation of Dynamic Flexibility in Workflows**. In: Proc. Coopis'06 (2006)

[AVM+04] R. Aggarwal, Kunal Vernal, John Miller and William Milnor : **Constraint-driven Web Service Composition in METEOR-S**: In Proc. SCC'04, pp. 23-30.

[CaSa01] Fabio Casati, Ming-Chien Shan: **Dynamic and adaptive composition of e-services.** Inf. Syst. 26(3): 143-163 (2001)

[CPE+08] Gerardo Canfora, Massimiliano Di Penta, Raffaele Esposito, Maria Luisa Villani: **A framework for QoS-aware binding and re-binding of composite web services**. Journal of Systems and Software 81(10): 1754-1769 (2008)

van Elst; Andreas Lauer; Heiko Maus; Sven Schwarz; Michael Sintek, A.A.A.B.L.:
**Frodo: A framework for distributed organizational memories. milestone 1: Requirements analysis and system architecture.** Dfki document (2001). URL
http://www.dfki.unikl.de/dfkidok/publications/D/01/01/abstract.html

[Kli00] Justus Klingemann: **Controlled Flexibility in Workflow Management.** CAiSE 2000: 126-141

© B. Weber and M. Reichert, 2012

# References (Loosely-specified Processes)



[PSS+07] Pesic, M., Schonenberg, M., Sidorova, N., van der Aalst, W.: **Constraint-Based Workflow Models: Change Made Easy**. In: Proc. CoopIS'07, pp. 77–94 (2007)

[RHA+06] N. Russell, A.H.M. ter Hofstede, W.M.P. van der Aalst, and N. Mulyar. W**orkflow Control-Flow Patterns: A Revised View.** BPM Center Report BPM-06-22 , BPMcenter.org, 2006.

[SSO01] Sadiq, S., Sadiq, W., Orlowska, M.: **Pockets of flexibility in workflow specifications.** In: Proc. ER'01, pp. 513–526 (2001)

[SSO05] Sadiq, S., Sadiq, W., Orlowska, M.: **A Framework for Constraint Specification and Validation in Flexible Workflows.** Information Systems 30(5), 349 – 378 (2005)

[SuWe03] Hilmar Schuschel, Mathias Weske: **Integrated Workflow Planning and Coordination.** DEXA 2003: 771-781

[ZNB+08] Liangzhao Zeng, Anne H. H. Ngu, Boualem Benatallah, Rodion M. Podorozhny, Hui Lei: **Dynamic composition and optimization of Web services.** Distributed and Parallel Databases 24(1-3): 45-72 (2008)

© B. Weber and M. Reichert, 2012

# References (Declarative Workflows)



[AaPe06] W.M.P. van der Aalst and M. Pesic : **DecSerFlow: Towards a Truly Declarative Service Flow Language.** In WS-FM 2006, 2006, pp 1-23.

[APS09] Wil M. P. van der Aalst, Maja Pesic, Helen Schonenberg: **Declarative workflows: Balancing between flexibility and support.** Computer Science - R&D 23(2): 99-113 (2009)

[PSS+07] Pesic, M., Schonenberg, M., Sidorova, N., van der Aalst, W.: **Constraint-Based Workflow Models: Change Made Easy**. In: Proc. CoopIS'07, pp. 77–94 (2007)

[SWD+08] Helen Schonenberg, Barbara Weber, Boudewijn F. van Dongen, Wil M. P. van der Aalst**: Supporting Flexible Processes through Recommendations Based on History**. BPM 2008: 51-66

[WRR08] Barbara Weber, Manfred Reichert, Stefanie Rinderle-Ma: **Change patterns and change support features - Enhancing flexibility in process-aware information systems.** Data Knowl. Eng. 66(3): 438-466 (2008)

[WBB04] Jacques Wainer, Fábio de Lima Bezerra, Paulo Barthelmess: **Tucupi: a flexible workflow system based on overridable constraints.** SAC 2004: 498-502

[WRZ+09] Barbara Weber, Hajo A. Reijers, Stefan Zugal, Werner Wild: **The Declarative Approach to Business Process Execution: An Empirical Test**. CAiSE 2009: 470-485

© B. Weber and M. Reichert, 2012