

Enabling Flexibility in Process-aware Information Systems

Challenges, Methods, Technologies

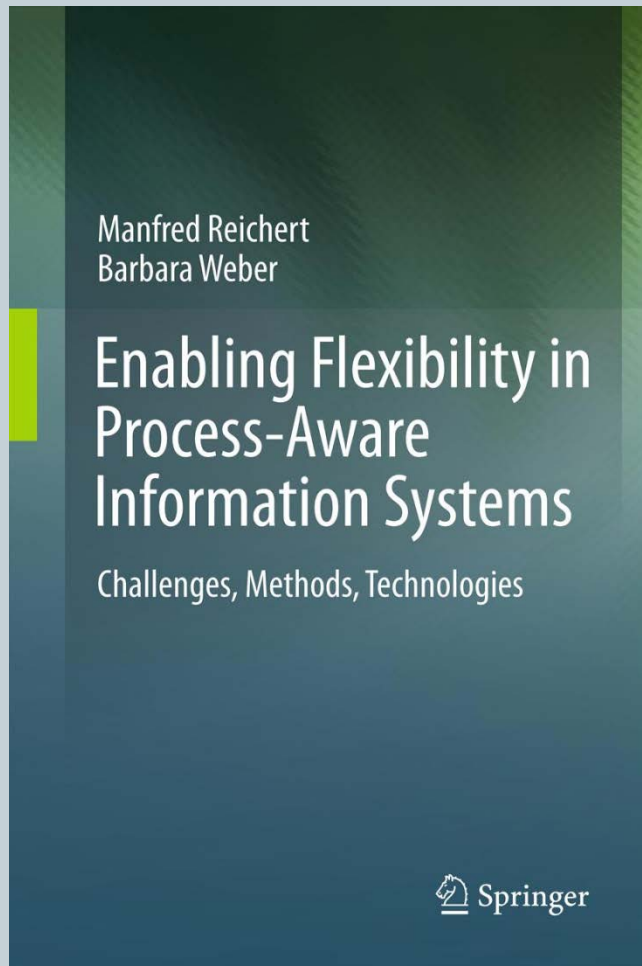


MONTEVIDEO, DECEMBER 11TH 2012



**PRESENTED BY
BARBARA WEBER
UNIV. OF INNSBRUCK**

Content



- **Keynote based on new Springer book *Enabling Flexibility in Process-Aware Information Systems***

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
 - Handling of anticipated exceptions
 - Handling unforeseen exceptions with Ad-hoc Changes
 - Process Evolution
 - Process Monitoring, Mining & Analysis
 - Business Process Compliance

Content



- Part 4– Loosely-specified Process Models
 - Concretizing Loosely-specified process models
 - Constraint-based process models
 - User and Data Driven Processes
 - A Framework for Object-Aware Processes

*Not part of this keynote
For details see Chapters
11-14 of the book*

Business Processes and Workflows

Part 1 - Process-aware Information Systems



MONTEVIDEO, DECEMBER 11TH 2012



PRESENTED BY
BARBARA WEBER
UNIV. OF INNSBRUCK

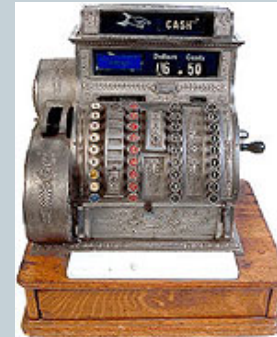
A Retail Process



Welcome customer



Offer Clothes

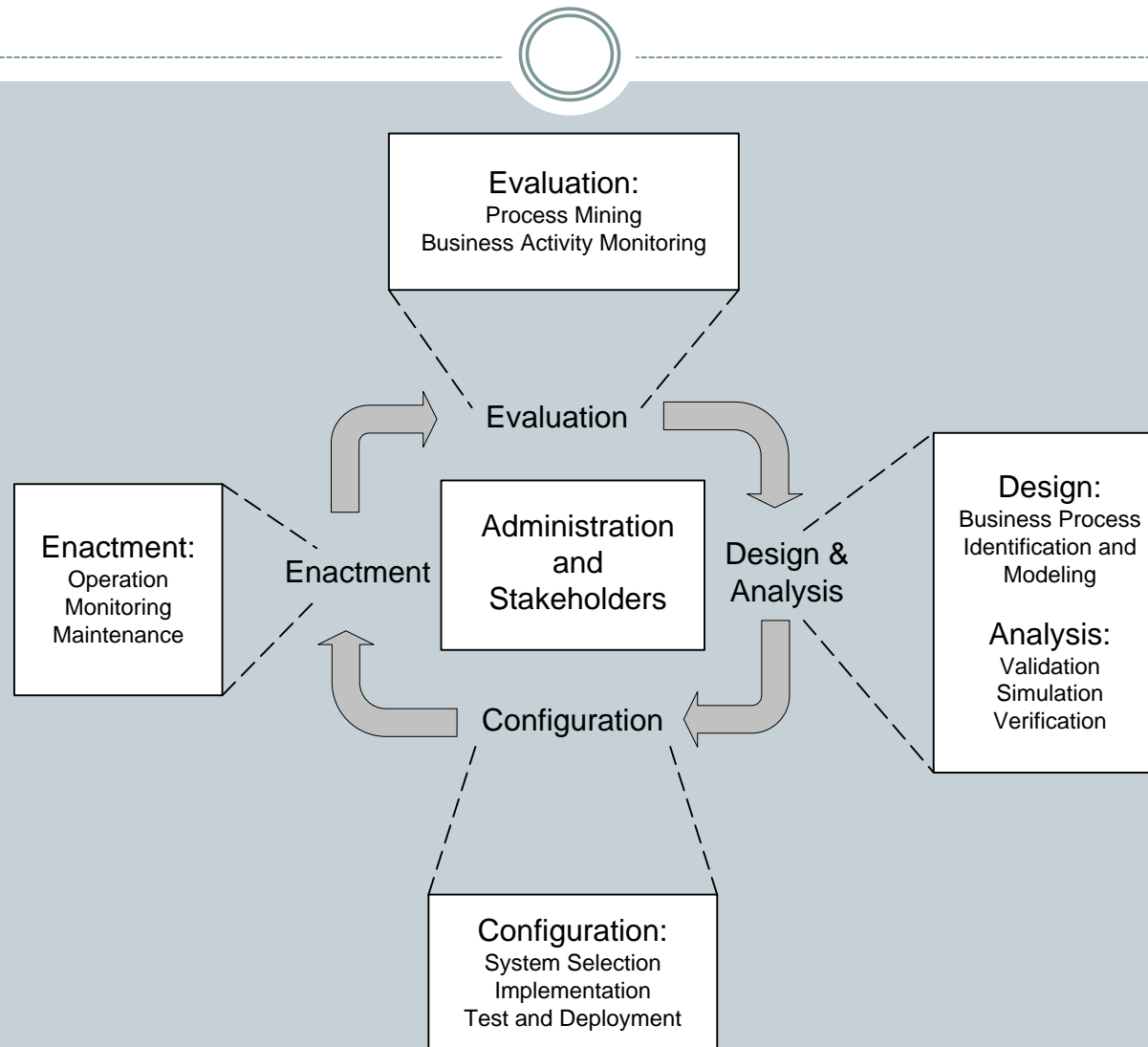


Bill Clothes



Hand over clothes

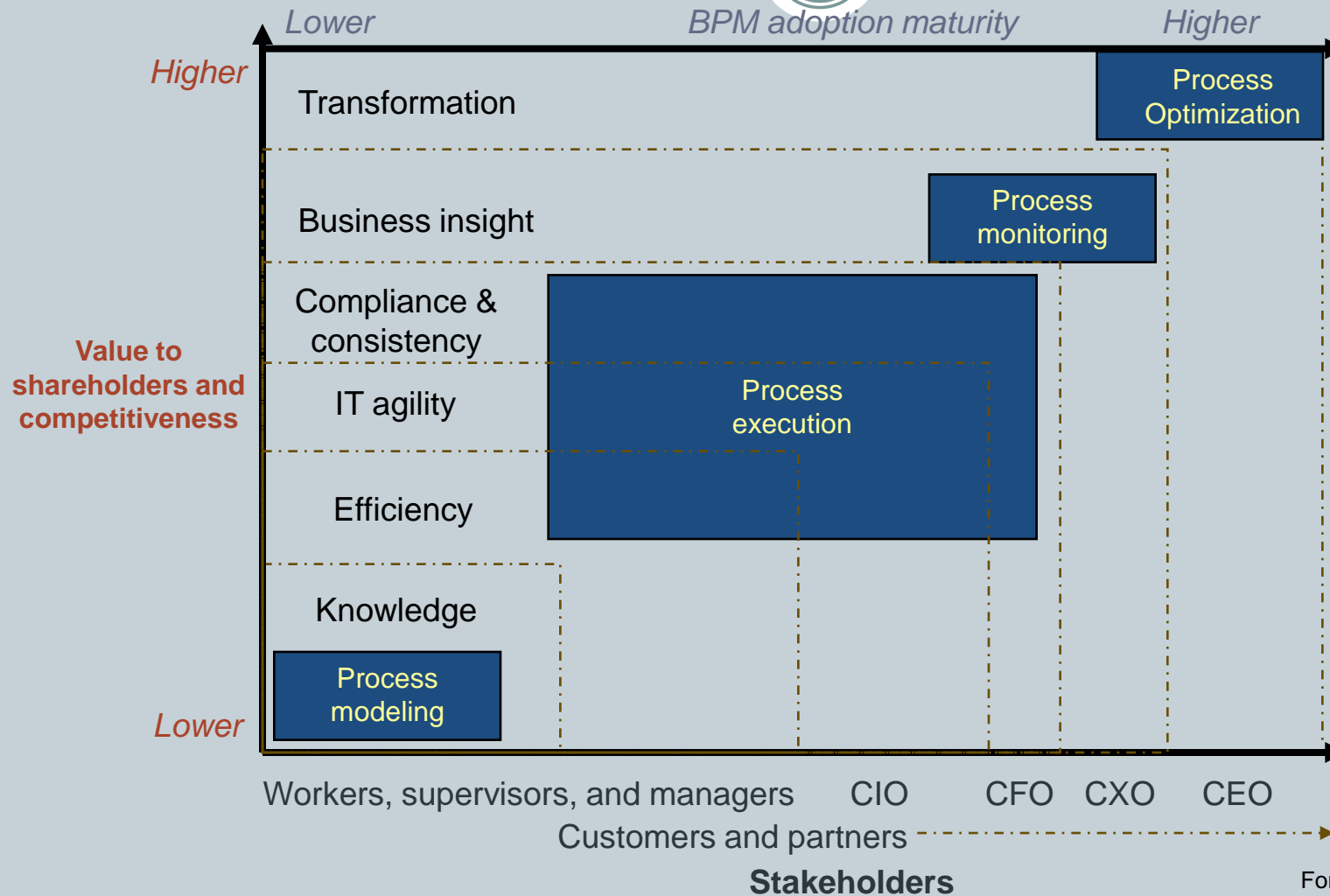
Business Process Lifecycle



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

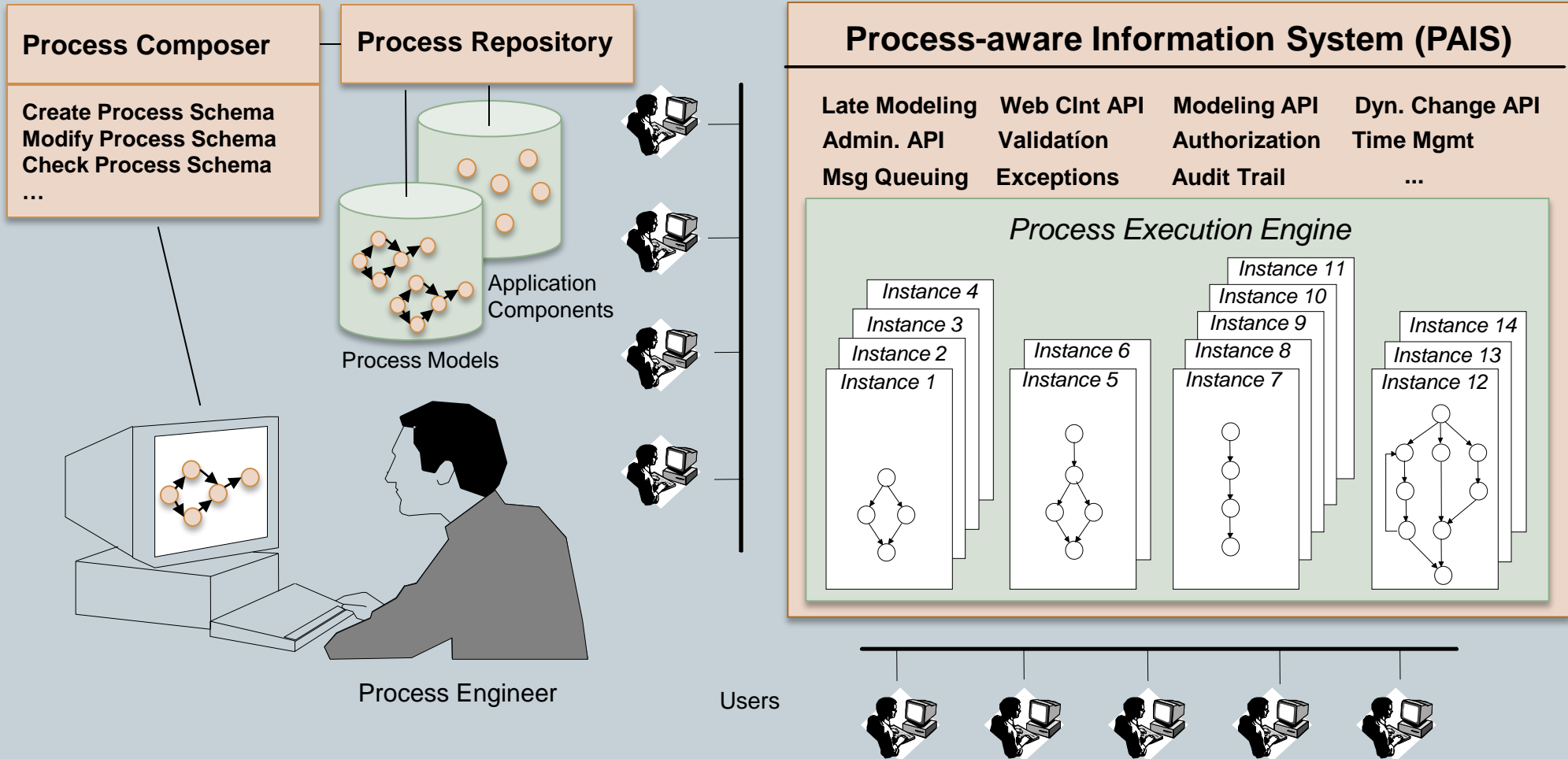
Fig 1.5. Business process lifecycle

BPM Value Proposition



Forester 2007 BPM Market Overview

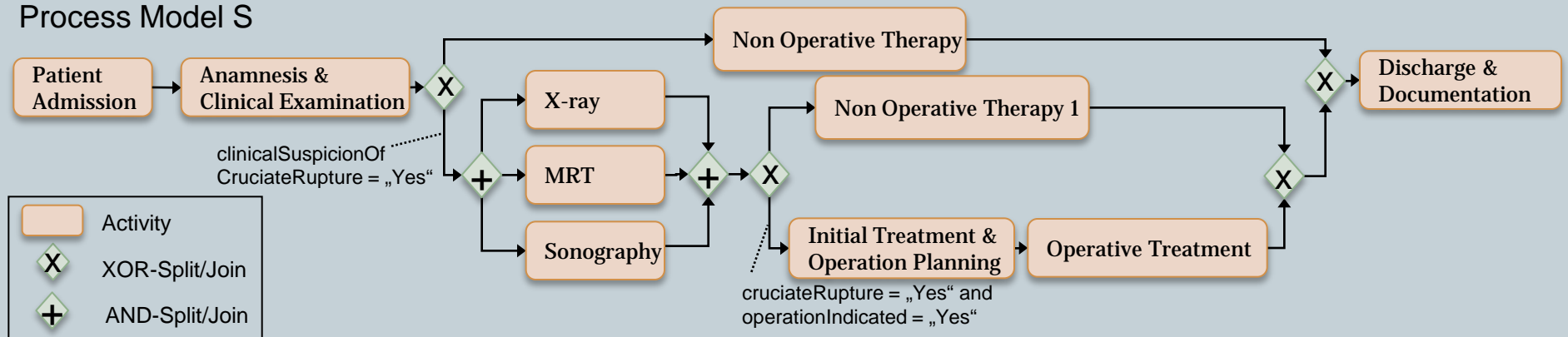
Process-aware Information System



Simple Process Model

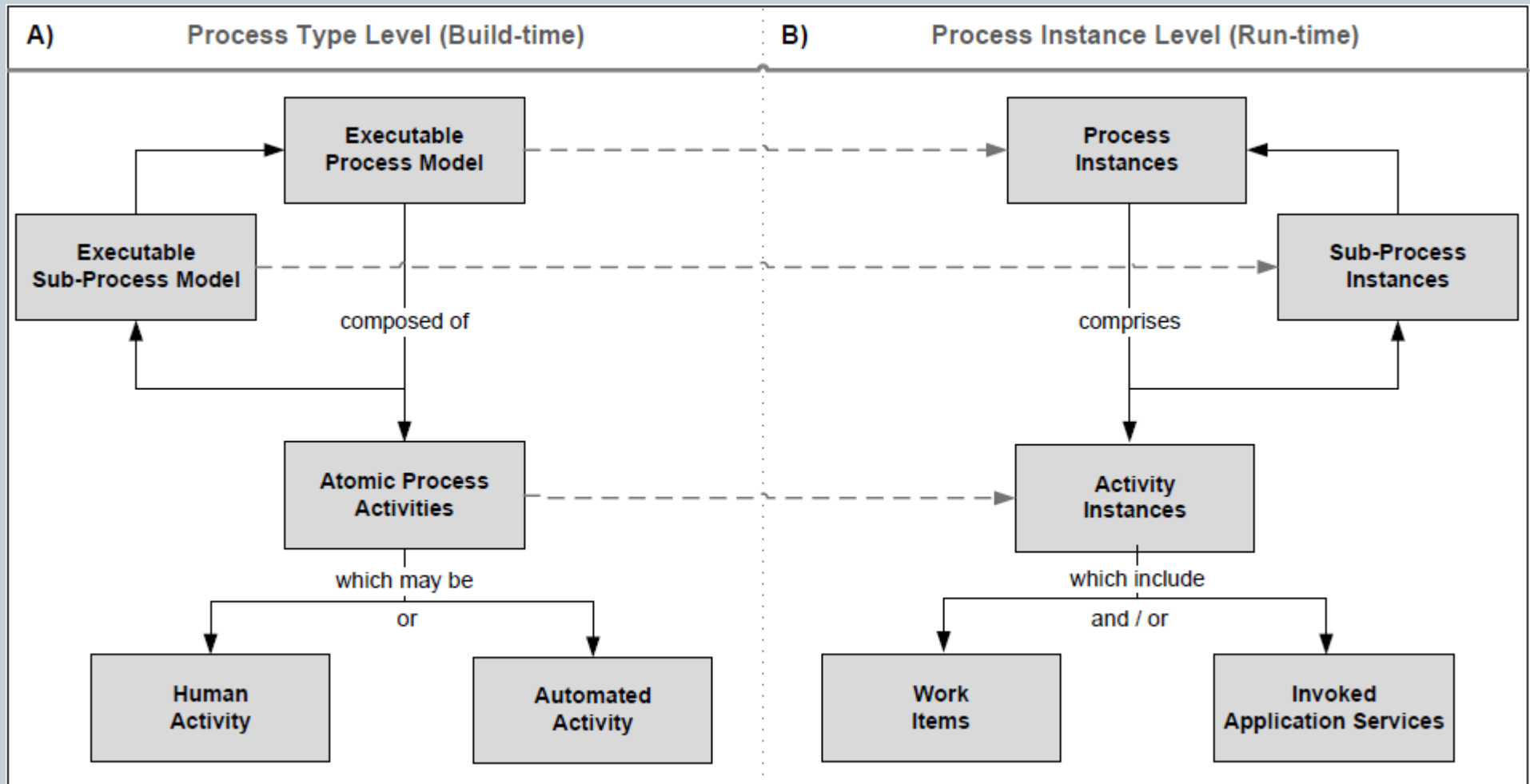


Process Model S



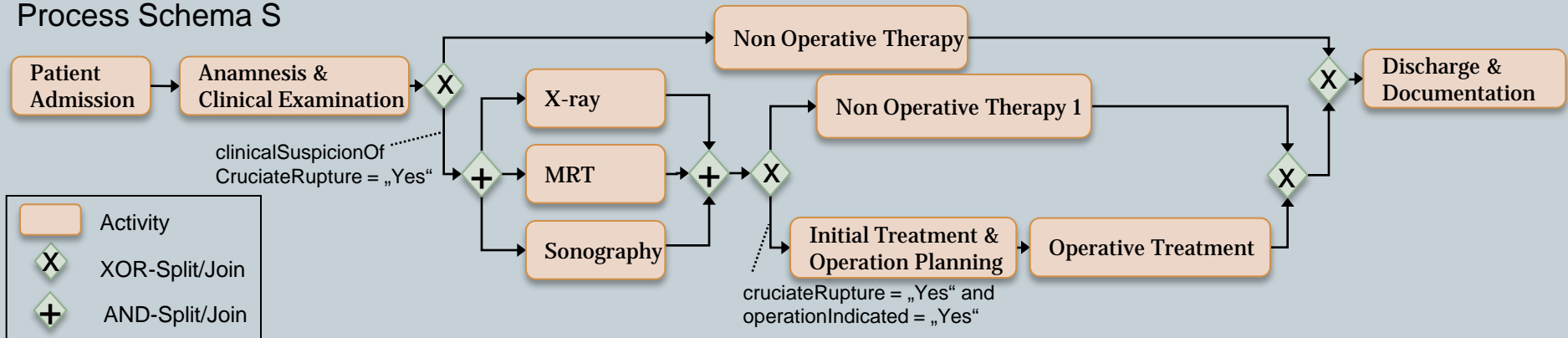
Built-time versus Run-time

- Process Type versus Process Instance Level -

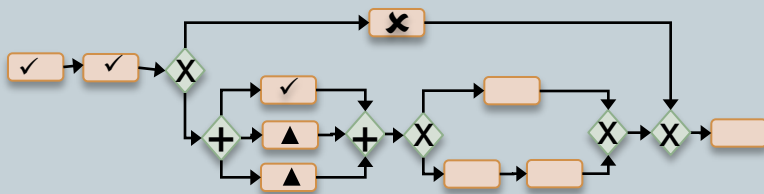


Business Process – System Perspective

Process Schema S



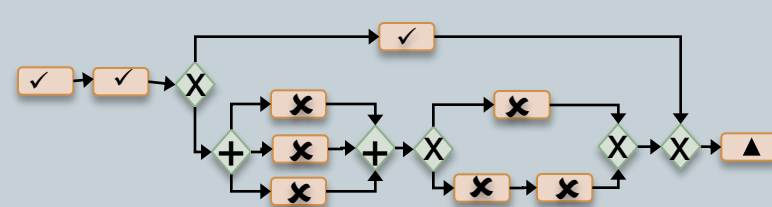
Process Instance I1



Execution Trace:

$\sigma_1 = \langle \text{„Patient Admission“}, \text{„Anamnesis \& Clinical Examination“}, \text{„X-ray“} \rangle$

Process Instance I2



Execution Trace:

$\sigma_2 = \langle \text{„Patient Admission“}, \text{„Anamnesis \& Clinical Examination“}, \text{„Non Operative Therapy“} \rangle$

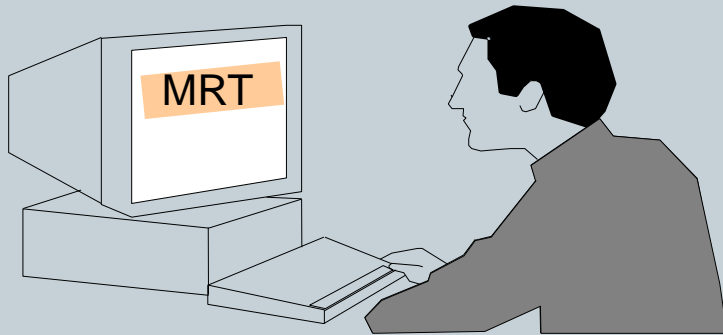
Activity States:

▲ Enabled

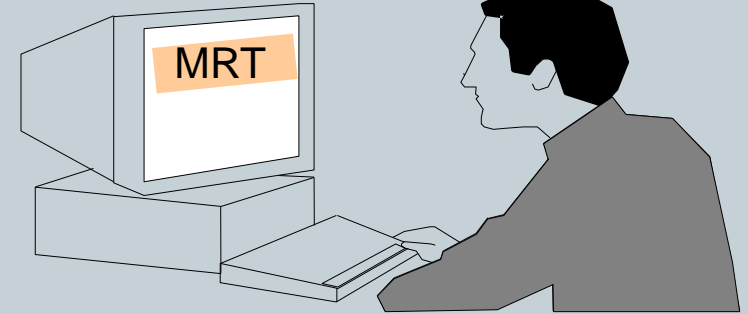
✓ Completed

✗ Skipped

User Perspective and Work item Lifecycle

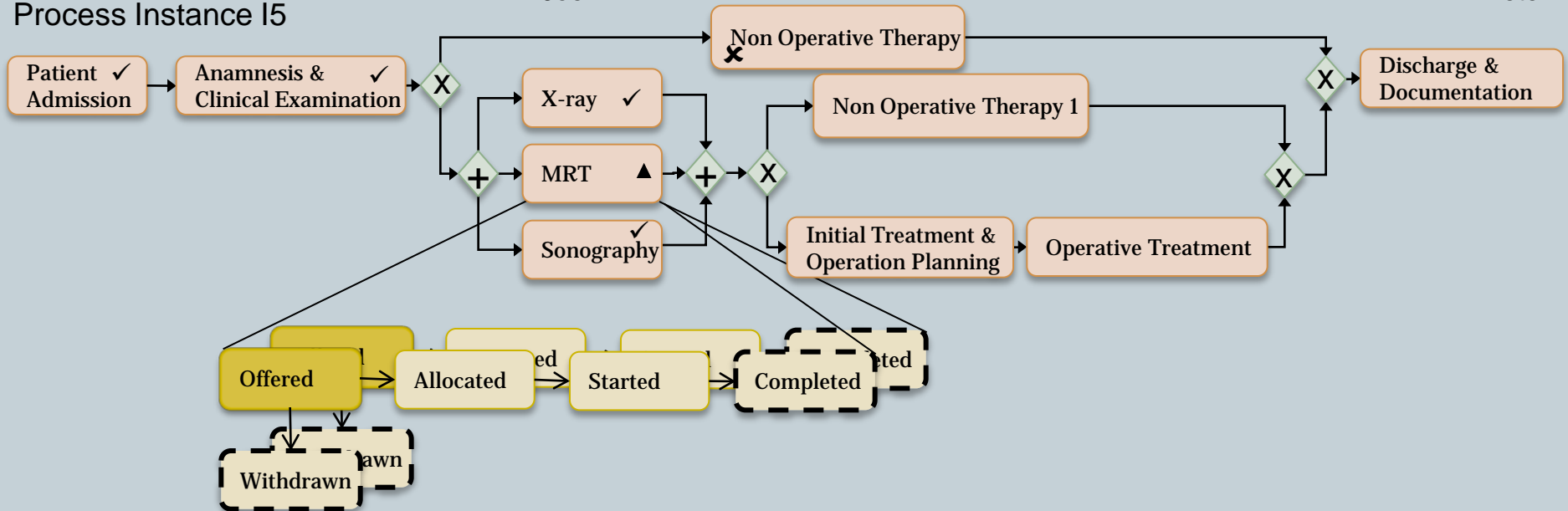


Joe



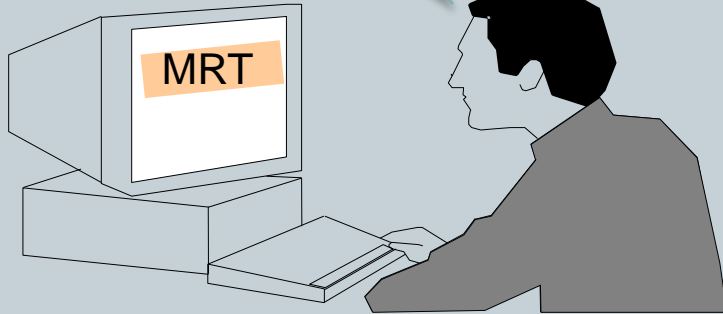
Peter

Process Instance I5

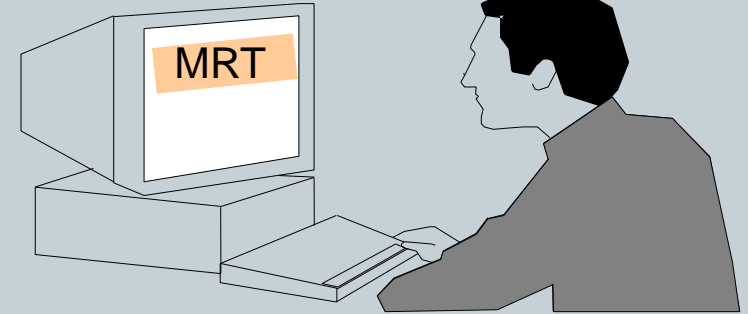


Let's do the MRT

User Perspective

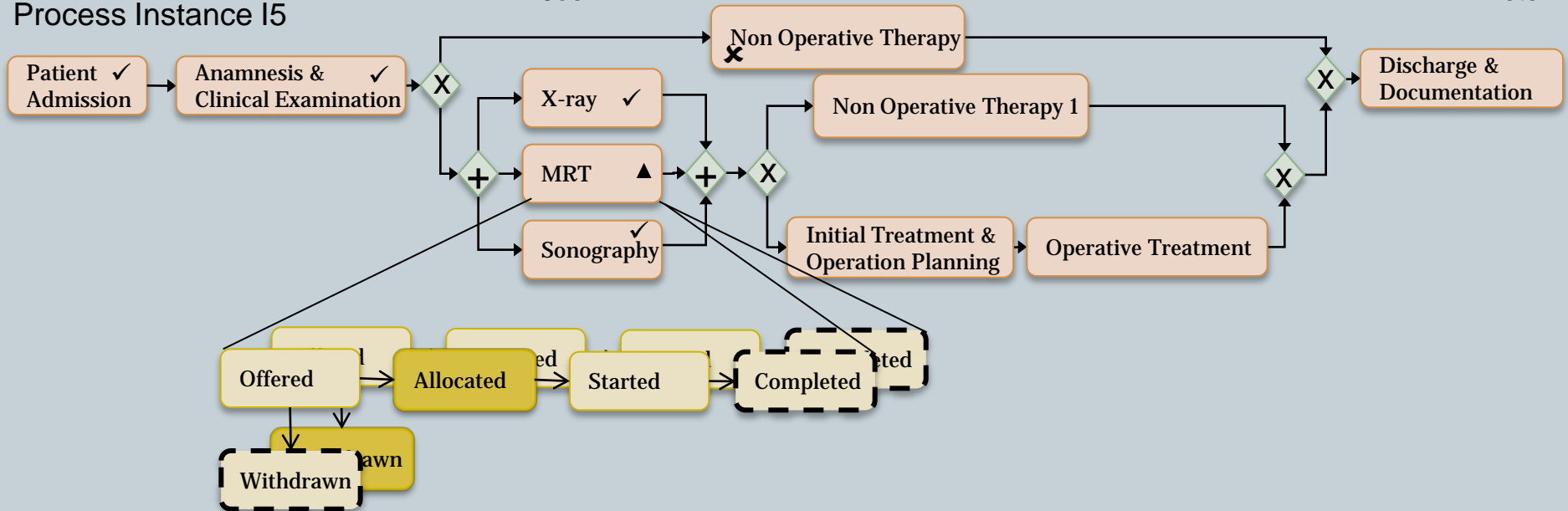


Joe



Peter

Process Instance I5



User Perspective

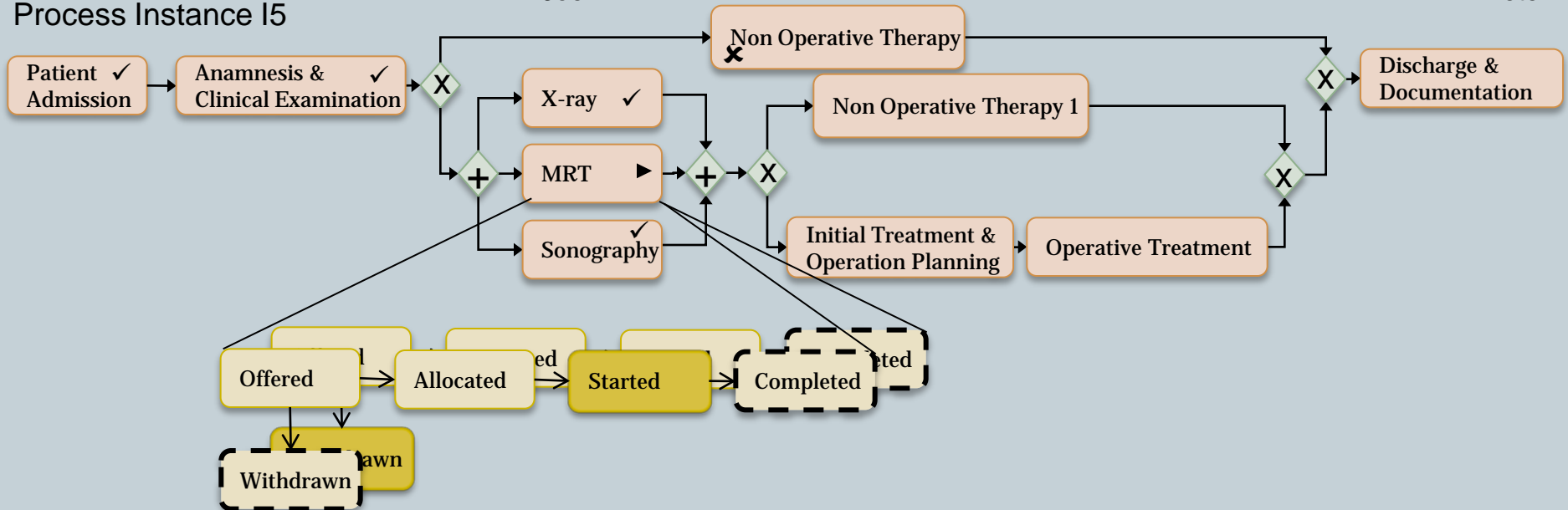


Joe



Peter

Process Instance I5



User Perspective

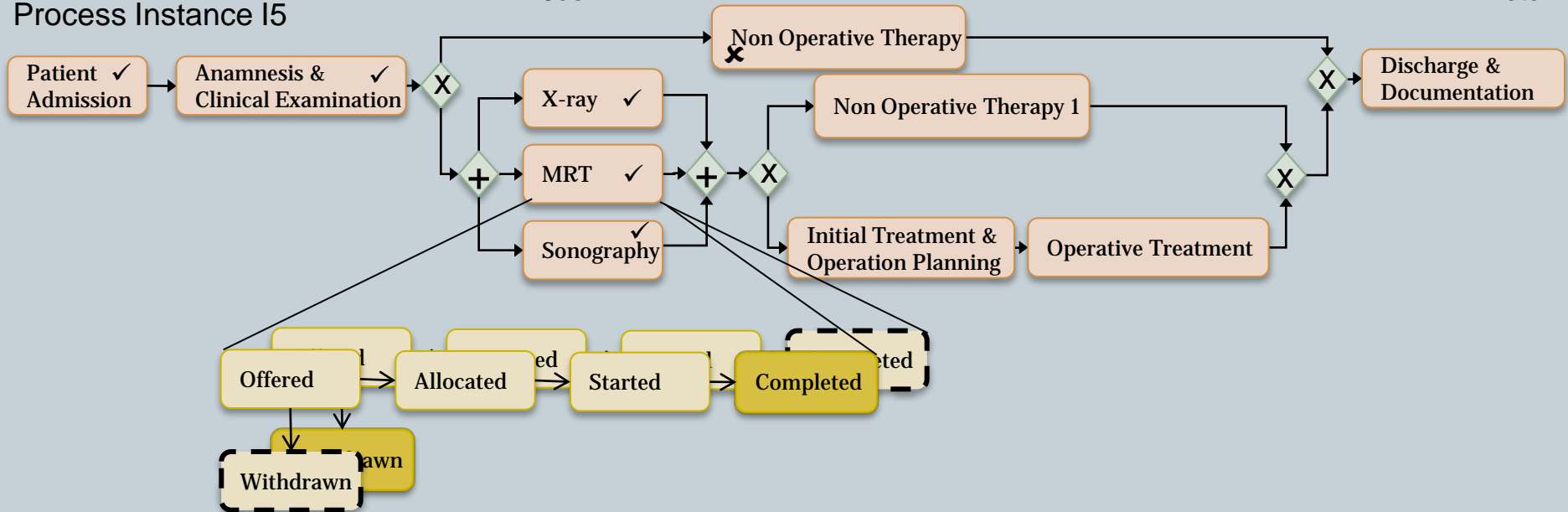


Joe



Peter

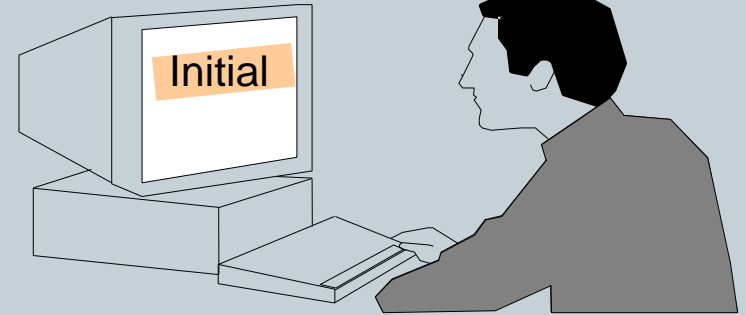
Process Instance I5



User Perspective

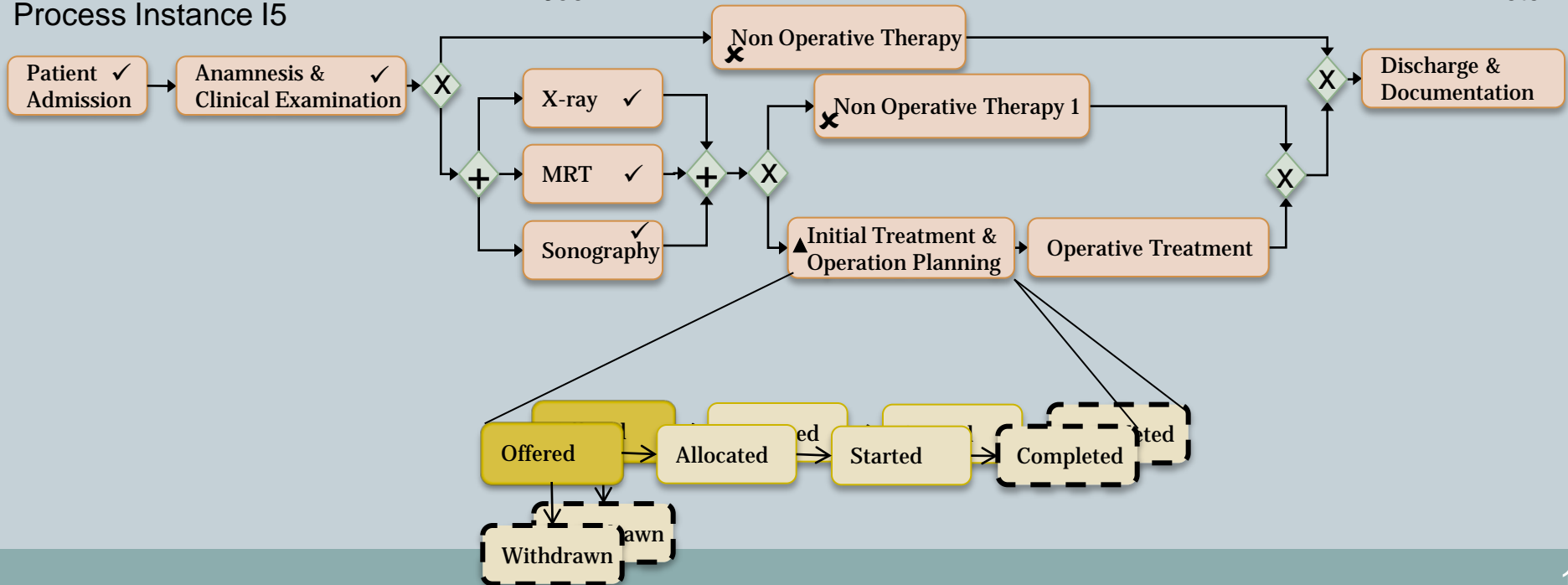


Joe



Peter

Process Instance I5



Business Processes and Workflows

Part 2 - Flexibility Issues

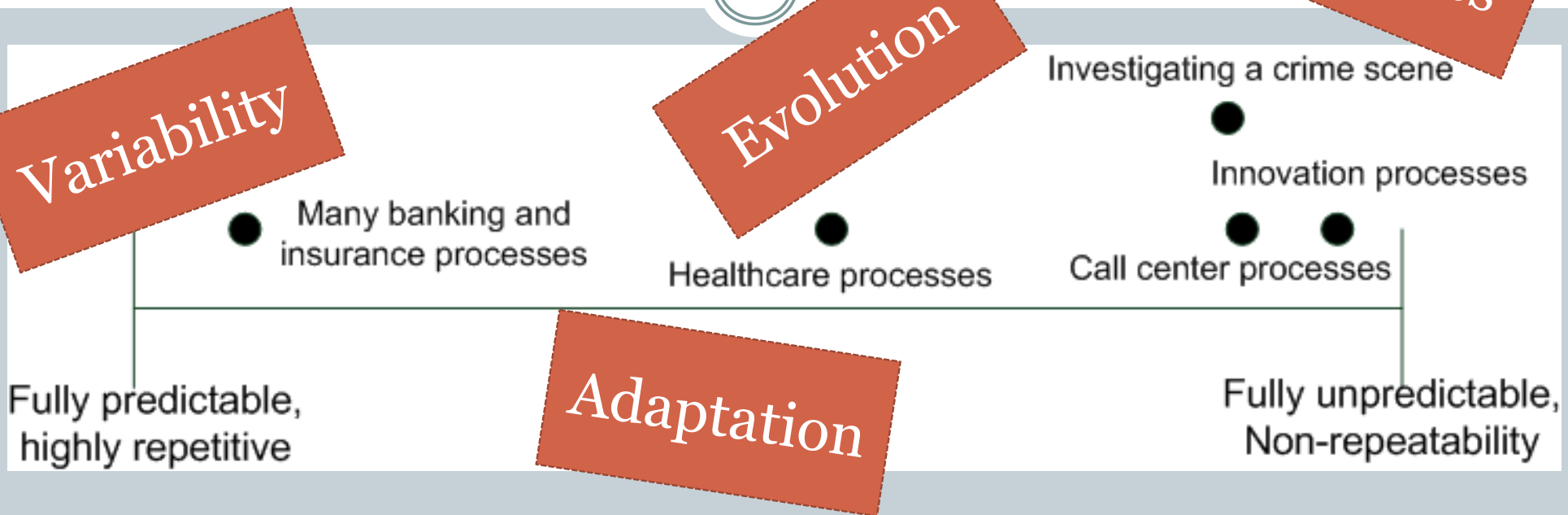


MONTEVIDEO, DECEMBER 11TH 2012



**PRESENTED BY
BARBARA WEBER
UNIV. OF INNSBRUCK**

Process Spectrum



Processes on the right side of the spectrum are mostly **knowledge-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

Variability



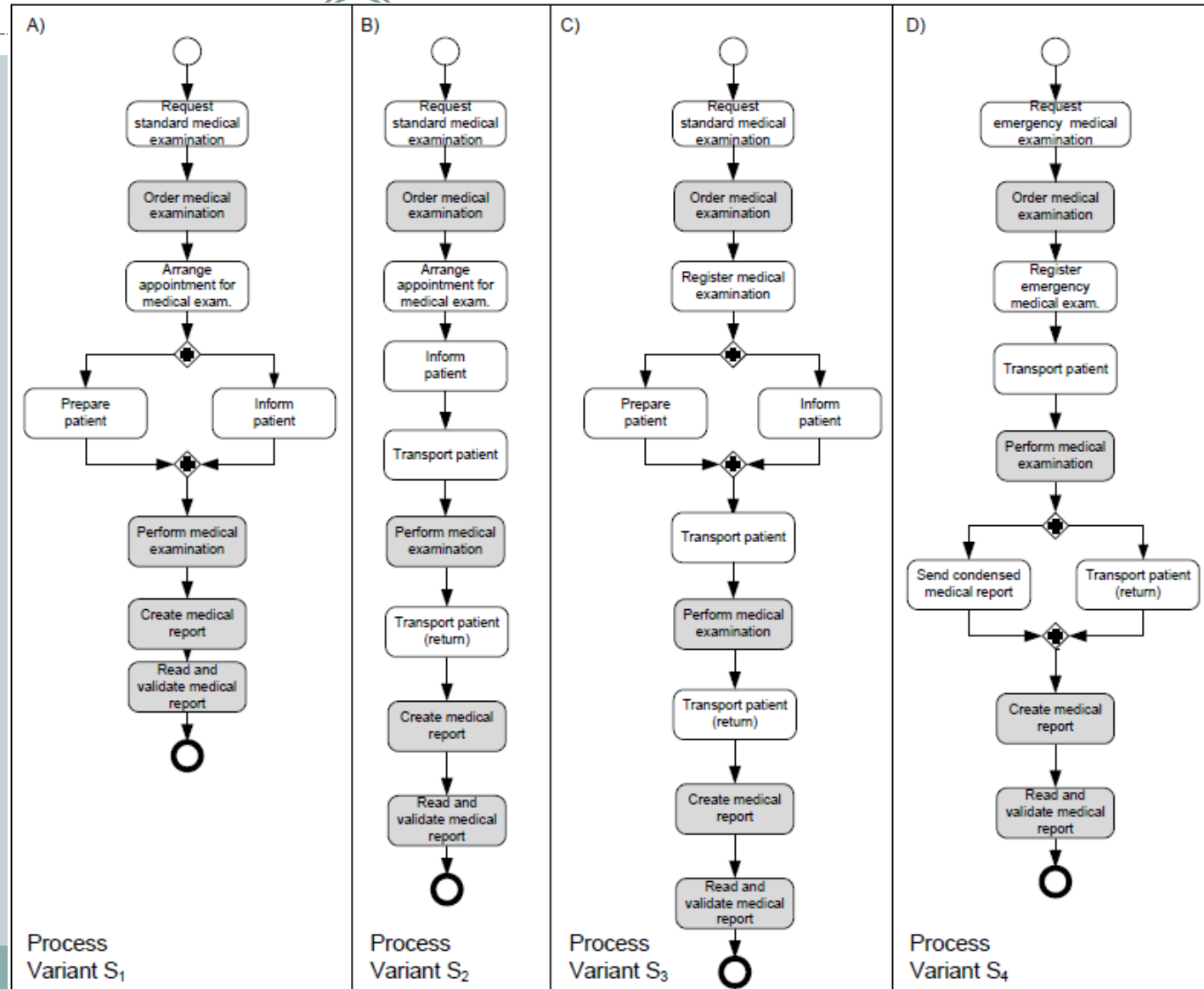
- Variability is typical for many domains and requires that processes are handled differently depending on the particular context
- Drivers
 - Product and service variability
 - Differences in regulations
 - Different customer groups
 - Temporal differences

Example:
Handling Medical
Examinations

Example: Handling Medical Examinations

Variety of related variants

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



Adaptation



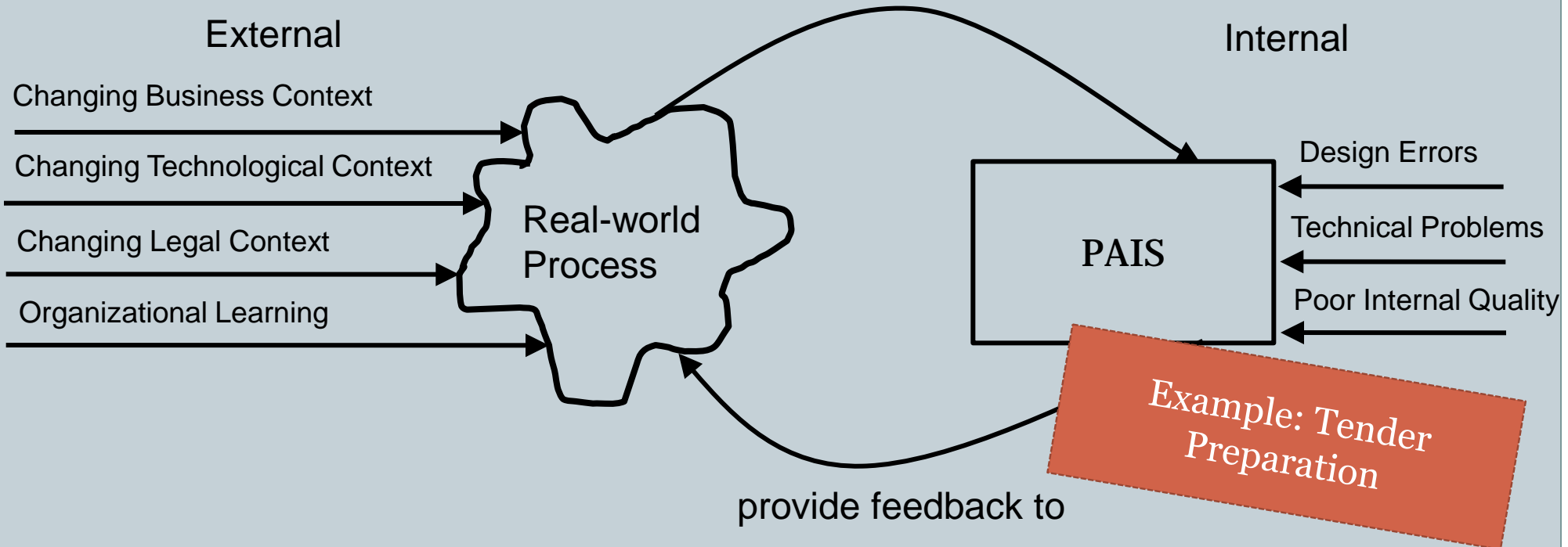
- Ability to adapt the process and its structure to temporary events
- Drivers
 - Special Situations
 - Exceptions
- Anticipation of Adaptation
 - Planned
 - Unanticipated

Example: Examination
Procedures in a Hospital

Evolution



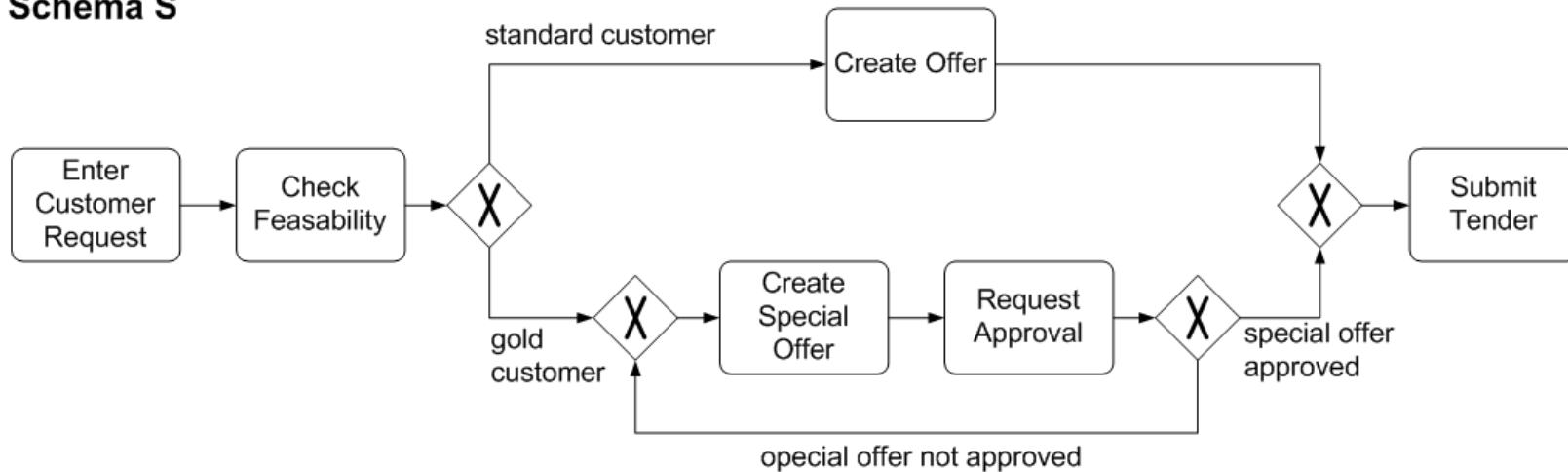
- Ability of the implemented process to change when the business process evolves
- Drivers



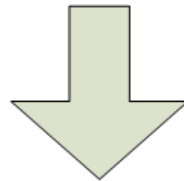
Example: Tender Preparation



Schema S



S evolves to S'



by applying change Δ_S with

$\Delta_S = \langle \text{Delete}(S, \text{Create Special Offer}), \text{Delete}(S, \text{Request Approval}) \rangle$

Schema S'



Evolution



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

Evolution



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

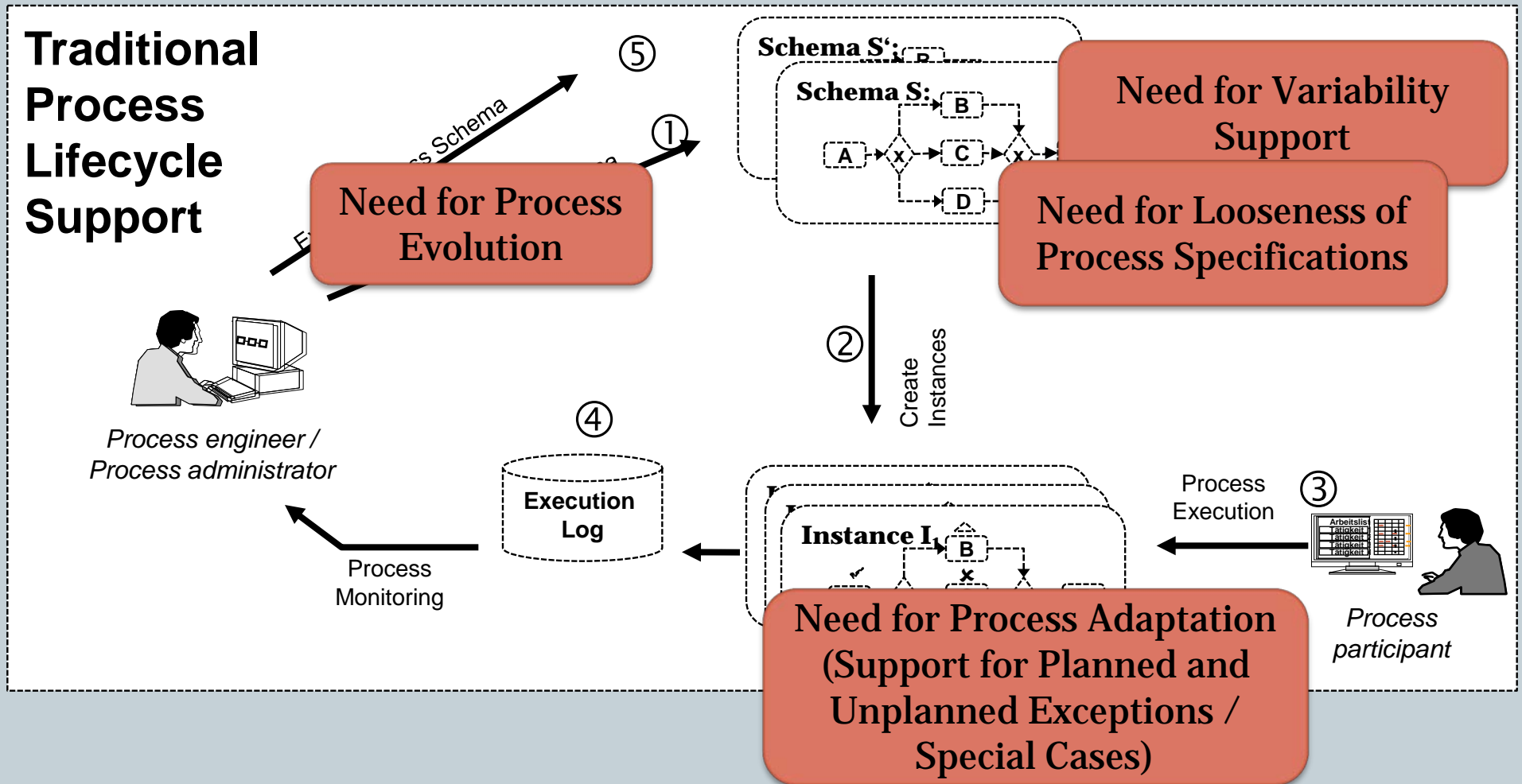
Looseness



- Knowledge-intensive processes cannot be fully pre-specified, but require loose specifications
- Drivers
 - Unpredictability
 - Non-Repeatability
 - Emergence

Example: Treatment Processes in a Hospital

Flexibility Issues along the Process Lifecycle



Flexibility Needs and Technological Requirements



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

Business Processes and Workflows

Pre-specified Process Models and Flexibility by Design



MONTEVIDEO, DECEMBER 11TH 2012



PRESENTED BY
BARBARA WEBER
UNIV. OF INNSBRUCK

Basic Control Flow Concepts – Activities



Atomic
Activities



Automated

Web services, Java applications,
database function



Human

Electronic forms

Complex
Activities



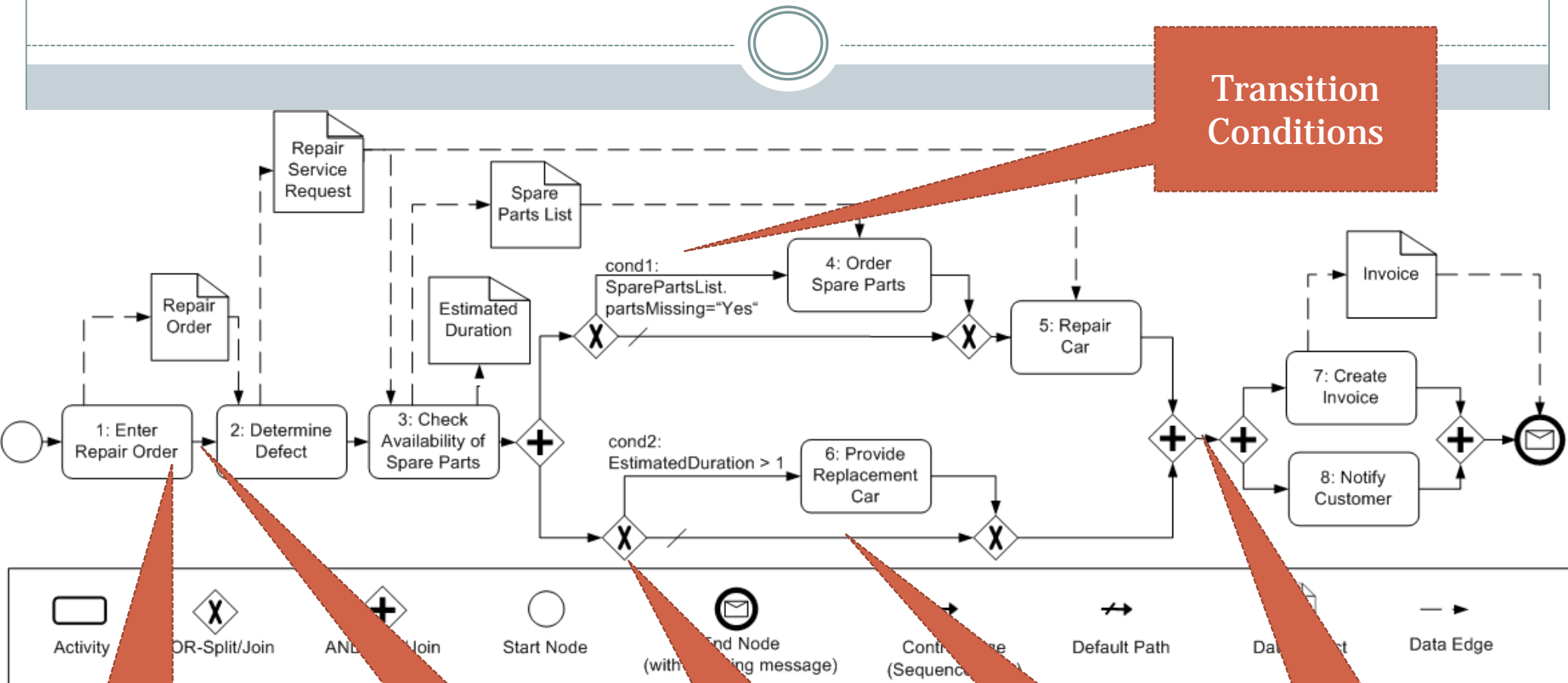
Refer to sub-process models

Basic Control Flow Concepts



- **Control Connectors (i.e., Gateways)**
 - (X)OR-Split / (X)OR-Join
 - AND-Split / AND-Join
- **Control Flow Edges**
 - Sequence Flow
 - Default Path
- **Transition Conditions**

Basic Control Flow Concepts - Example



Transition Conditions

Atomic Activity

Sequence Flow

XOR Gateway

Default Path

AND Gateway

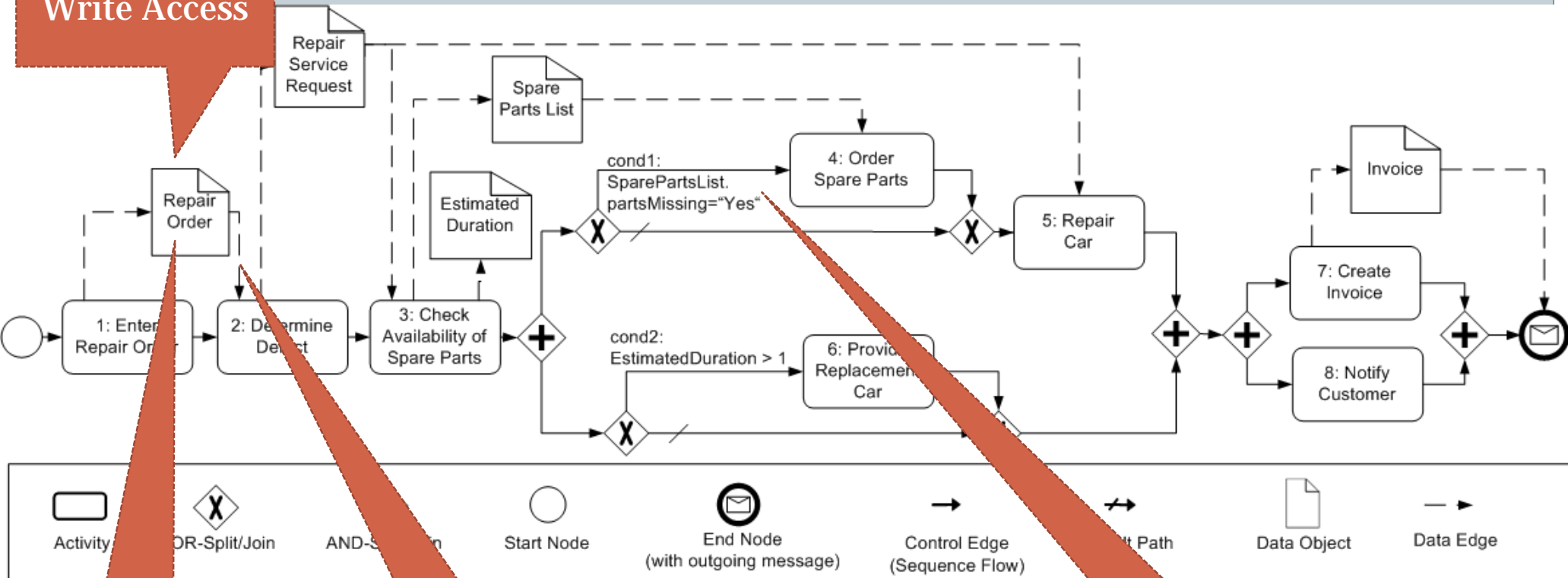
Basic Data Flow Concepts



- **Data objects + Data edges**
- **Data objects can be linked to**
 - activities via data edges
 - ✦ Read access
 - ✦ Write access
 - referenced by transition conditions
 - attached to outgoing messages

Basic Data Flow Concepts - Example

Data Edge –
Write Access



Data Object

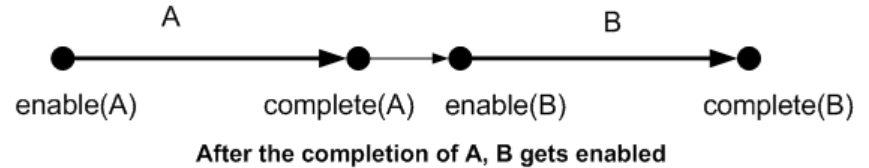
Data Edge –
Read Access

Transition
Condition
references
SparePartsList

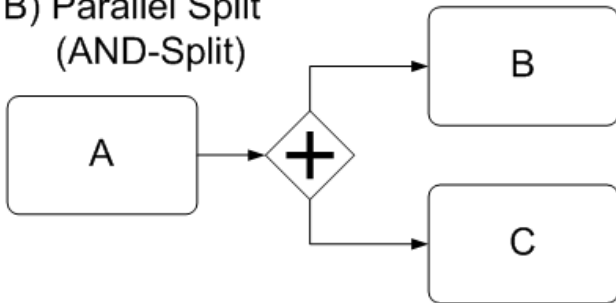
Examples of Control Flow Patterns (1)



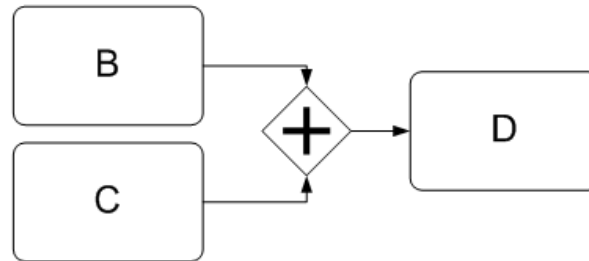
A) Sequence Pattern



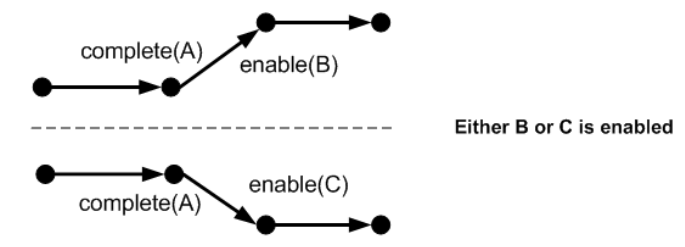
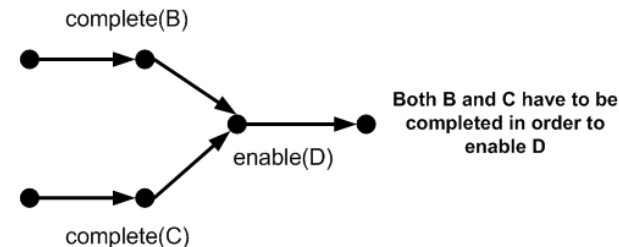
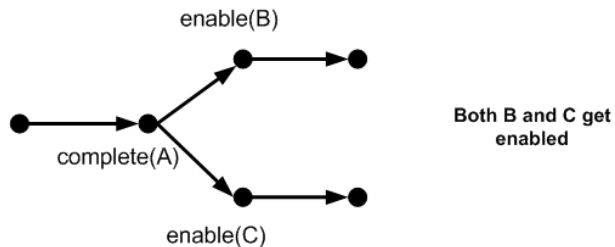
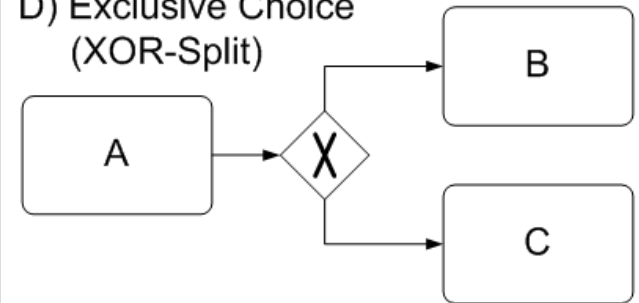
B) Parallel Split (AND-Split)



C) Synchronization (AND-Join)



D) Exclusive Choice (XOR-Split)

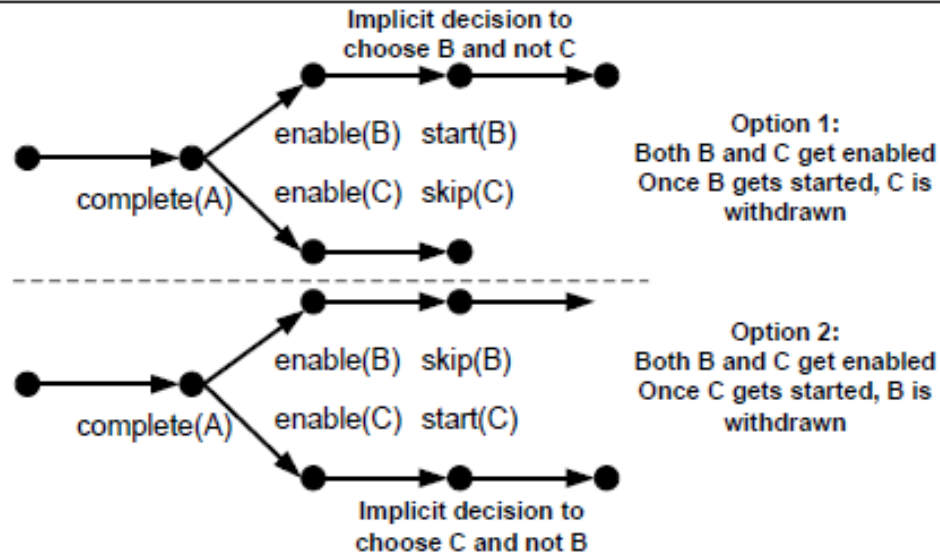
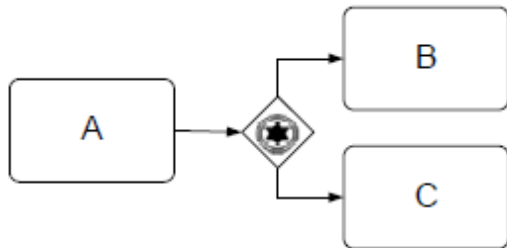


For an overview of patterns visit: <http://www.workflowpatterns.org>

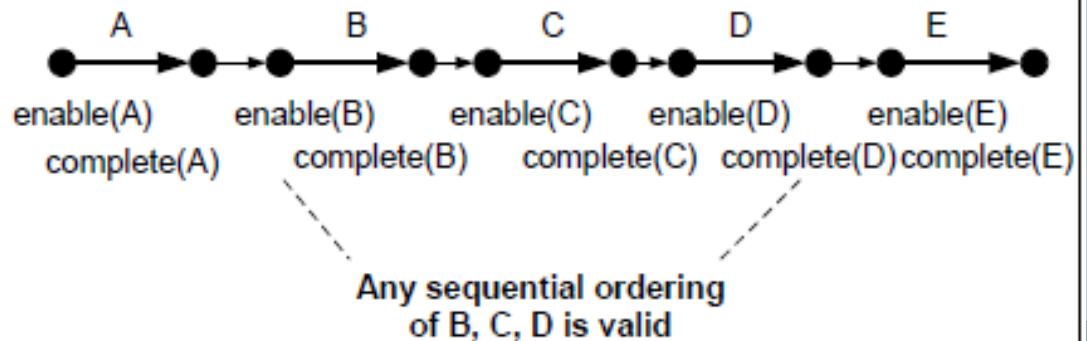
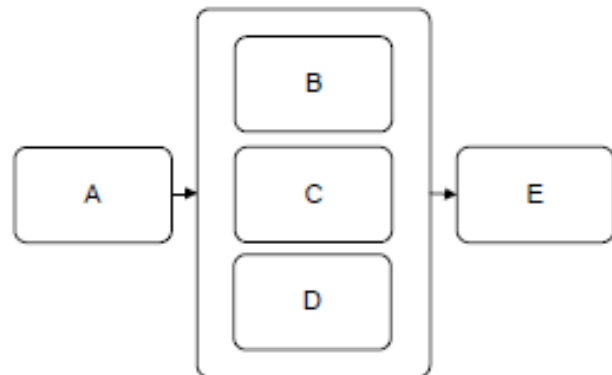
Examples of Control Flow Patterns (2)



D) Deferred Choice



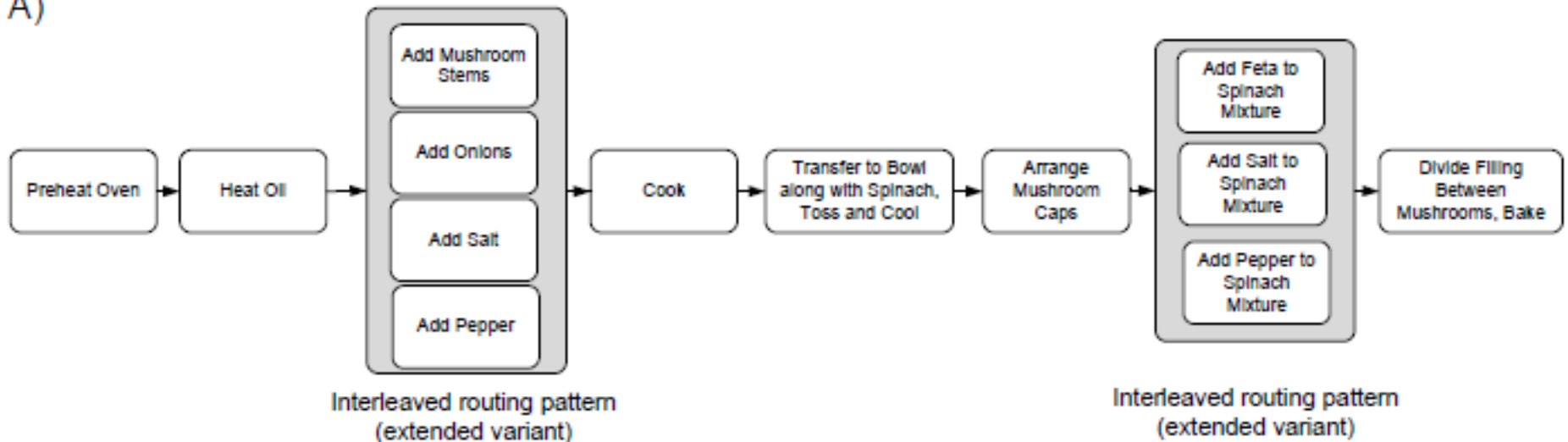
E) Interleaved Routing



Expressiveness and Flexibility by Design

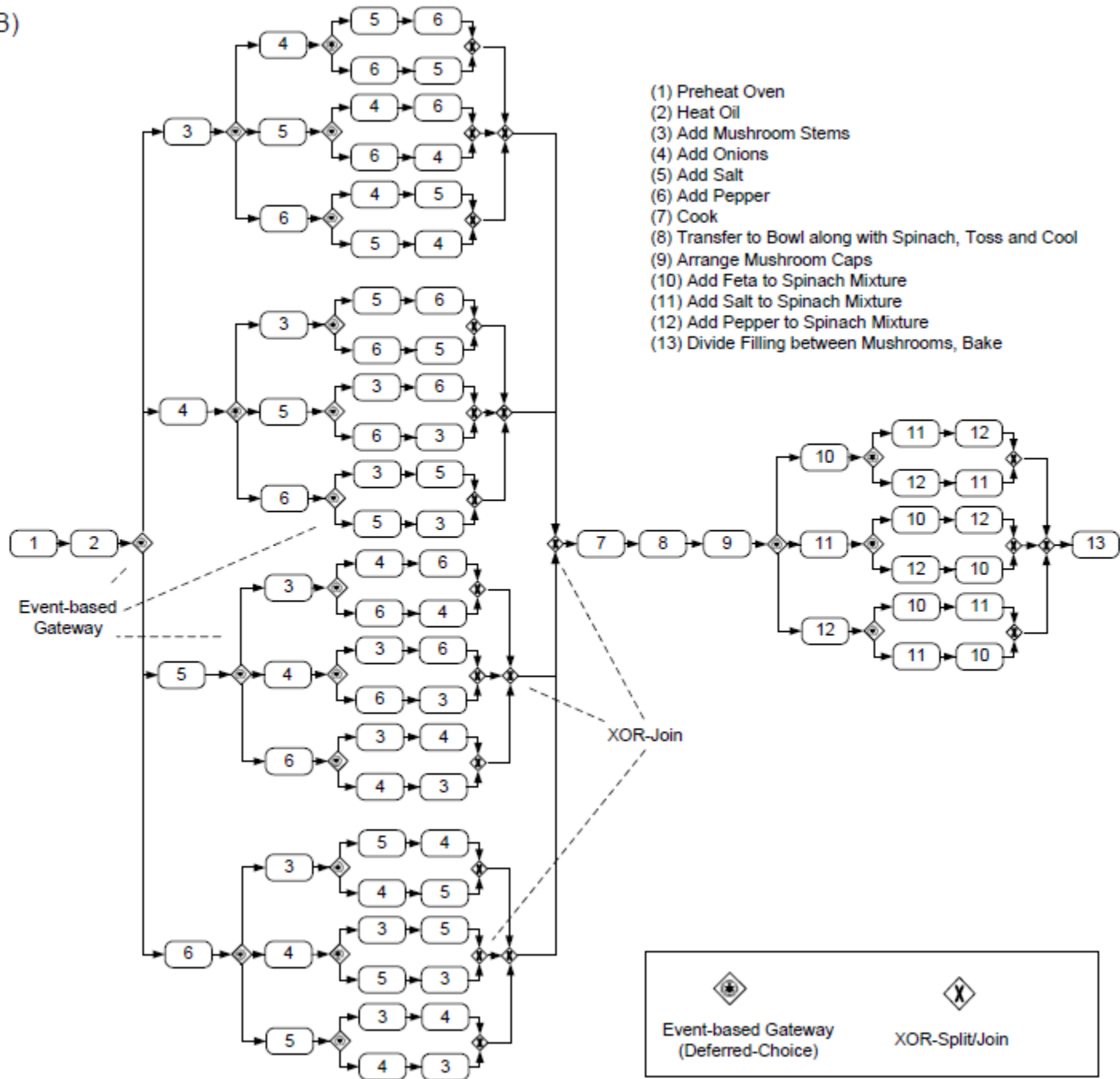


A)



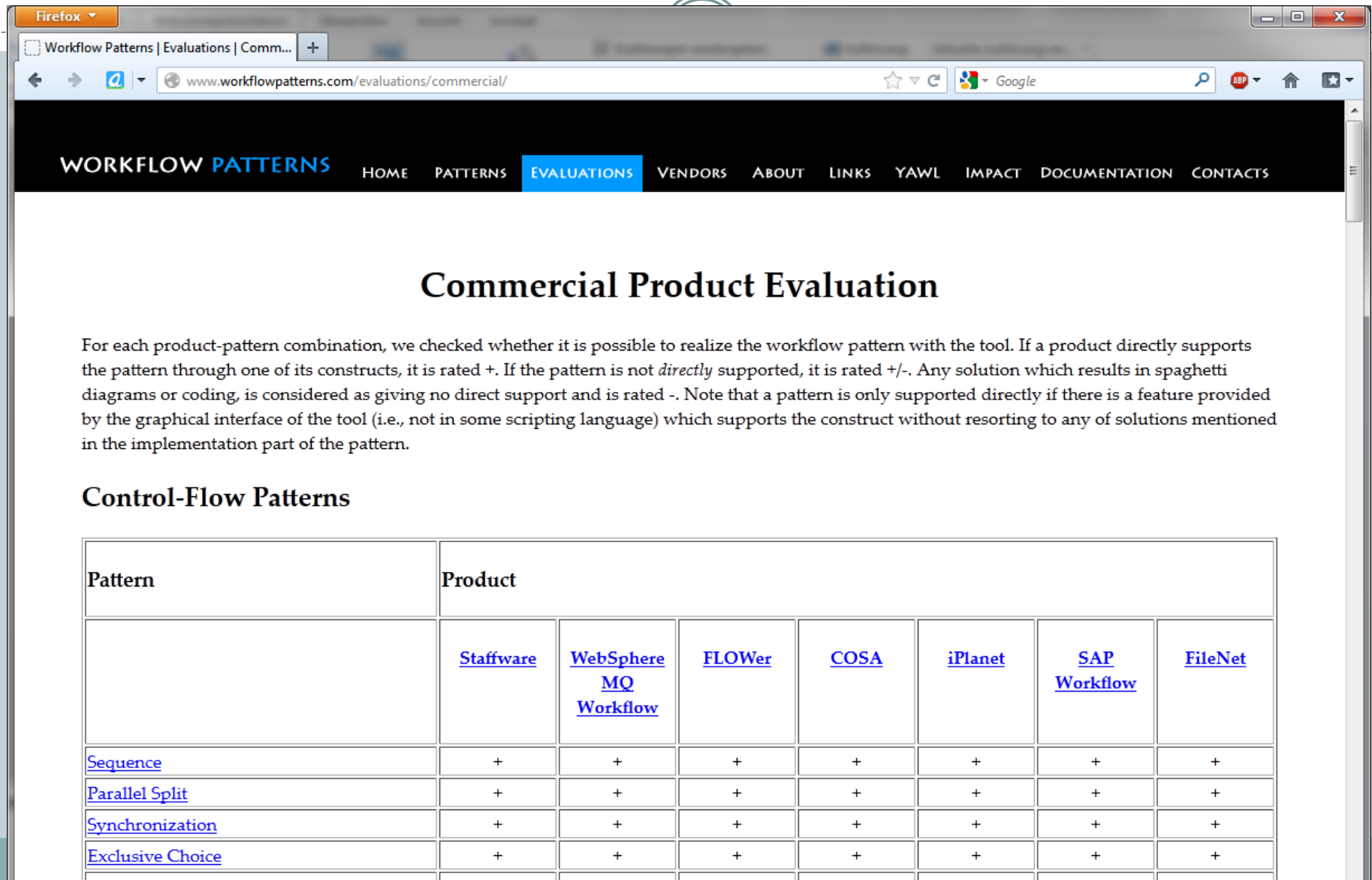
(Missing) Expressiveness and Flexibility by Design

B)



- (1) Preheat Oven
- (2) Heat Oil
- (3) Add Mushroom Stems
- (4) Add Onions
- (5) Add Salt
- (6) Add Pepper
- (7) Cook
- (8) Transfer to Bowl along with Spinach, Toss and Cool
- (9) Arrange Mushroom Caps
- (10) Add Feta to Spinach Mixture
- (11) Add Salt to Spinach Mixture
- (12) Add Pepper to Spinach Mixture
- (13) Divide Filling between Mushrooms, Bake

Evaluation of Existing PAISs



The screenshot shows a Firefox browser window displaying the 'Commercial Product Evaluation' page on the Workflow Patterns website. The page features a navigation menu with 'EVALUATIONS' highlighted. The main content area is titled 'Commercial Product Evaluation' and includes a detailed paragraph explaining the evaluation methodology. Below this, a section titled 'Control-Flow Patterns' contains a table with 8 columns representing different products and 5 rows representing different workflow patterns. The table shows that all listed patterns are supported (+) by all listed products.

Commercial Product Evaluation

For each product-pattern combination, we checked whether it is possible to realize the workflow pattern with the tool. If a product directly supports the pattern through one of its constructs, it is rated +. If the pattern is not *directly* supported, it is rated +/- . Any solution which results in spaghetti diagrams or coding, is considered as giving no direct support and is rated -. Note that a pattern is only supported directly if there is a feature provided by the graphical interface of the tool (i.e., not in some scripting language) which supports the construct without resorting to any of solutions mentioned in the implementation part of the pattern.

Control-Flow Patterns

Pattern	Product						
	Staffware	WebSphere MQ Workflow	FLOWer	COSA	iPlanet	SAP Workflow	FileNet
Sequence	+	+	+	+	+	+	+
Parallel Split	+	+	+	+	+	+	+
Synchronization	+	+	+	+	+	+	+
Exclusive Choice	+	+	+	+	+	+	+

Business Processes and Workflows

Process Configuration



MONTEVIDEO, DECEMBER 11TH 2012



PRESENTED BY
BARBARA WEBER
UNIV. OF INNSBRUCK

*Not part of this keynote
For details see Chapter 5
of the book*

Business Processes and Workflows

Exception and Compensation Handling

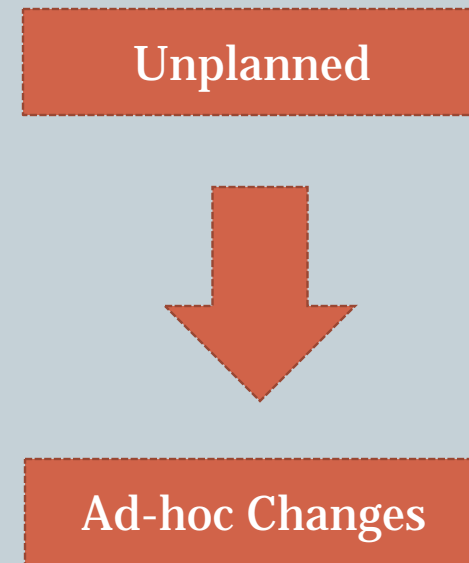
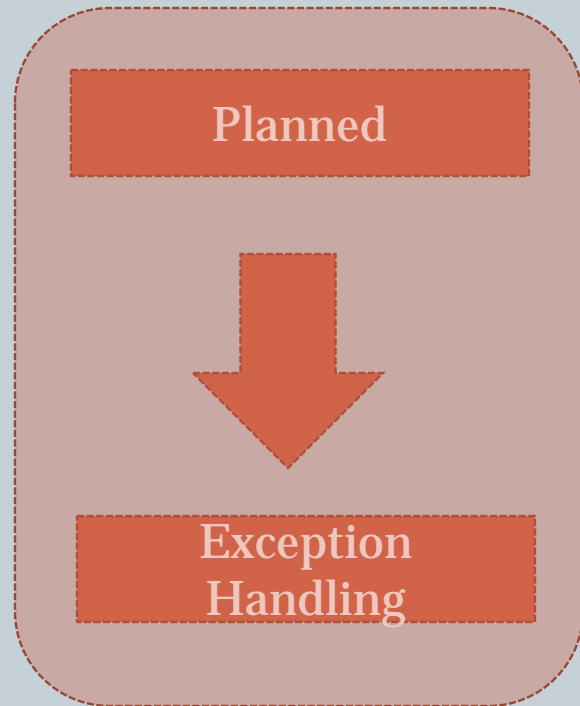


MONTEVIDEO, DECEMBER 11TH 2012



PRESENTED BY
BARBARA WEBER
UNIV. OF INNSBRUCK

Process Adaptations



Exception Handling in PAIS

Sources for Exceptions

Activity Failure

Technical

Semantical

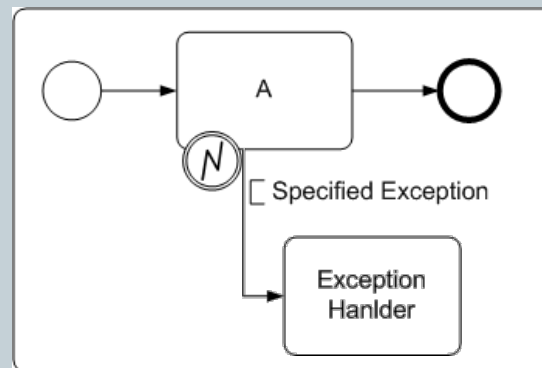
Deadline Expiry

Resource Unavailability

Inconsistence real-world / PAIS

Constraint Violations

Upon detection of a particular exception



a suitable handler is chosen

Exception Handler

Trying Alternatives

Ordered

Unordered

Add Behavior

Deferred Fixing

Immediate Fixing

Retry

Exception-driven Rework

Cancelling Behavior

Reject

Compensate

Resource Patterns

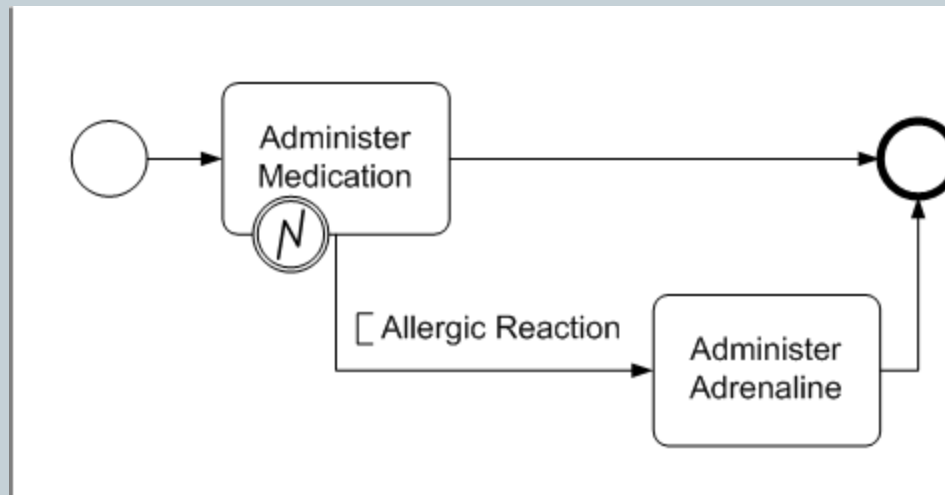
Delegate

Escalate

...

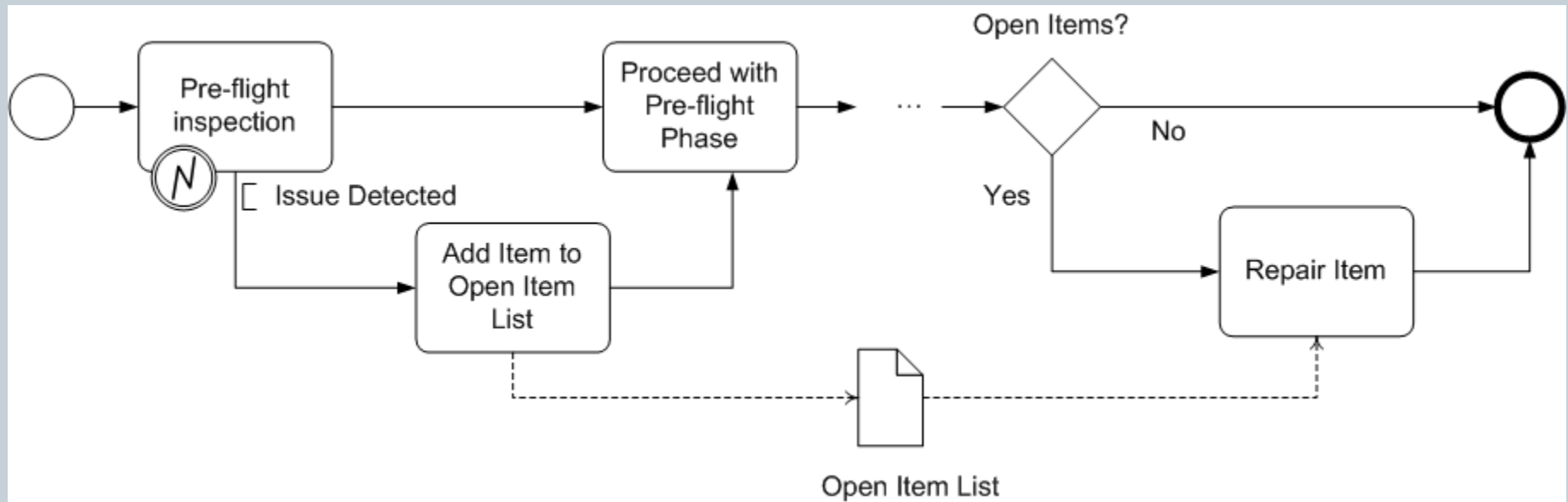
Exception Handling Patterns

- Immediate Fixing -



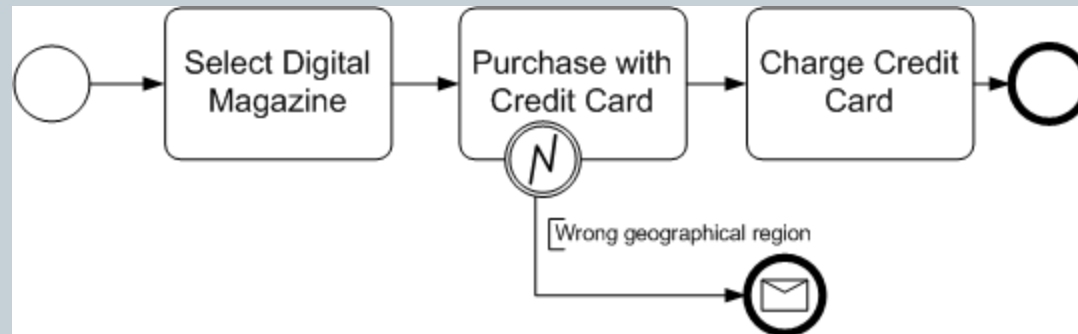
Exception Handling Patterns

- Deferred Fixing -



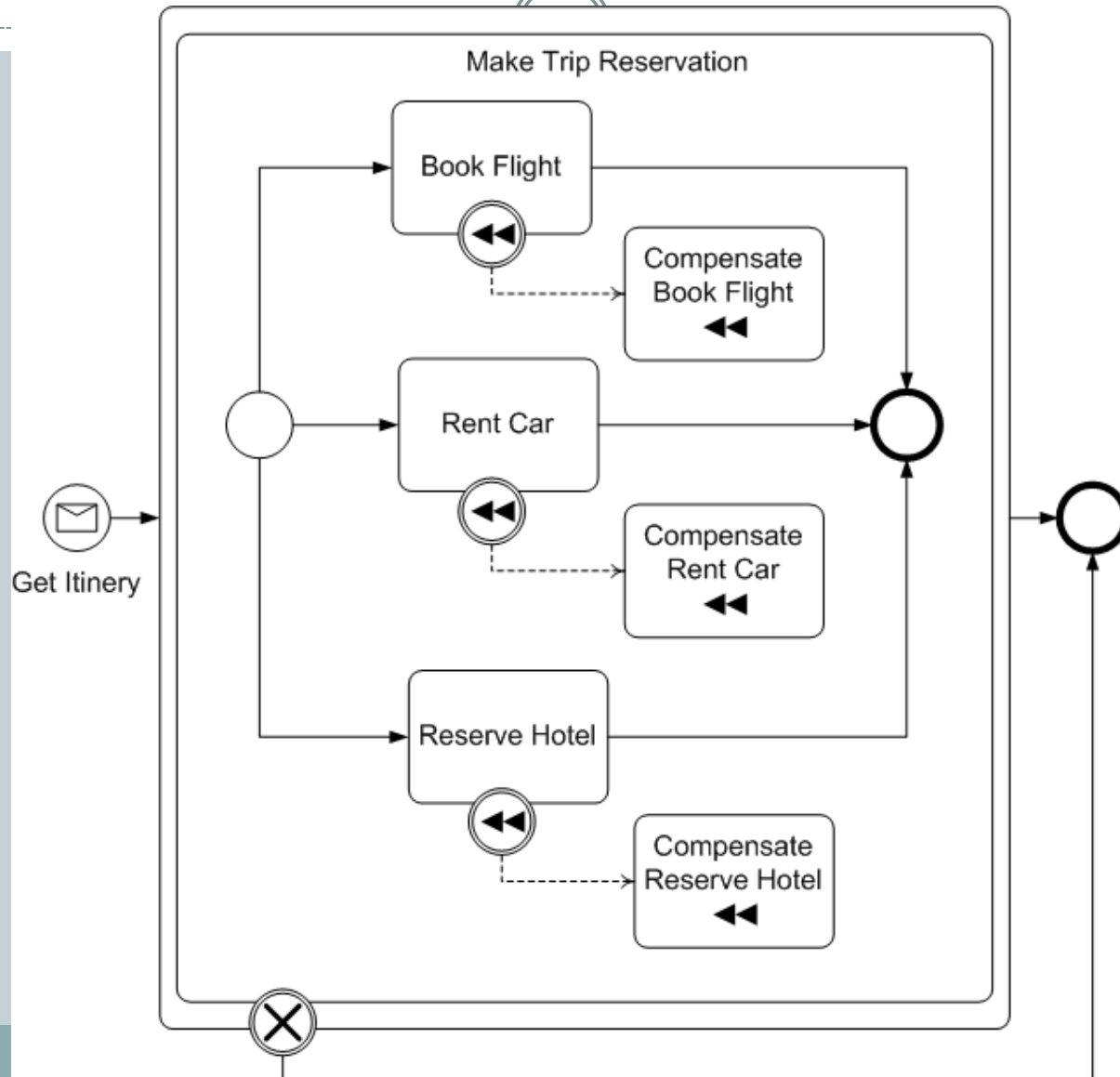
Exception Handling Patterns

- Reject -



Exception Handling Patterns

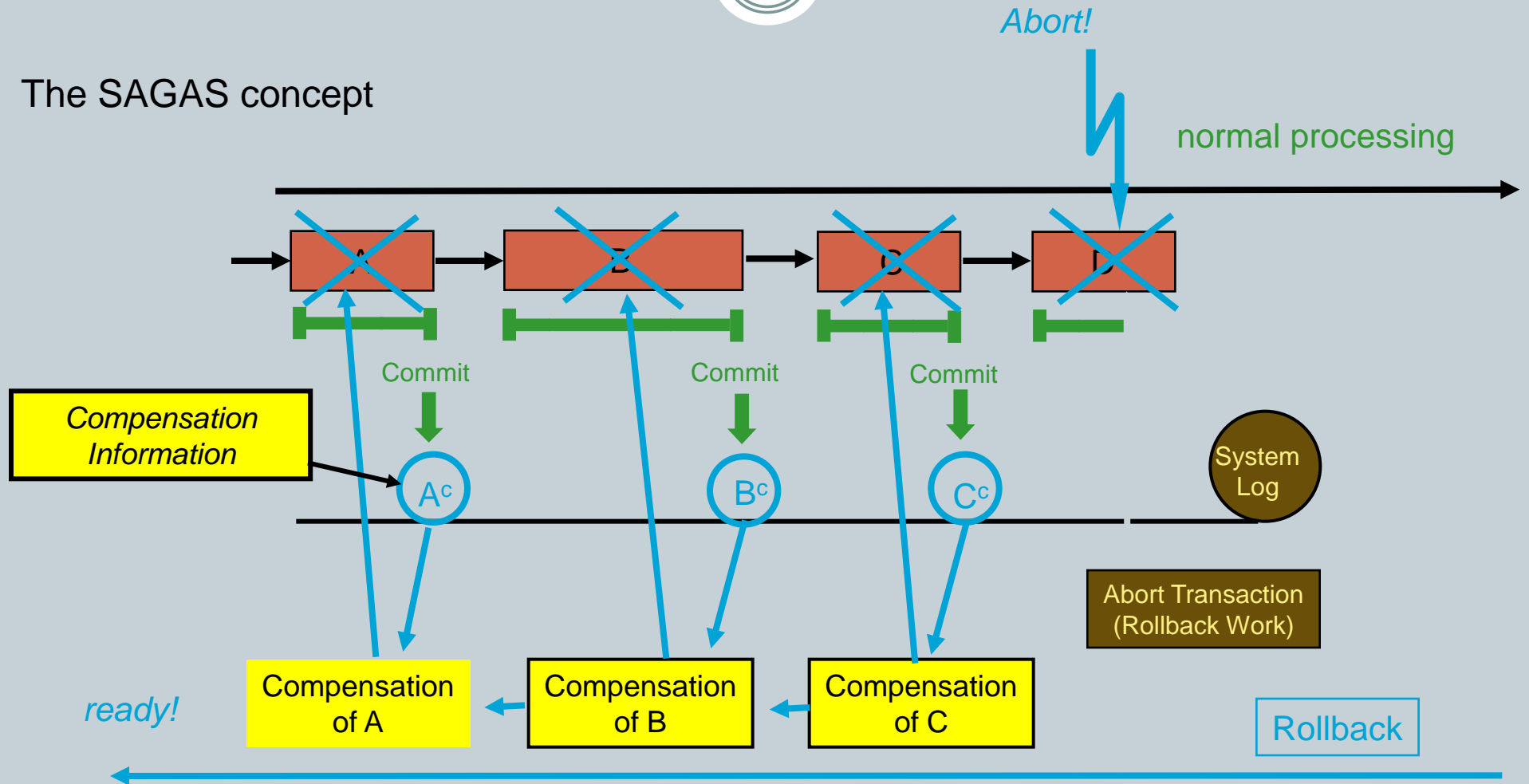
- Compensate -



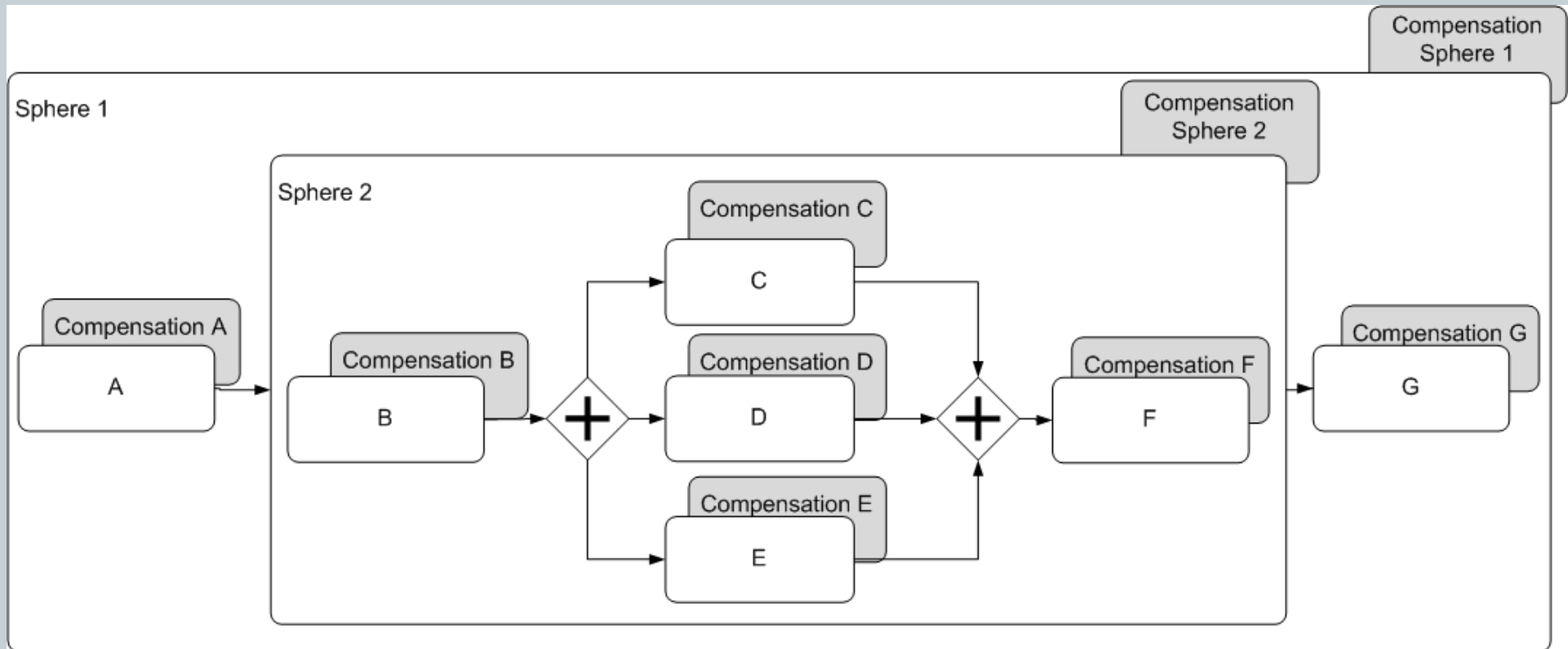
Compensation Handling - Sagas

49

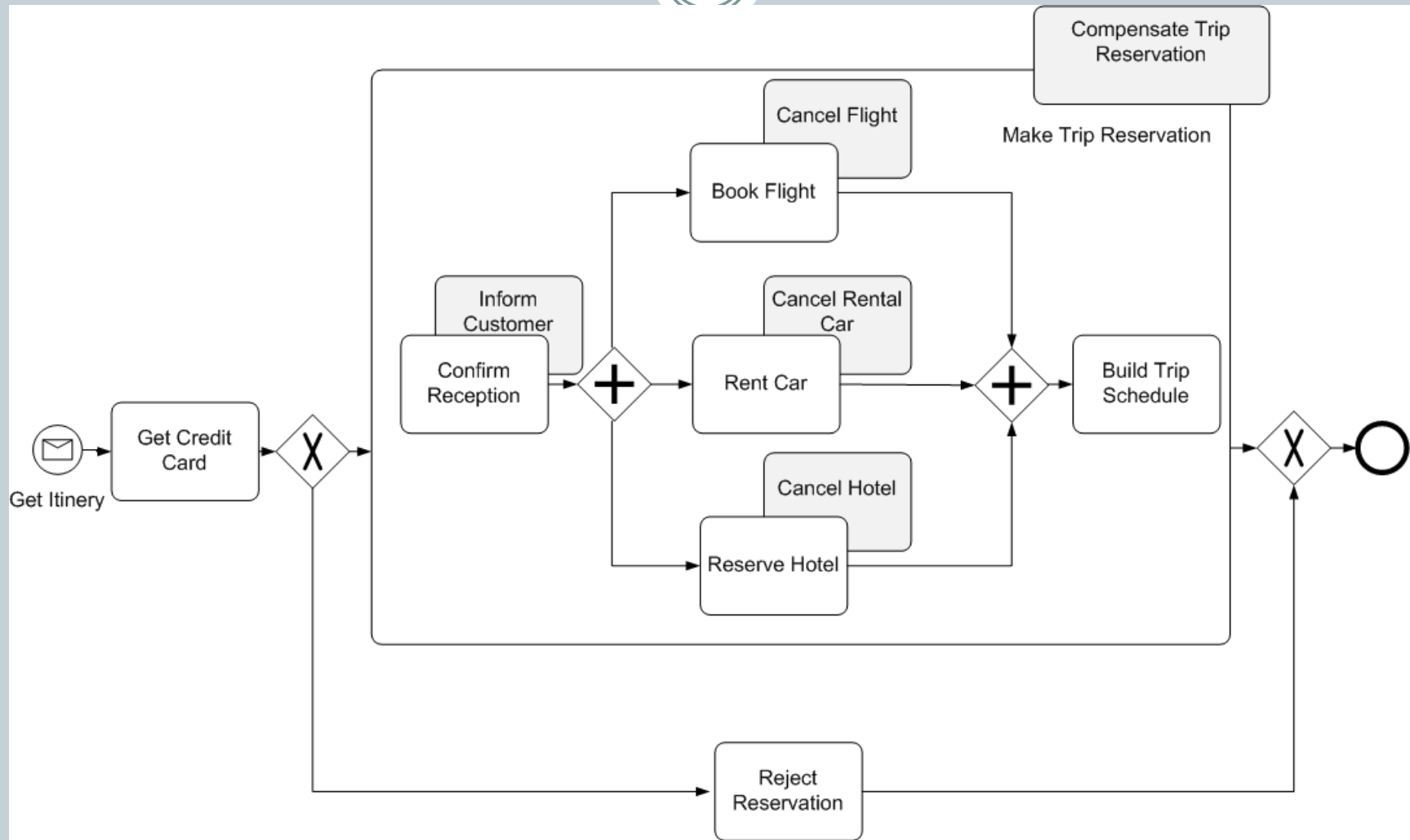
The SAGAS concept



Compensation Spheres



Compensation Spheres



Compensation and Fault Handling in BPEL (1)

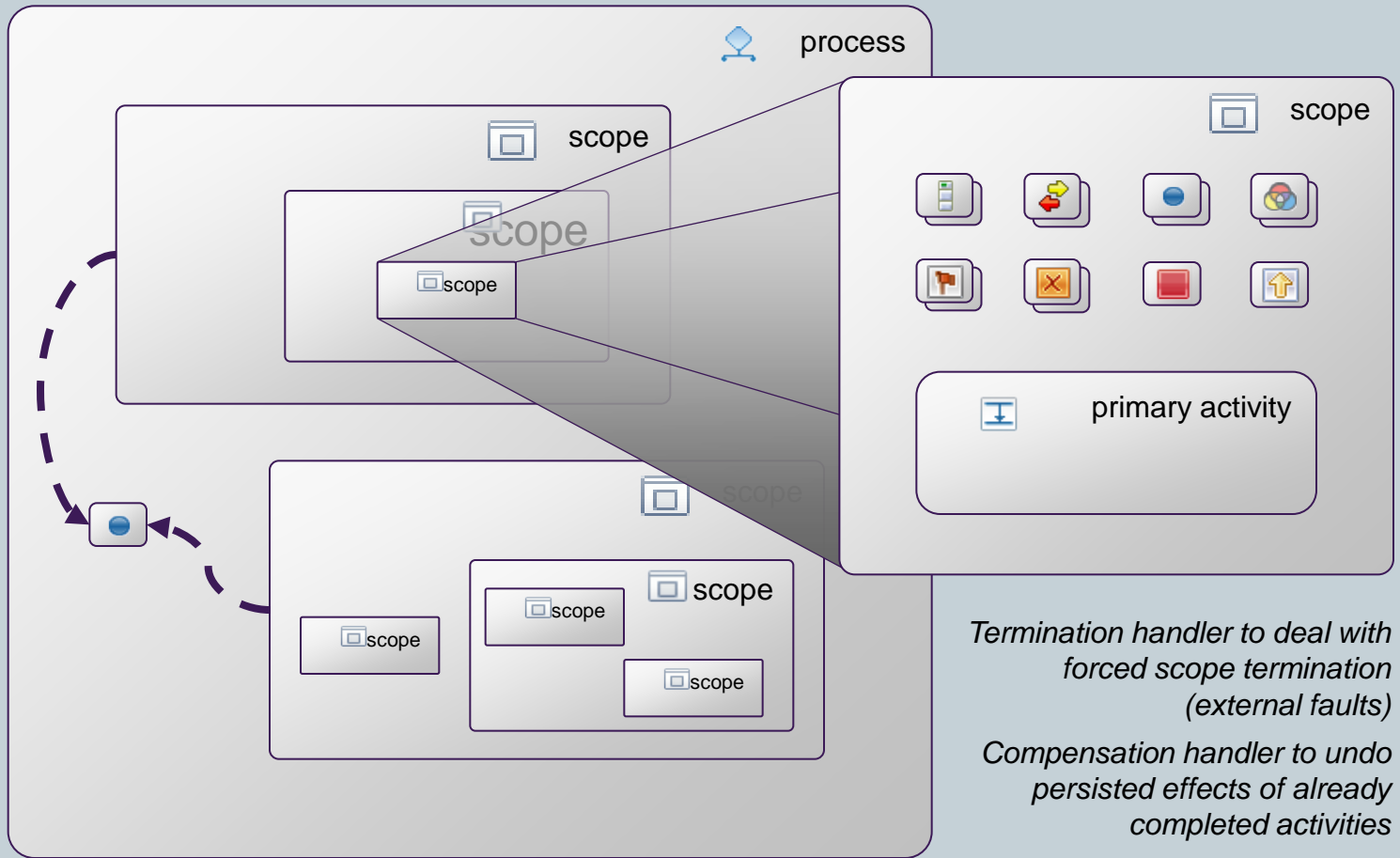
52

Scopes provide a context which influences the execution behavior of its enclosed activities

Local declarations: partner links, message exchanges, variables, correlation sets

Local handlers: event handlers, fault handlers, a termination handler, and a compensation handler

Isolated scopes provide control of concurrent access to shared resources

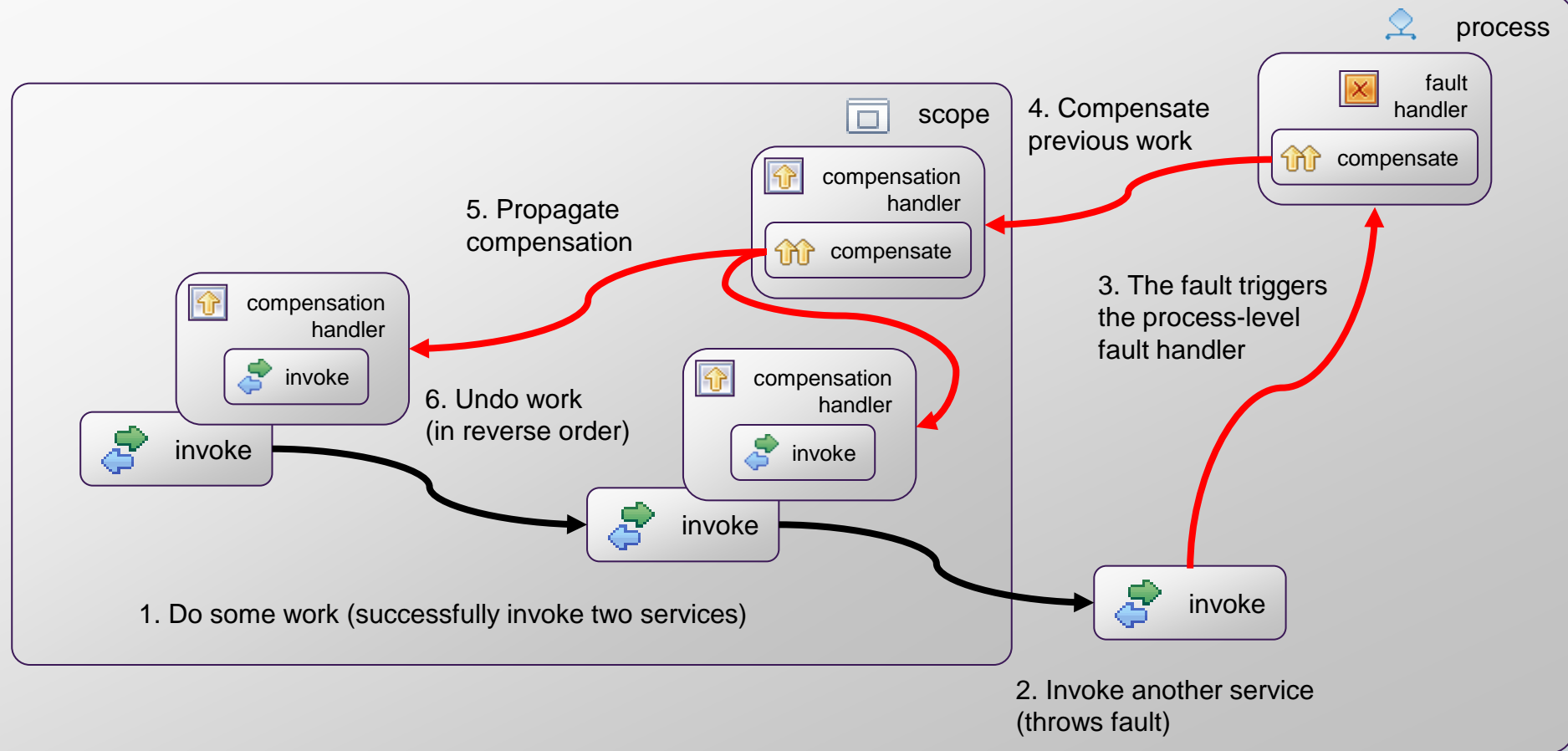


Termination handler to deal with forced scope termination (external faults)

Compensation handler to undo persisted effects of already completed activities

Compensation and Fault Handling in BPEL (2)

53



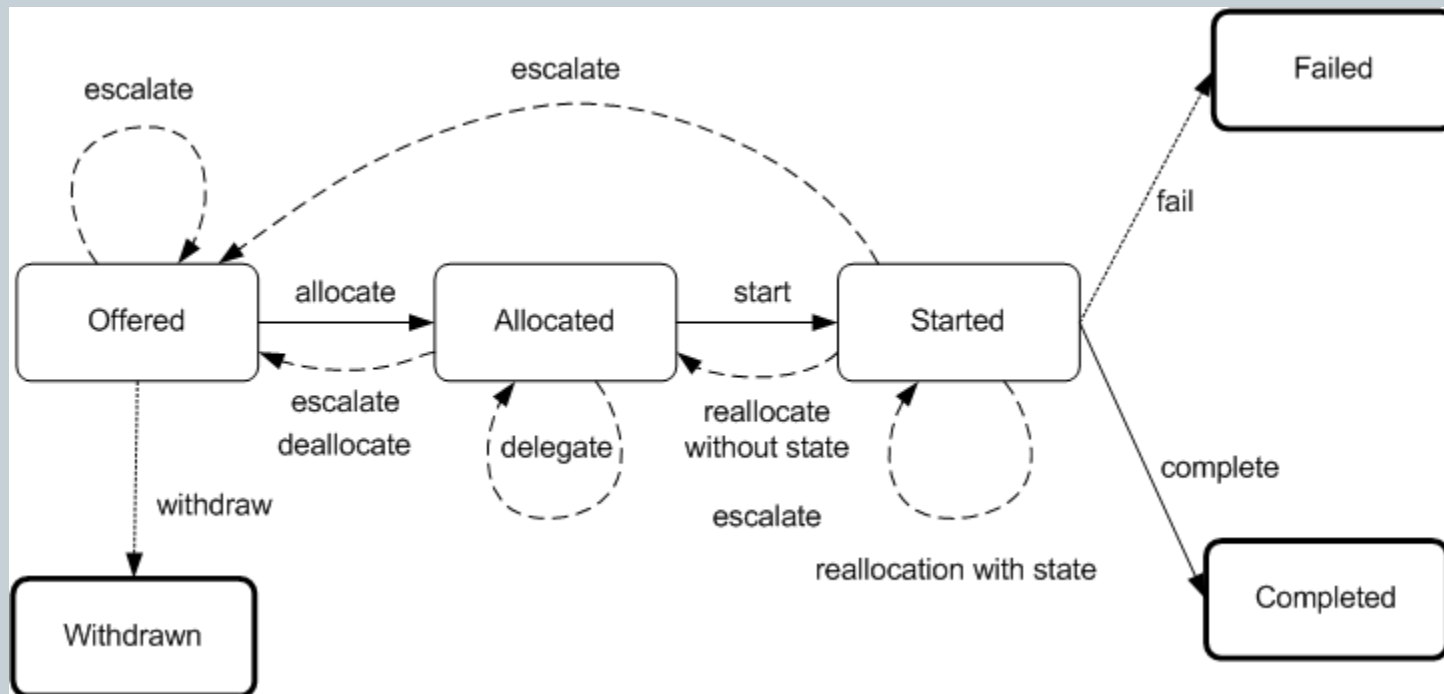
This example shows the default compensation behavior supported by BPEL; i.e., a completed scope is compensated by invoking the compensation handlers of its constituting activities in reverse order. However, a more specific compensation handler for a scope may be provided as well (e.g., only compensating some of the already completed activities or invoking a specific process dealing with the exception).

Resource Patterns



- **Exception Handling Patterns (like Deferred Fixing, Reject etc.) focus on behavioral changes**
- **Many exceptions (e.g., resource unavailability or deadline expiry) require changes regarding resource perspective like *delegation, escalation or reallocation***

Resource Patterns



Selected Resource Patterns



Resource Pattern	Description	Original States	Resulting States
Delegation	A resource allocates a previously allocated work item to another resource (i.e., reallocate).	Allocated	Allocated
Escalation	The system attempts to progress a stalled work item by offering or allocating it to another resource.	Allocated, Offered, Started	Allocated, Offered
Stateful Reallocation	A resource allocates a work item it has started to another resource and the current state of the work item is retained.	Started	Started
Stateless Reallocation	A resource allocates a work item it has started to another resource, but the current state is not retained (i.e. the work item is restarted).	Started	Allocated
Deallocation	A resource makes a previously allocated work item available, i.e., the work item can be offered to other resources.	Allocated	Offered

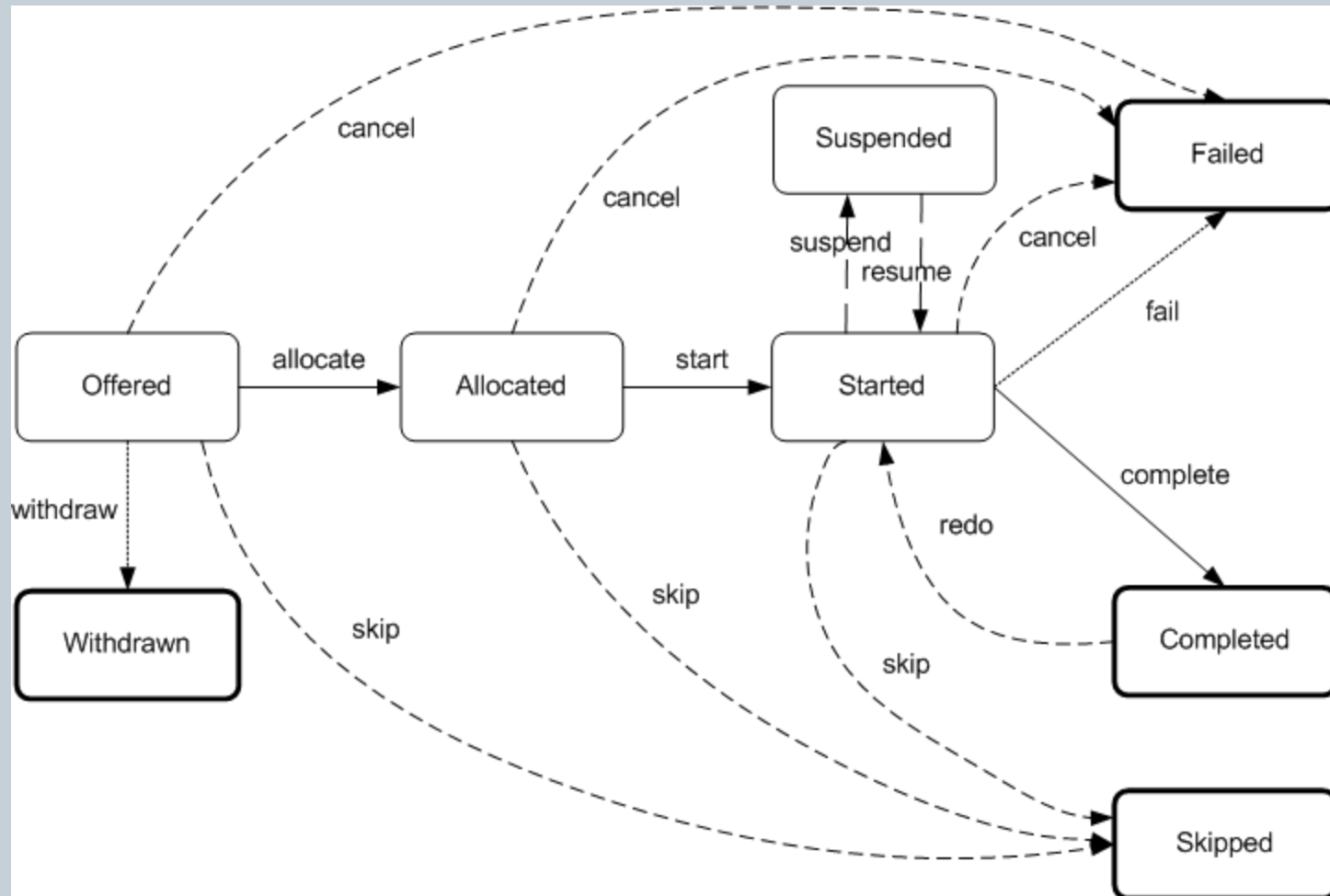
For more details visit: <http://www.workflowpatterns.org>

Flexible Handling of Work Items



- Application of Exception Handling patterns often requires changes to the lifecycle of work items.
- Work items may have to be
 - *Skipped*
 - *Redone*
 - *Done ahead of time*
 - *Canceled*
 - *Suspended/Resumed*

Flexible Handling of Work items



Flexible Handling of Workitems



Resource Pattern	Description	Original States	Resulting States
Suspension-Resumption	A resource temporarily suspends the execution of a work item or recommences the execution of a previously suspended work item.	Started Suspended	Suspended Started
Skipping	A resource skips the execution of an offered, allocated or started work item.	Offered Allocated Started	Skipped
Redo	A resource re-executes a work item already completed earlier.	Completed	Started
Predo	A resource executes an activity not yet enabled (and therefore not yet offered) in the context of a particular process instance; the activity is executed ahead of time.	-	-
Cancel	A work item is aborted and changes to state failed.	Offered Allocated Started	Failed

For more details visit: <http://www.workflowpatterns.org>

Business Processes and Workflows

Handling Unforeseen Exceptions



MONTEVIDEO, DECEMBER 11TH 2012



**PRESENTED BY
BARBARA WEBER
UNIV. OF INNSBRUCK**

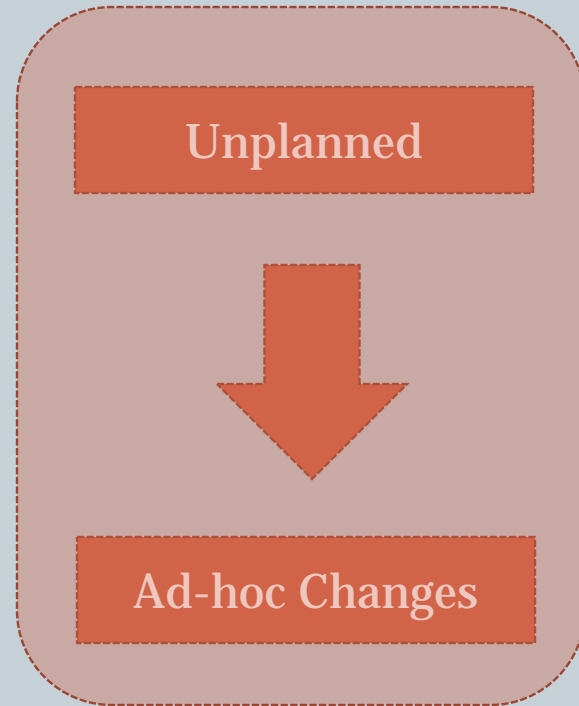
Process Adaptations



Planned

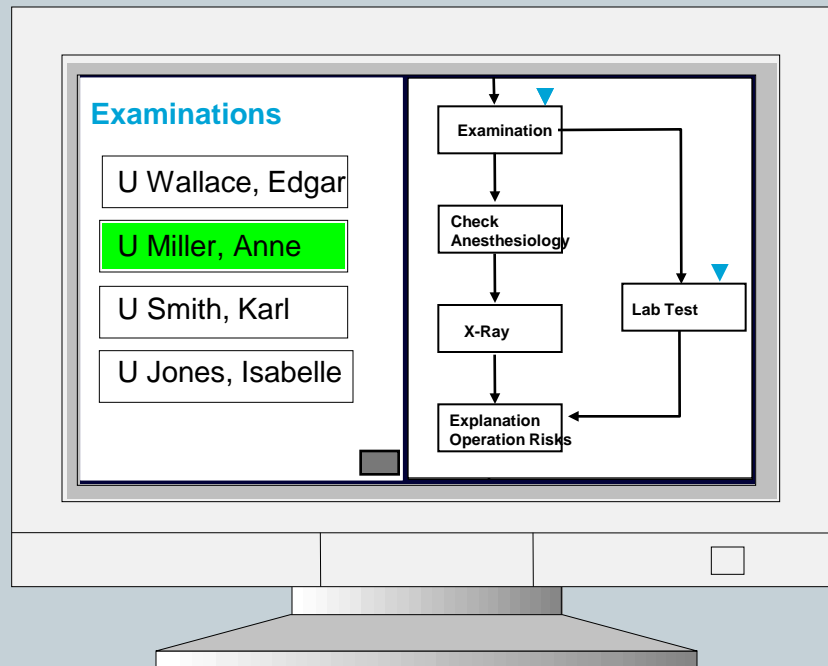


Exception
Handling

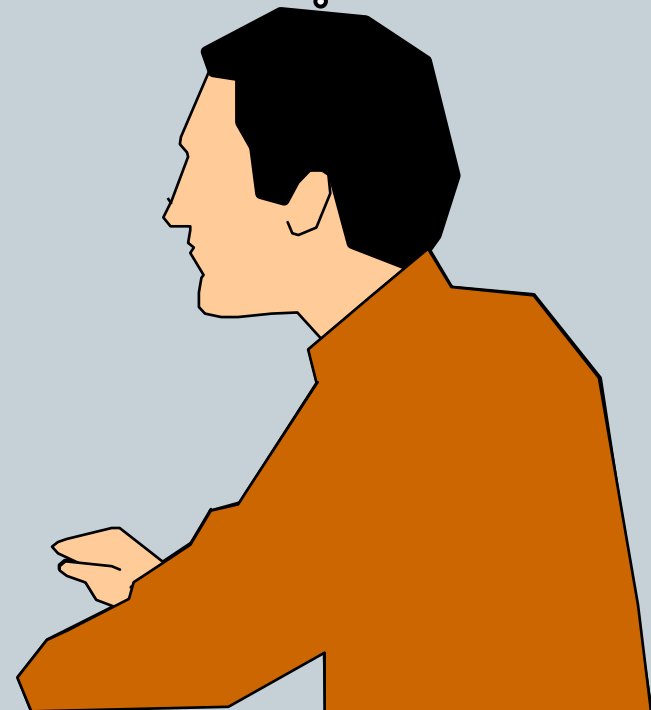


User View on an Ad-hoc Process Change

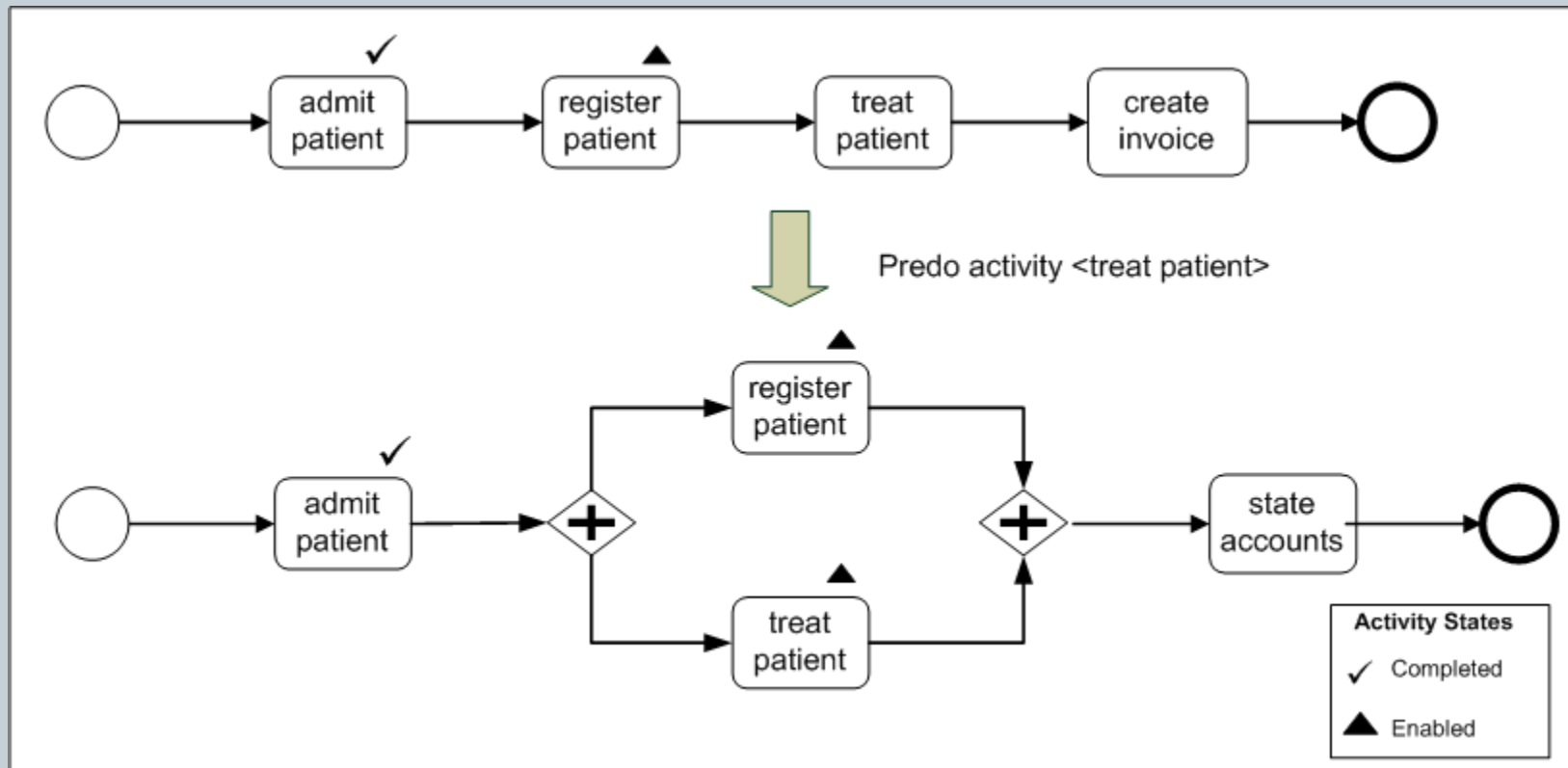
62



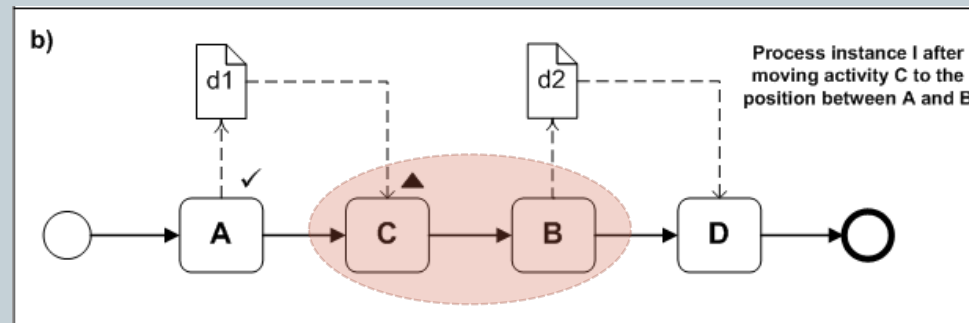
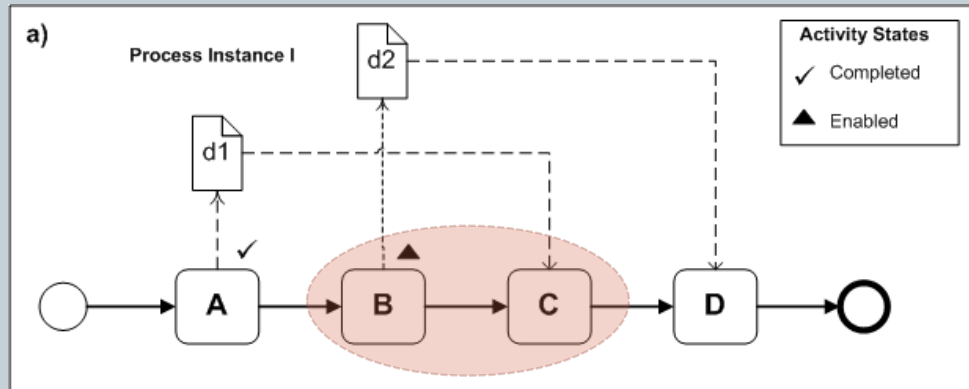
Exception –
We need an additional lab
test !



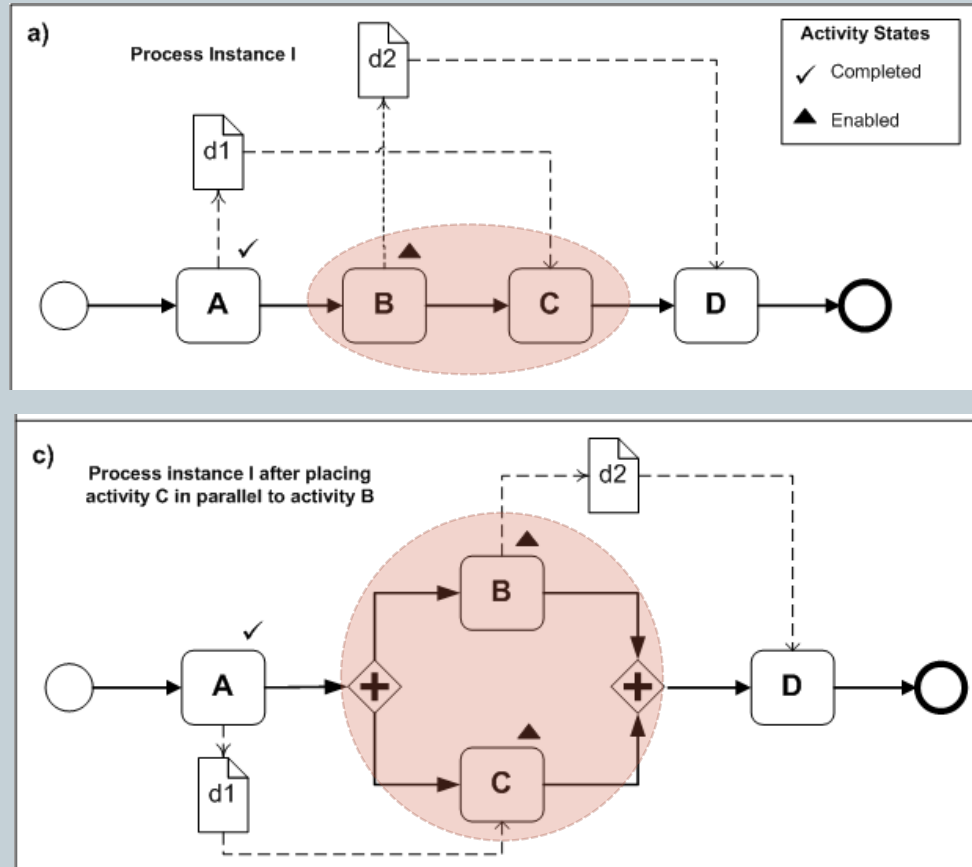
Behavioral Changes Require Structural Process Model Adaptations



Behavioral Changes Require Adaptations of the Process Instance State

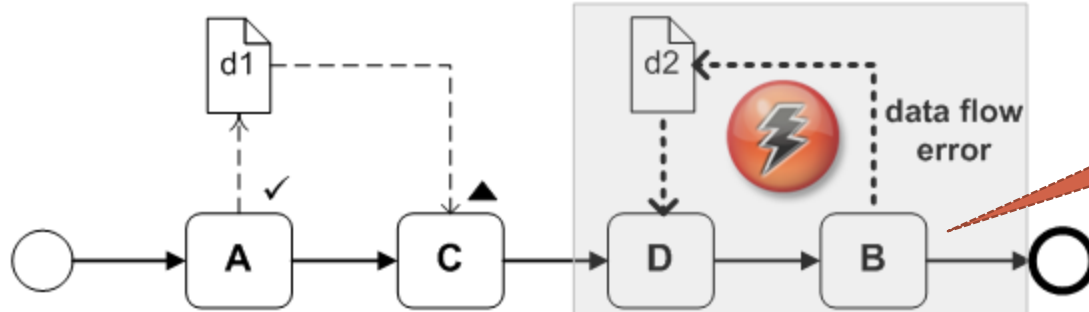


Behavioral Changes Require Adaptations of the Process Instance State



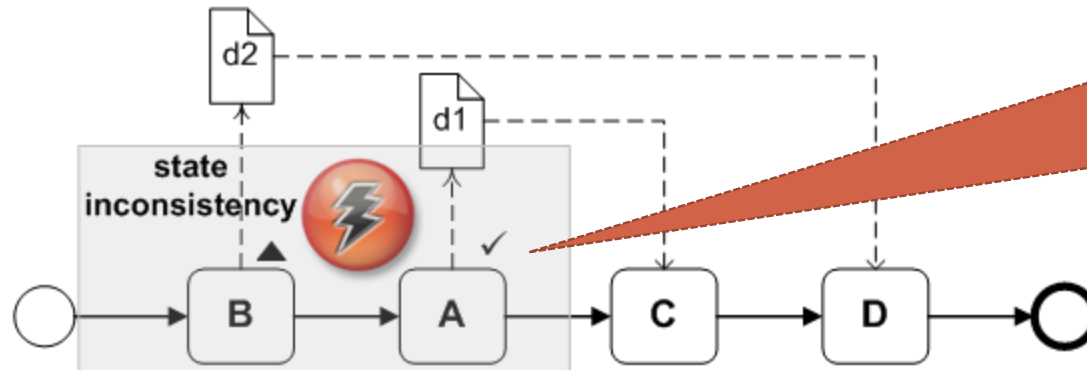
Behavioral Changes Must not Violate Process Model Soundness and Proper Instance Execution

a) Process instance I after moving activity B to the position after D



Data flow error caused by missing data

b) Process instance I after moving activity B to the position before A



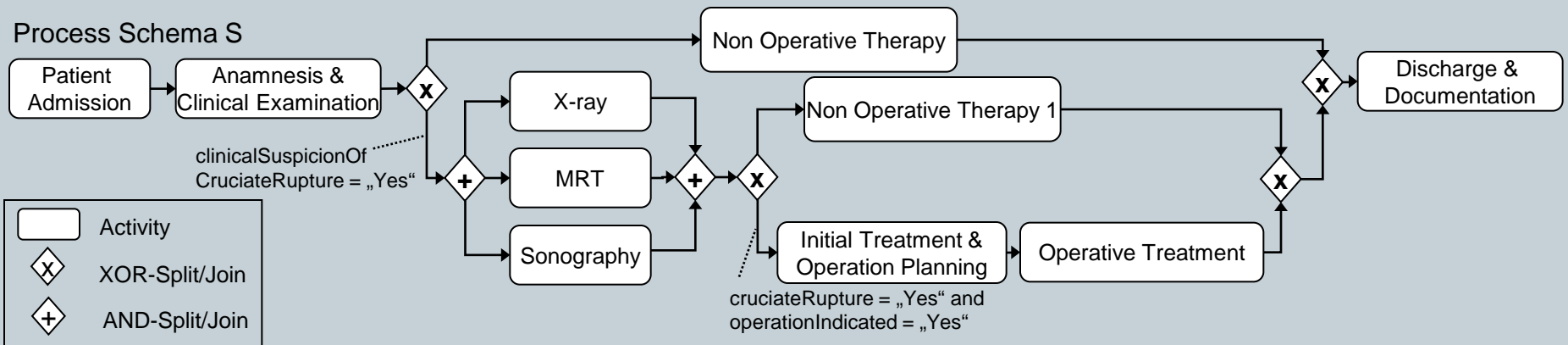
No Proper Completion ensured. End node can be reached since B is still enabled

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

67

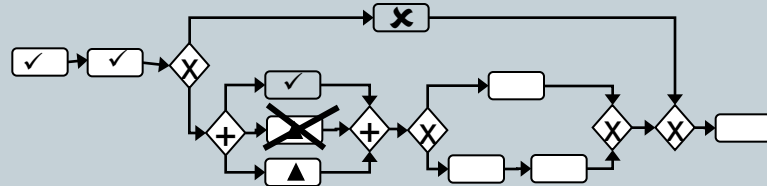
Process Type Level

Process Schema S



Process Instance Level

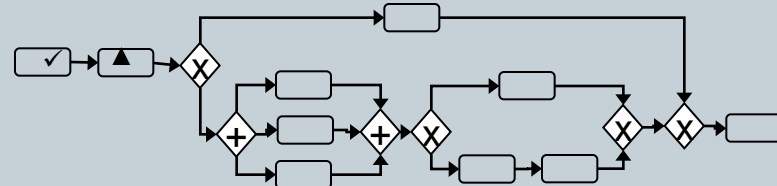
Process Instance I1



Execution Trace:

$\sigma_1 = \langle \text{„Patient Admission“}, \text{„Anamnesis & Clinical Examination“}, \text{„X-ray“} \rangle$

Process Instance I2



Execution Trace:

$\sigma_2 = \langle \text{„Patient Admission“} \rangle$

Structurally Adapting Pre-Specified Process Models



Change Primitives

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

High-Level Change Operations

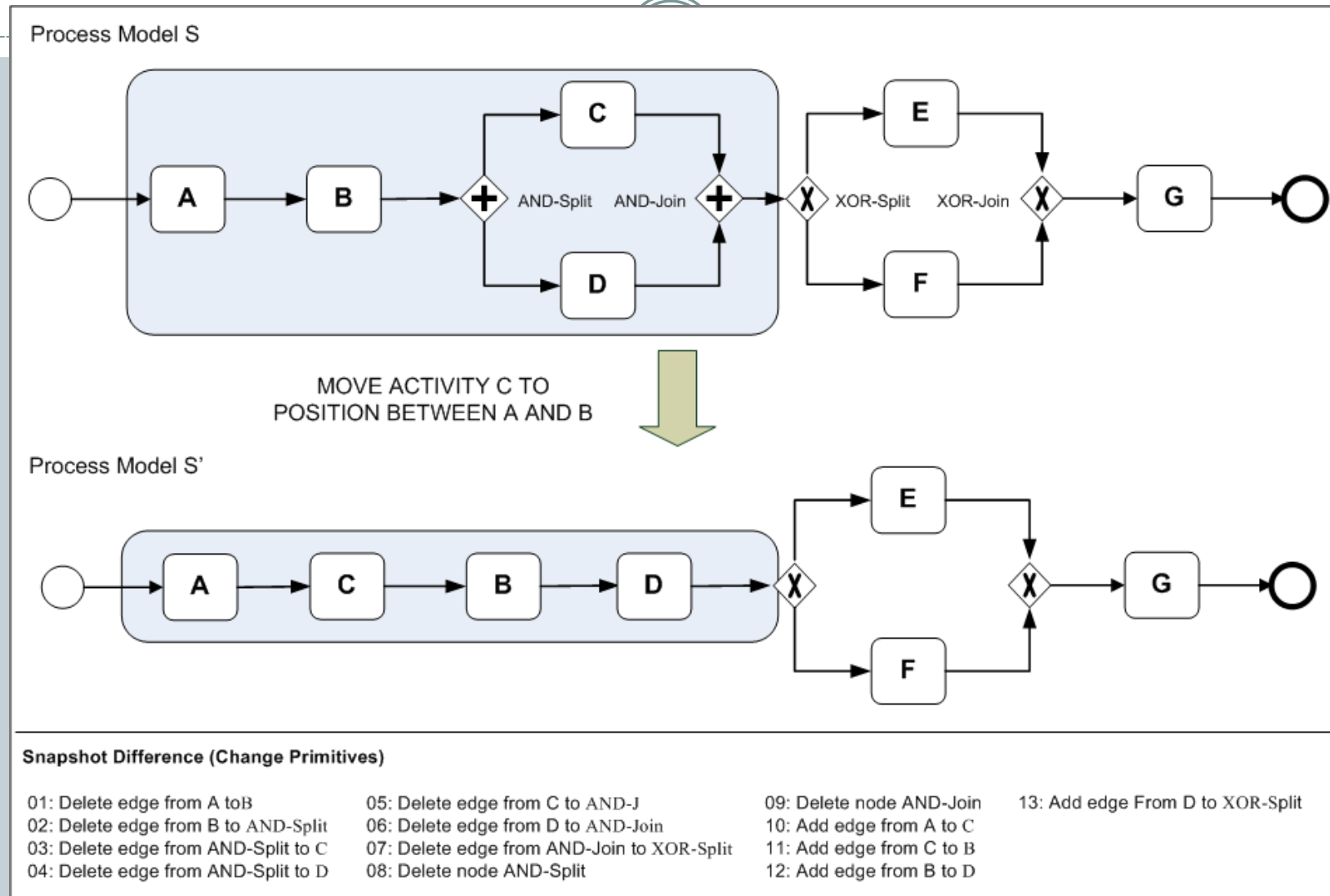
- Combines a set of change primitives
- Referred to as Adaptation Patterns in the following

Adaptation Patterns



Adding / Deleting Process Fragments	AP1:	Insert Process Fragment
	AP2:	Delete Process 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

Adaptation Patterns versus Change Primitives

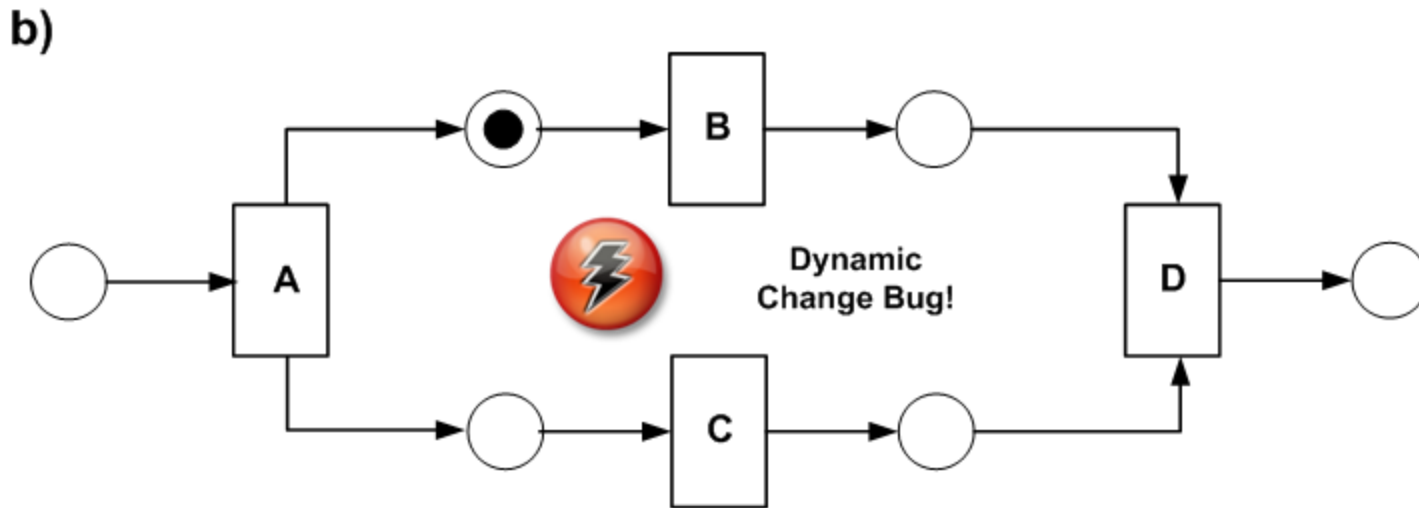
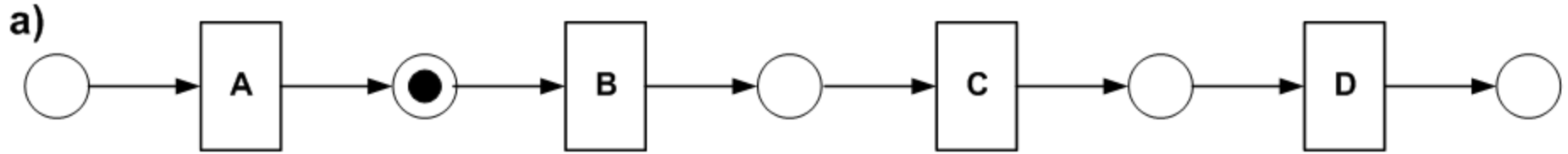


Adaptation Patterns versus Change Primitives



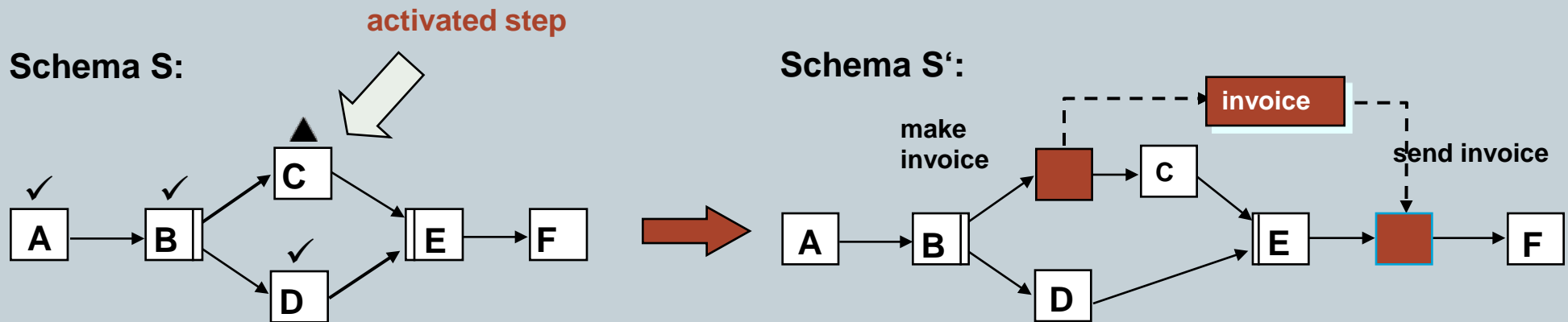
Change Primitives	Process Adaptation Patterns
Operate on single elements of process schema	Provide high-level change operations
Correctness has to be checked after adaptation	Correctness-by-construction
No Assumption regarding structure of process schema	Process schema needs to be block-structured

Dynamic Change Bug



Correctness of Process Instance Changes

Ensuring Dynamic Correctness



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

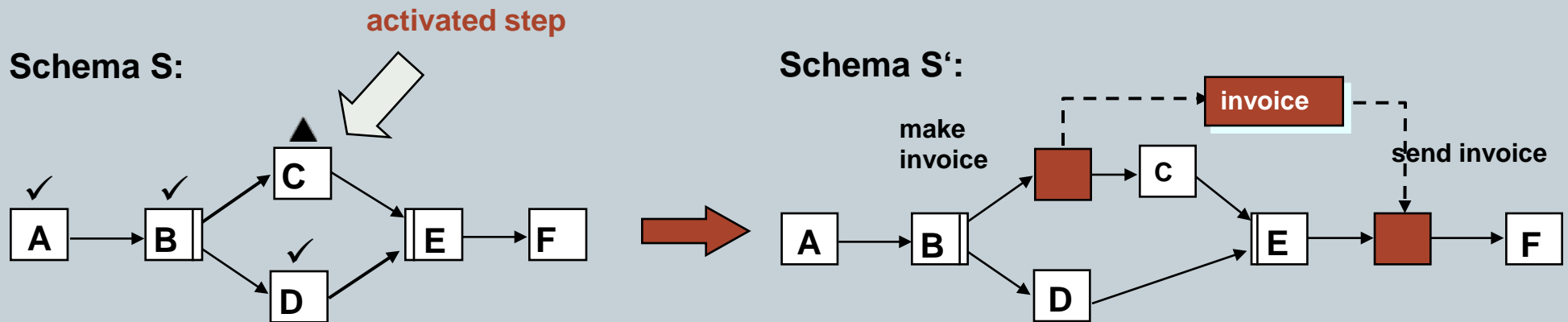
Need for general correctness criterion

⇒ State Compliance

[ReDa98, RRW08a, RRD04a, RRD04b]

Correctness of Process Instance Changes

Ensuring Dynamic Correctness



$\langle A \rangle, \langle B \rangle, \langle D \rangle \Rightarrow$ Trace reproducible on new schema?

More complicated: loop backs

Further challenges:

- How to efficiently check for compliance?
- How to efficiently migrate process instances?

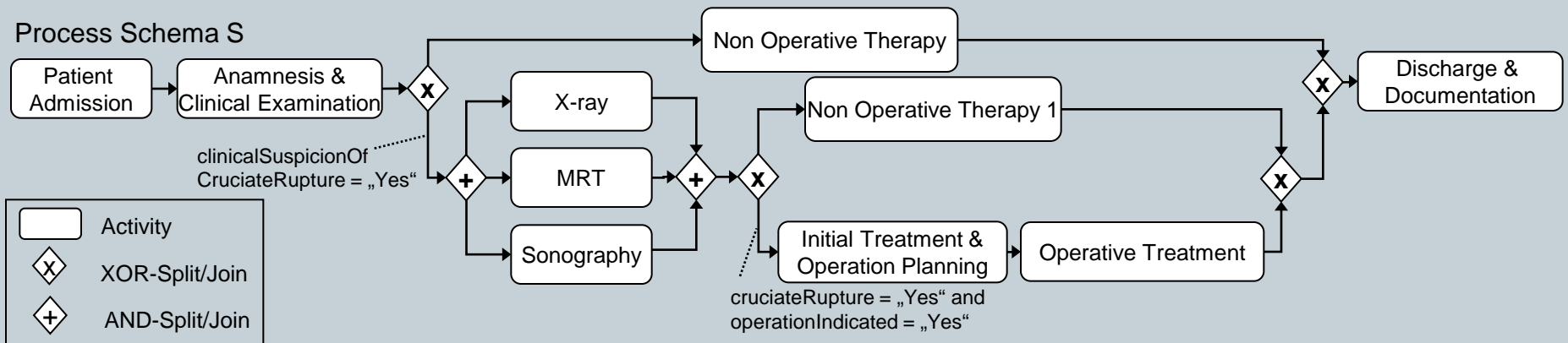
[RRD04a, RRD04b]

Correctness of Process Instance Changes

75

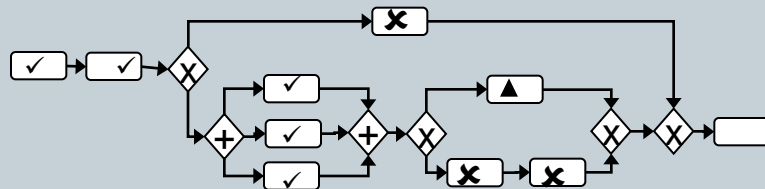
Process Type Level

Process Schema S



Process Instance Level

Process Instance I3



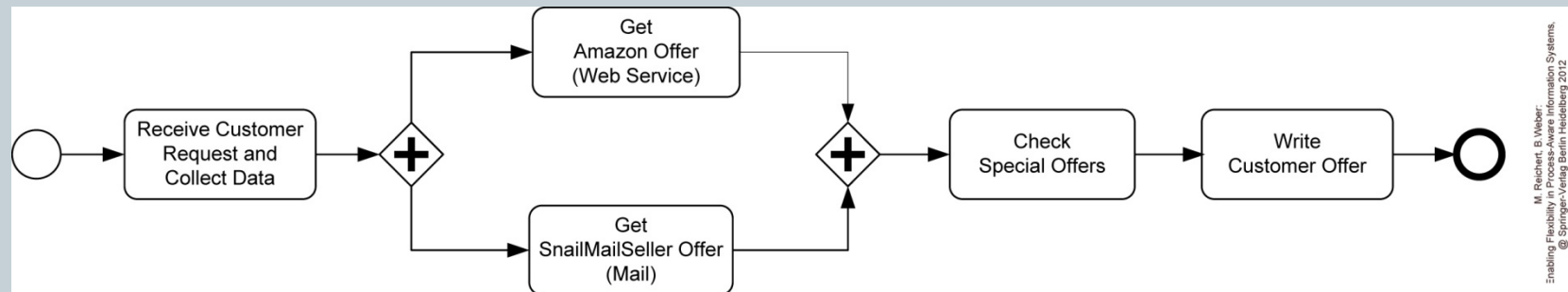
Execution Trace:

$\sigma_3 = \langle \text{„Patient Admission“}, \text{„Anamnesis & Clinical Examination“}, \text{„MRT“}, \text{„X-ray“}, \text{„Sonography“} \rangle$

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

Handling Planned and Unplanned Exceptions with AristaFlow BPM Suite

- Example: Order Handling Process



Handling Planned and Unplanned Exceptions with AristaFlow BPM Suite

The screenshot displays the AristaFlow Process Template Editor interface. The main workspace shows a BPM diagram with a flow starting at 'Start', leading to a decision node 'User Signalled Fail?'. If 'no', the flow goes to an activity node 'i Activity '%s:Node Name%' Failed', which is highlighted in green. If 'yes', the flow goes to a connector node '#3' and then to 'End'. A table at the top lists input parameters for the process template:

Template ID	Iteration	Is Activity	Executing Agent ID	Executing Org Position ID	Node Staff Assignment Rule	D
STRING	INTEGER	BOOLEAN	INTEGER	INTEGER	STRING	UI

On the right, the 'Data Elements' panel lists 19 data elements:

Name	Type
Data Context	USERDEFINED
EBP Type	INTEGER
Error Code	INTEGER
Error Message	STRING
Error State	STRING
Executing Agent	STRING
Executing Org	STRING
Instance ID	STRING
Iteration	INTEGER
Node ID	INTEGER
Node Name	STRING
Node Staff Assignment Rule	STRING
Template ID	STRING
Template Name	STRING

At the bottom, the 'Properties' panel shows the 'Node Basics' for the selected activity:

Node Basics

Name: Activity '%s:Node Name%' Failed [Edit...]

Description: An activity in the system failed, the failed state was not explicitly triggered by a user abort. [Edit...]

Staff Assignment: orgposition(name='supervisor') [Edit...]

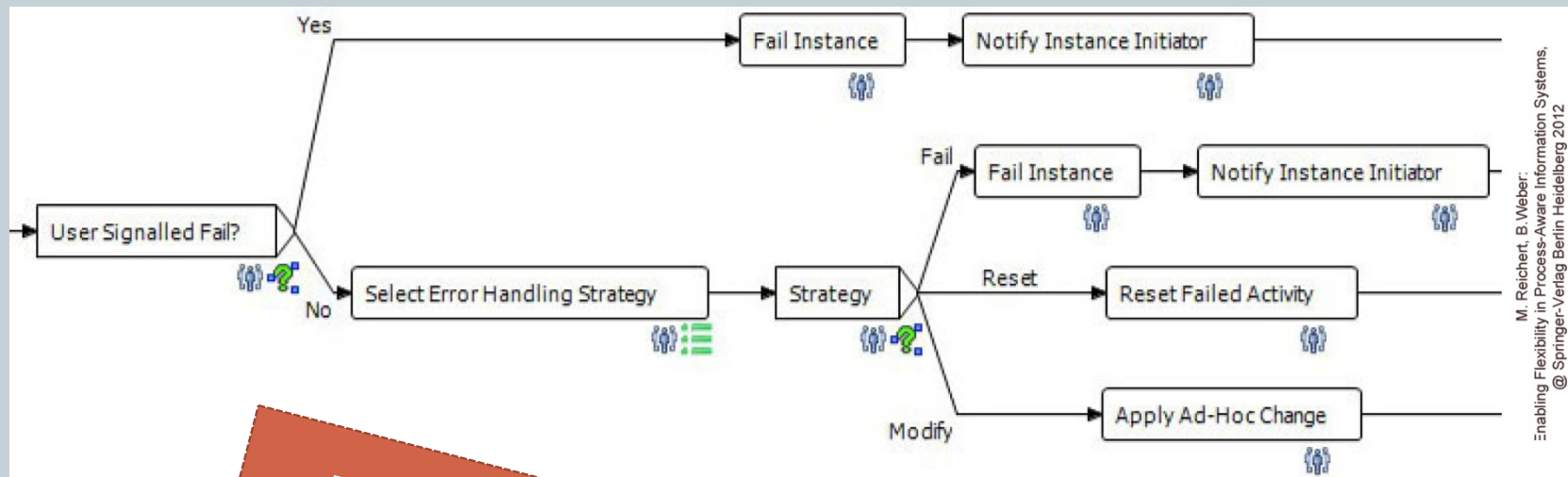
Input parameters of the process template

Properties of the selected activity

User assigned to the selected activity

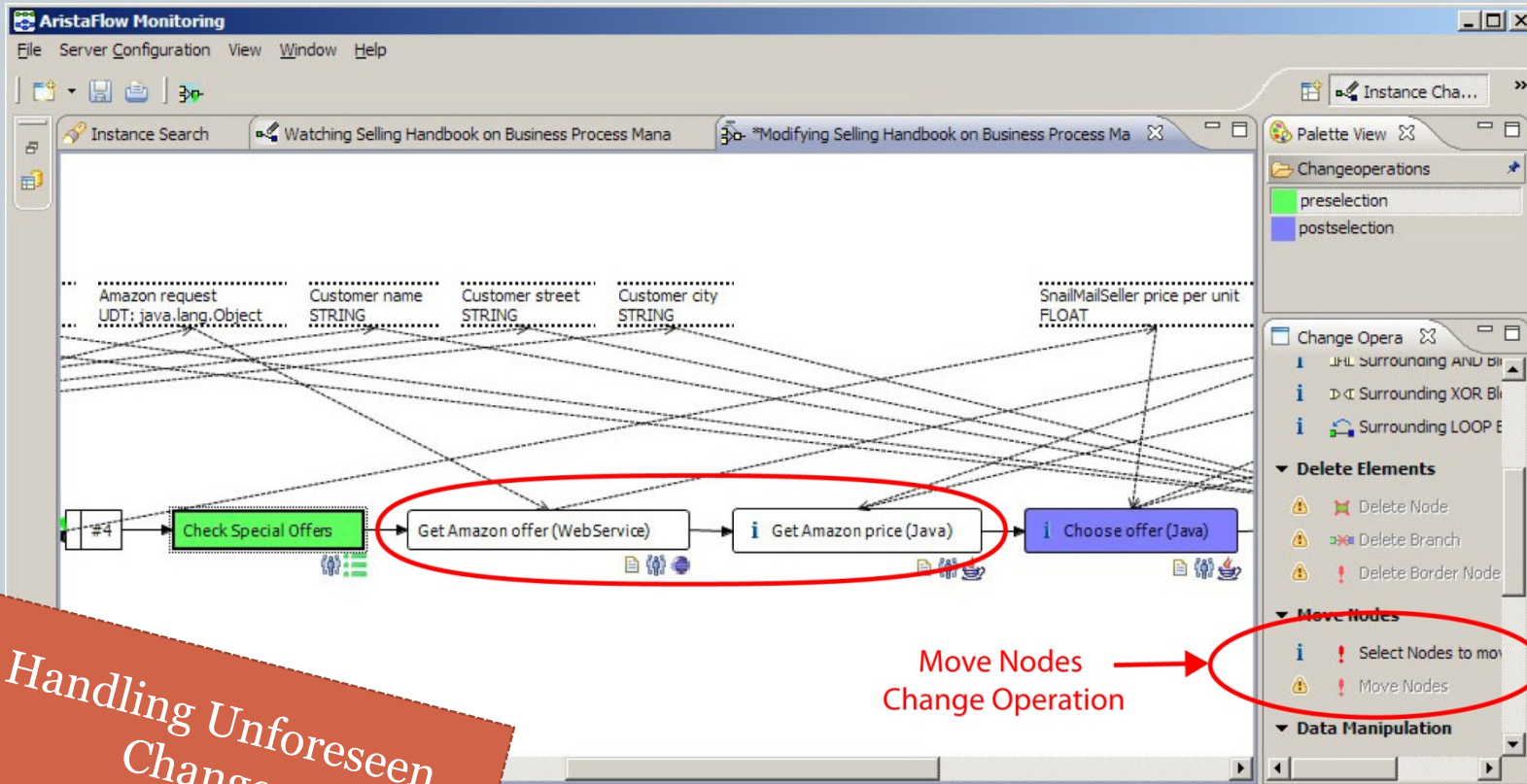
Handling a Simple Exception

Handling Planned and Unplanned Exceptions with AristaFlow BPM Suite



Example of a more complex exception

Handling Planned and Unplanned Exceptions with AristaFlow BPM Suite



Handling Unforeseen Changes

Move Nodes
Change Operation

Business Processes and Workflows

Process Monitoring, Analysis and Mining



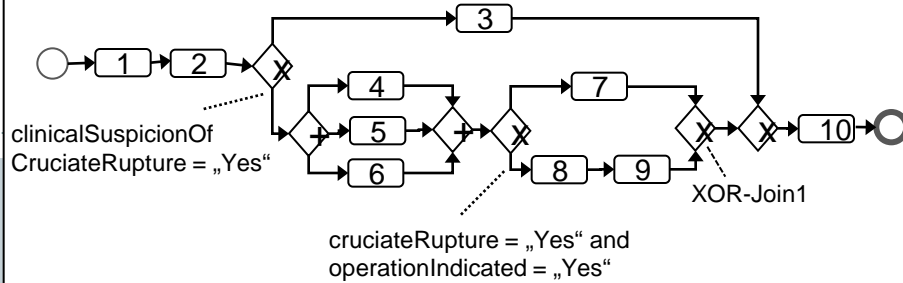
MONTEVIDEO, DECEMBER 11TH 2012



PRESENTED BY
BARBARA WEBER
UNIV. OF INNSBRUCK

A) Process Model

Process Model S



- 1: Patient Admission
 2: Anamnesis & Clinical Examination
 3: Non Operative Therapy
 4: X-Ray
 5: MRT
 6: Sonography
 7: Non Operative Therapy 1
 8: Initial Treatment & Operation Planning
 9: Operative Therapy
 10: Discharge & Documentation

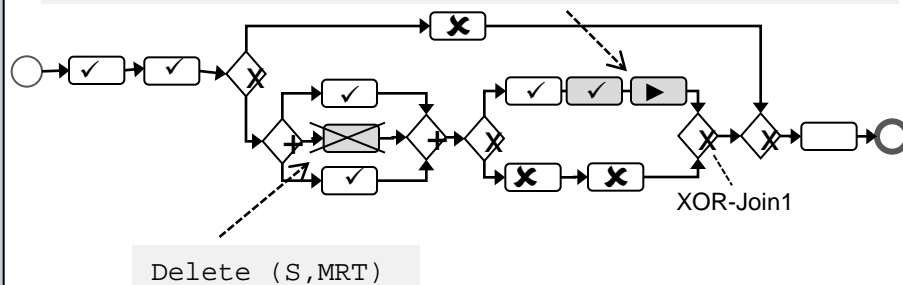
C) Execution Log Entries of Process Instance 4711

Activity	Event	User	Timestamp
Patient Admission	Start	Garry	2007/09/08 15:30
Patient Admission	Complete	Garry	2007/09/08 15:45
Anamnesis & Clinical Examination	Start	Helen	2007/09/09 11:00
Anamnesis & Clinical Examination	Complete	Helen	2007/09/09 11:45
X-Ray	Start	Paula	2007/09/09 12:34
Sonography	Start	Sandy	2007/09/09 13:20
X-Ray	Complete	Paula	2007/09/09 14:00
Sonography	Complete	Sandy	2007/09/09 14:30
Non Operative Therapy 1	Start	Peter	2007/09/10 09:10
Non Operative Therapy 1	Complete	Peter	2007/09/10 09:45
Follow-up Examination	Start	Helen	2007/09/12 11:07
Follow-up Examination	Complete	Helen	2007/09/12 11:20
Puncture	Start	Helen	2007/09/12 11:21

B) Process Instances

Process Instance 4711 on S

Insert(S, Follow-up Examination, Non Operative Therapy, XOR-Join 1)
 Insert(S, Puncture, Follow-up Examination, XOR-Join 1)



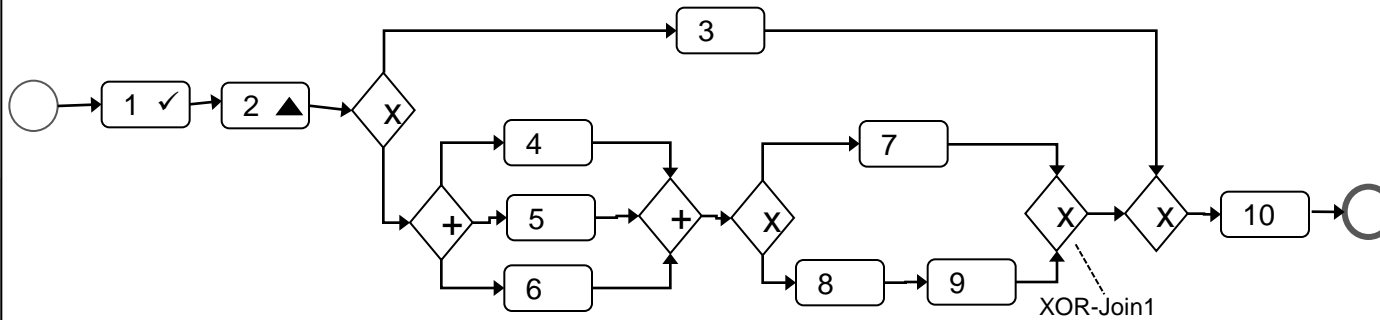
D) Change Log Entries of Process Instance 4711

Change TX	Applied Changes	User	Timestamp
001	Delete (S, MRT)	Paula	2007/09/09 12:50
002	Insert(S, Follow-up Examination, Non Operative Therapy, XOR-Join 1)	Helen	2007/09/10 09:00
002	Insert(S, Puncture, Follow-up Examination, XOR-Join 1)	Helen	2007/09/10 09:00

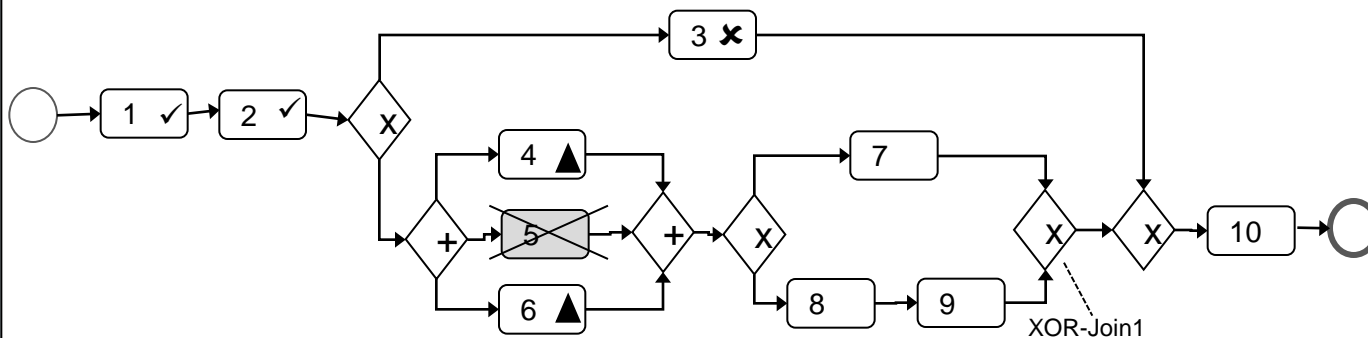
Activity States: ▶ Running ✓ Completed ✗ Skipped

Restoring Structure and State from Execution and Change Log

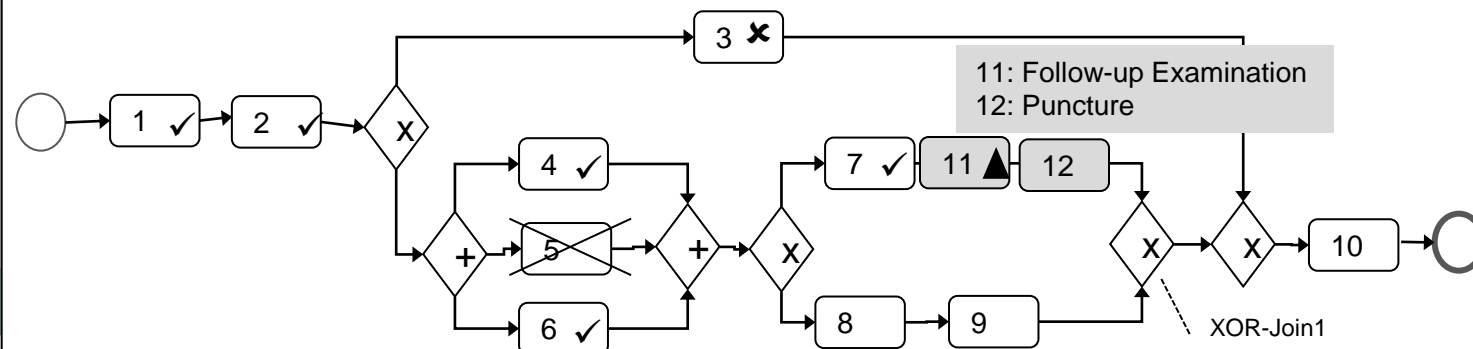
Process Instance 4711 2007/09/09 10:30



Process Instance 4711 2007/09/09 12:51



Process Instance 4711 2007/09/10 09:46



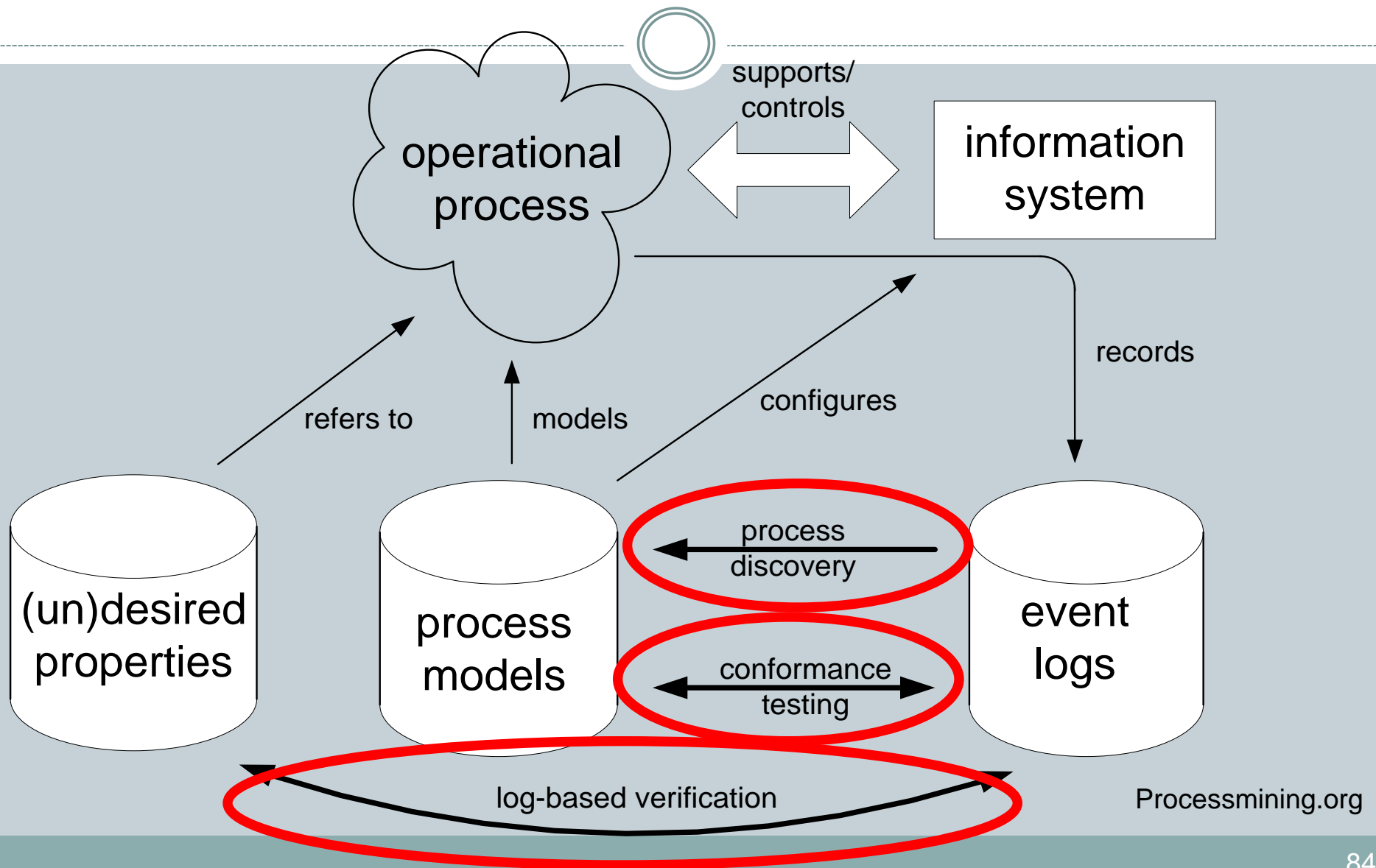
Mining Execution Logs



- **Process Discovery**
 - Presumes the presence of an event log and extracts information from such a log (e.g., process model, social network)
- **Conformance checking**
 - Analyzes whether or not the process instances in the log follow prescribed behavior of rules
- **Extension algorithms**
 - Enhance the process model based on information from the execution log (e.g., decision mining)

For more details see processmining.org

What can Process Mining be used for?

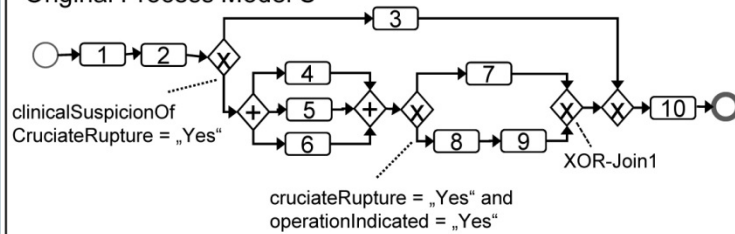


Process Discovery using Heuristic Miner



A) Original Process Model

Original Process Model S

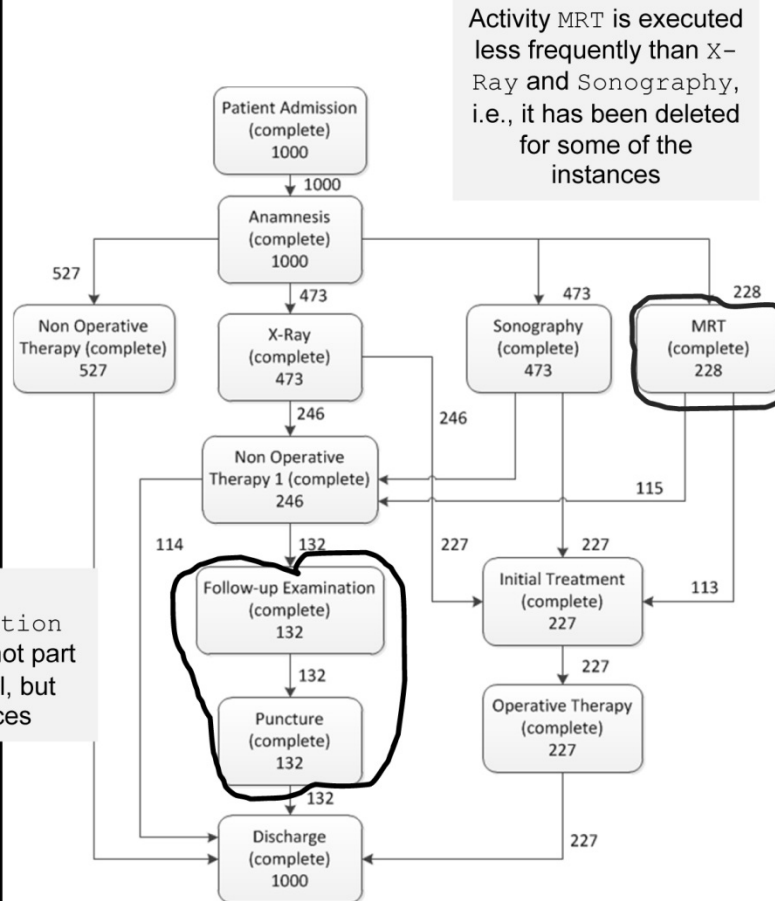


- 1: Patient Admission
- 2: Anamnesis & Clinical Examination
- 3: Non Operative Therapy
- 4: x-Ray
- 5: MRT
- 6: Sonography
- 7: Non Operative Therapy 1
- 8: Initial Treatment & Operation Planning
- 9: Operative Therapy
- 10: Discharge & Documentation

Activities

FollowUpExamination and Puncture are not part of the original model, but appear in 132 traces

B) Discovered Process Model



Discovering how the process was really executed

M. Reichert, B.Weber: Enabling Flexibility in Process-Aware Information Systems, @ Springer-Verlag Berlin Heidelberg 2012

Process Discovery using Heuristic Miner

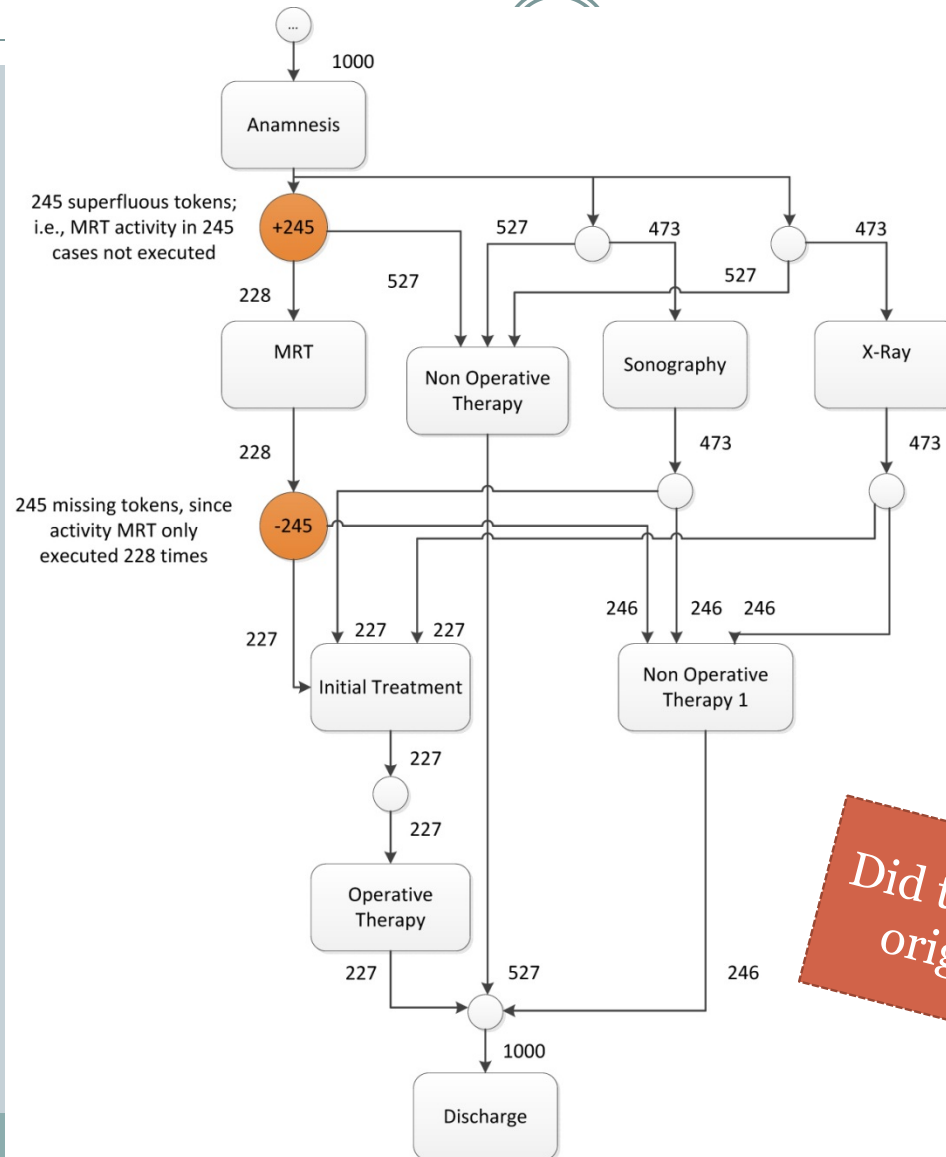


Pattern 0	Patient Adm. → Anamnesis → Non Op. Therapy → Discharge	527
Pattern 1	Patient Adm. → Anamnesis → X-Ray → Initial Treatment → Operative Therapy → Discharge ↳ Sono. ↗	114
Pattern 2	Patient Adm. → Anamnesis → X-Ray → Initial Treatment → Operative Therapy → Discharge ↳ Sono. ↗ ↳ MRT ↗	113
Pattern 3	Patient Adm. → Anamnesis → X-Ray → Non Op. Therapy 1 → Follow-up Ex. → Puncture → Discharge ↳ Sono. ↗	71
Pattern 4	Patient Adm. → Anamnesis → X-Ray → Non Op. Therapy 1 → Follow-up Ex. → Puncture → Discharge ↳ Sono. ↗ ↳ MRT ↗	61
Pattern 5	Patient Adm. → Anamnesis → X-Ray → Non Op. Therapy 1 → Discharge ↳ Sono. ↗	
Pattern 6	Patient Adm. → Anamnesis → X-Ray → Non Op. Therapy 1 → Discharge ↳ Sono. ↗ ↳ MRT ↗	

M. Reichert, B. Weber
Enabling Flexibility in Process-Aware Information Systems,
© Springer-Verlag Berlin Heidelberg 2012

What are the most frequent execution patterns?

Conformance Checking



M. Reichert, B. Weber:
 Enabling Flexibility in Process-Aware Information Systems,
 © Springer-Verlag Berlin Heidelberg 2012

Did the execution follow the original process model?

LTL Checker

cpnTools SimulationLogDeviations.mxml (2)

Analysis - LTL Checker (8)

Select formula :
eventually_activity_A_then_B

Check formula

Check options

- Check whole process
- Check until first failure
- Check until first success
- Skip if result is known

Open LTL file...
Save LTL file
Save LTL file as...

Description :
Does activity B occur after activity A occur?
Compute if there is an activity with name A and then, eventually there is an activity with name B

Arguments:

- A of type set (*ate.WorkflowModelElement*)
- B of type set (*ate.WorkflowModelElement*)

Valuate the parameters :

A	set	FollowUpExamination
B	set	Puncture

Set values as default
Delete formula

Checking for properties in
the execution log

Business Processes and Workflows

Process Evolution



MONTEVIDEO, DECEMBER 11TH 2012

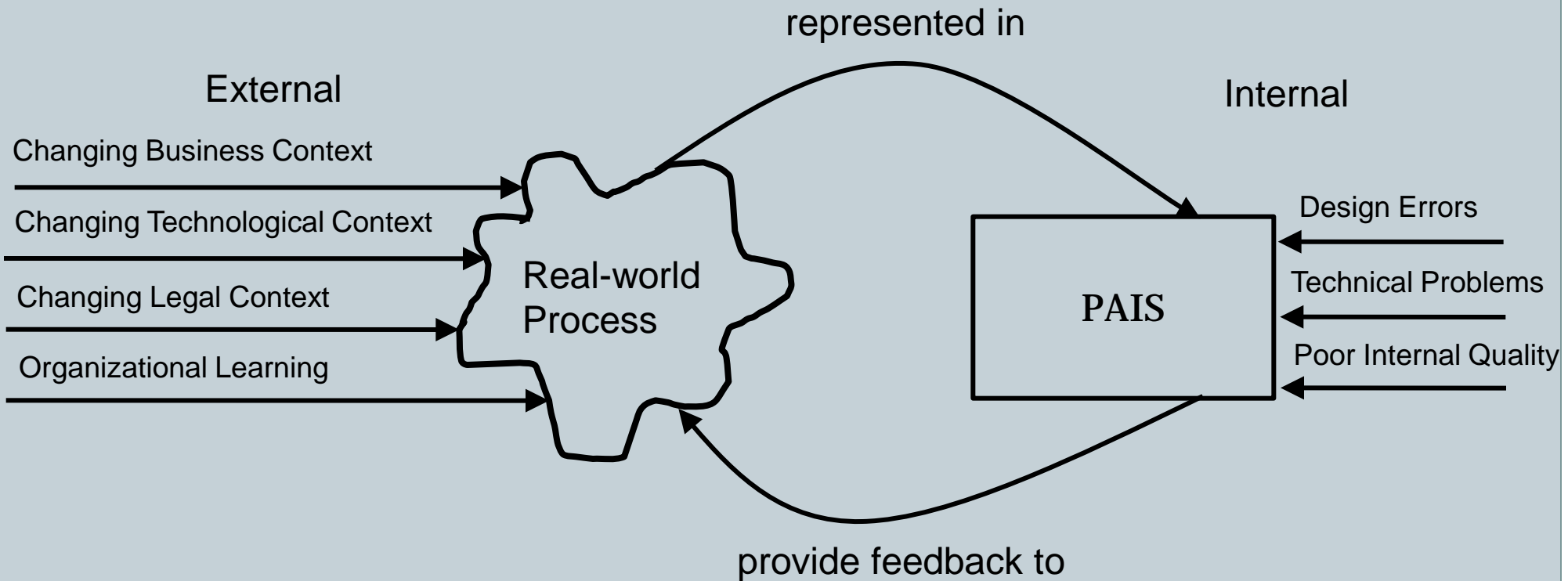


**PRESENTED BY
BARBARA WEBER
UNIV. OF INNSBRUCK**

Evolution



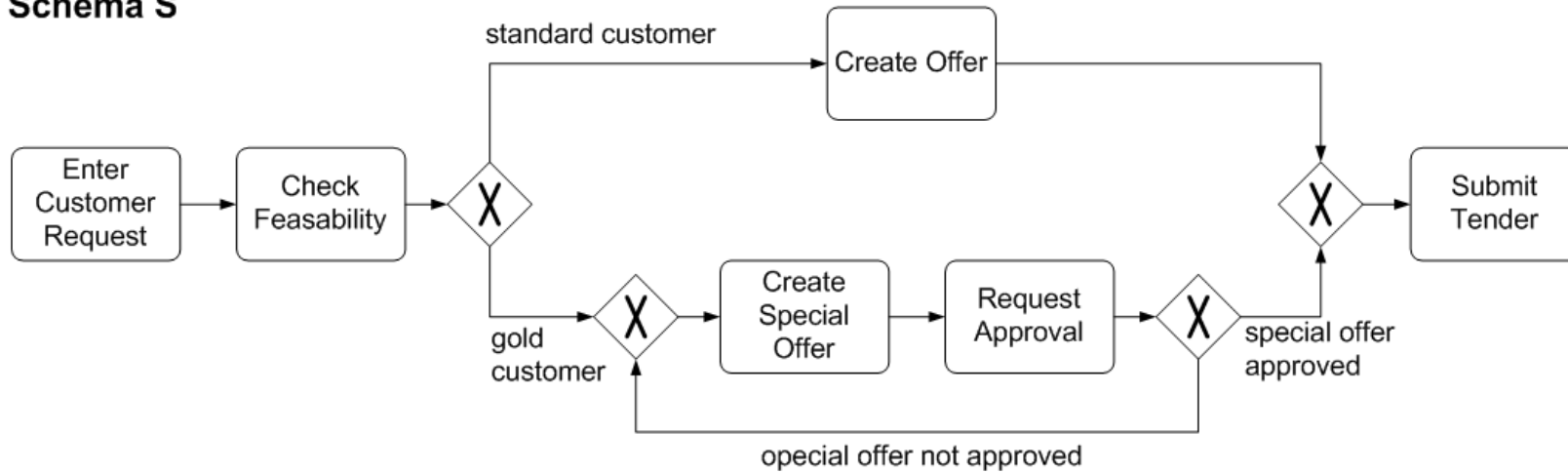
- **Drivers**



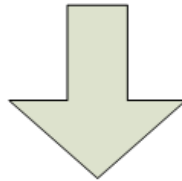
Schema Evolution

91

Schema S



S evolves to S'



by applying change Δ_S with

$\Delta_S = \langle \text{Delete}(S, \text{Create Special Offer}), \text{Delete}(S, \text{Request Approval}) \rangle$

Schema S'



Change Support Features

Schema Evolution, Version Control and Instance Migration

92

- **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

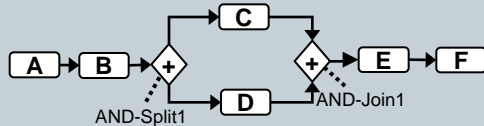
Scenario 1 - No Version Control

93

- Schema is overwritten and instances are migrated

Type change overwrites schema S

Process Schema S

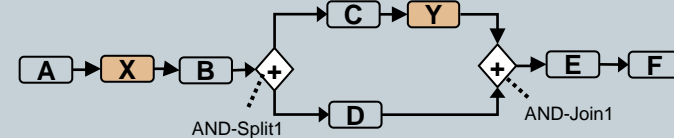


Insert X between A and B
Insert Y between C and AND-Join1

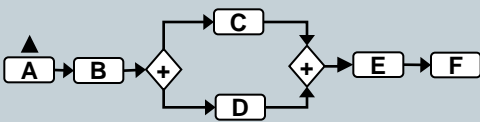


Schema Evolution

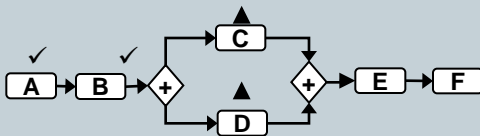
Process Schema S'



Process Instance I1

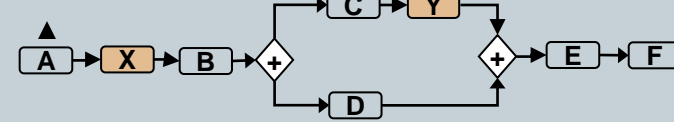


Process Instance I2

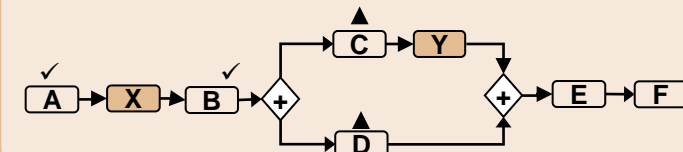


Change is propagated to all running process instances

Process Instance I1



Process Instance I2



Inconsistent state

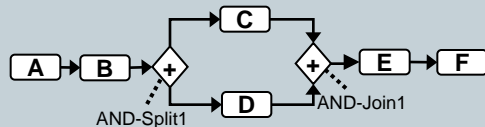
Scenario 2 - Version Control

94

- Co-existence of instances of different schema versions

Type change results into a new version of schema S

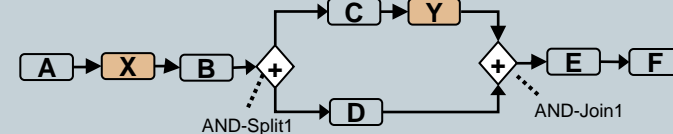
Process Schema S



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

Schema Evolution

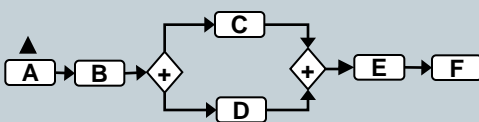
Process Schema S'



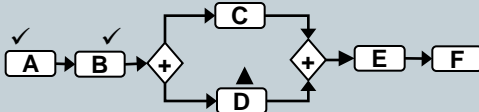
Old instances remain with schema S

Instances created from S (before schema evolution)

Process Instance I1

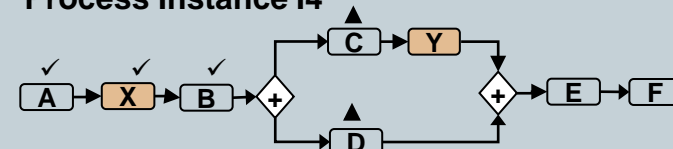


Process Instance I2

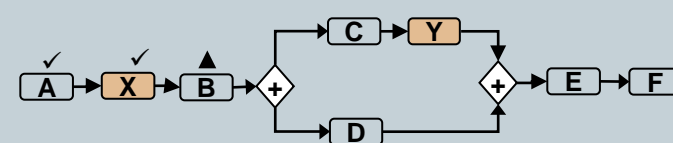


Instances created from S' (after schema evolution)

Process Instance I4



Process Instance I5



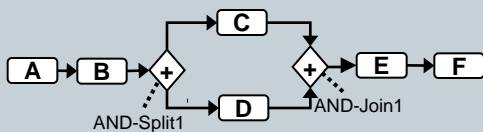
Scenario 3 – Instance Migration

95

- Compliant instances are migrated to the new schema

Type change results into a new version of schema S

Process Schema S

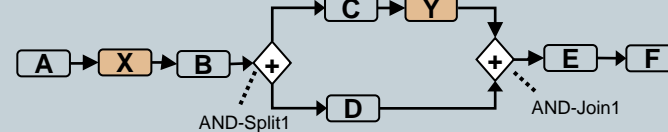


Insert X between A and B
Insert Y between C and AND-Join1



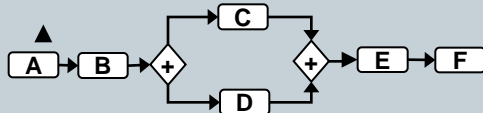
Schema Evolution

Process Schema S'



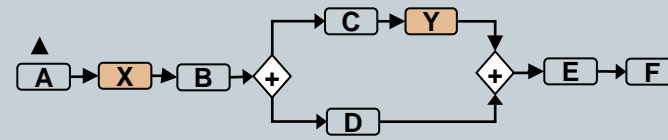
Migration of compliant process instances to S'

Process Instance I1

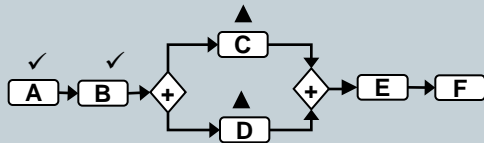


Propagation
of compliant
process instances
to schema S'
(incl. state adaptations)

Process Instance I1



Process Instance I2



Process Instance I₂ not compliant with S'

Business Processes and Workflows

Business Process Compliance



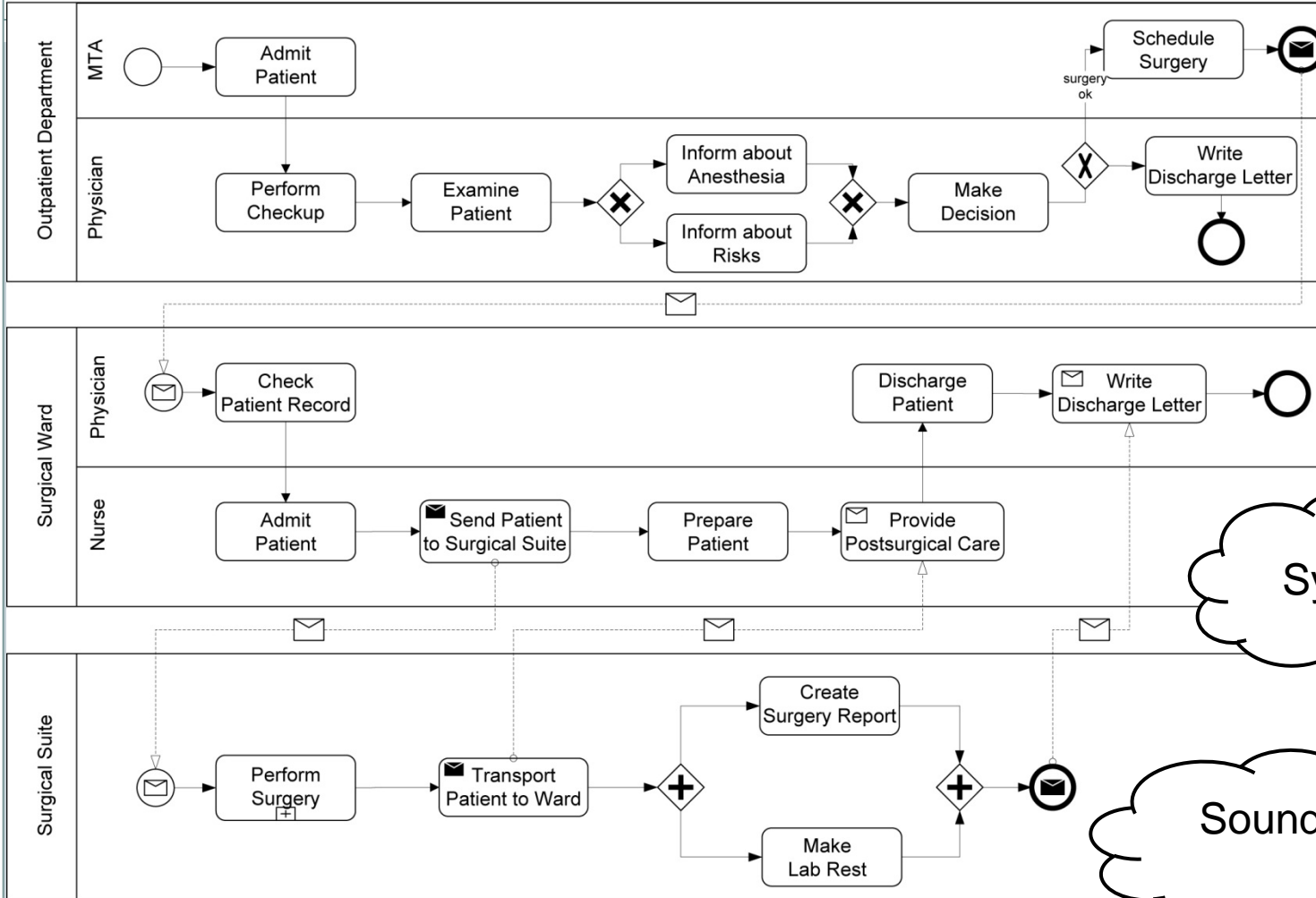
MONTEVIDEO, DECEMBER 11TH 2012



PRESENTED BY
BARBARA WEBER
UNIV. OF INNSBRUCK

*Not part of this keynote
For details see Chapter
10 of the book*

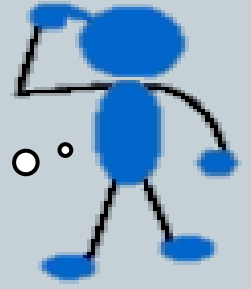
Motivation



Process correct?

Syntax? ✓

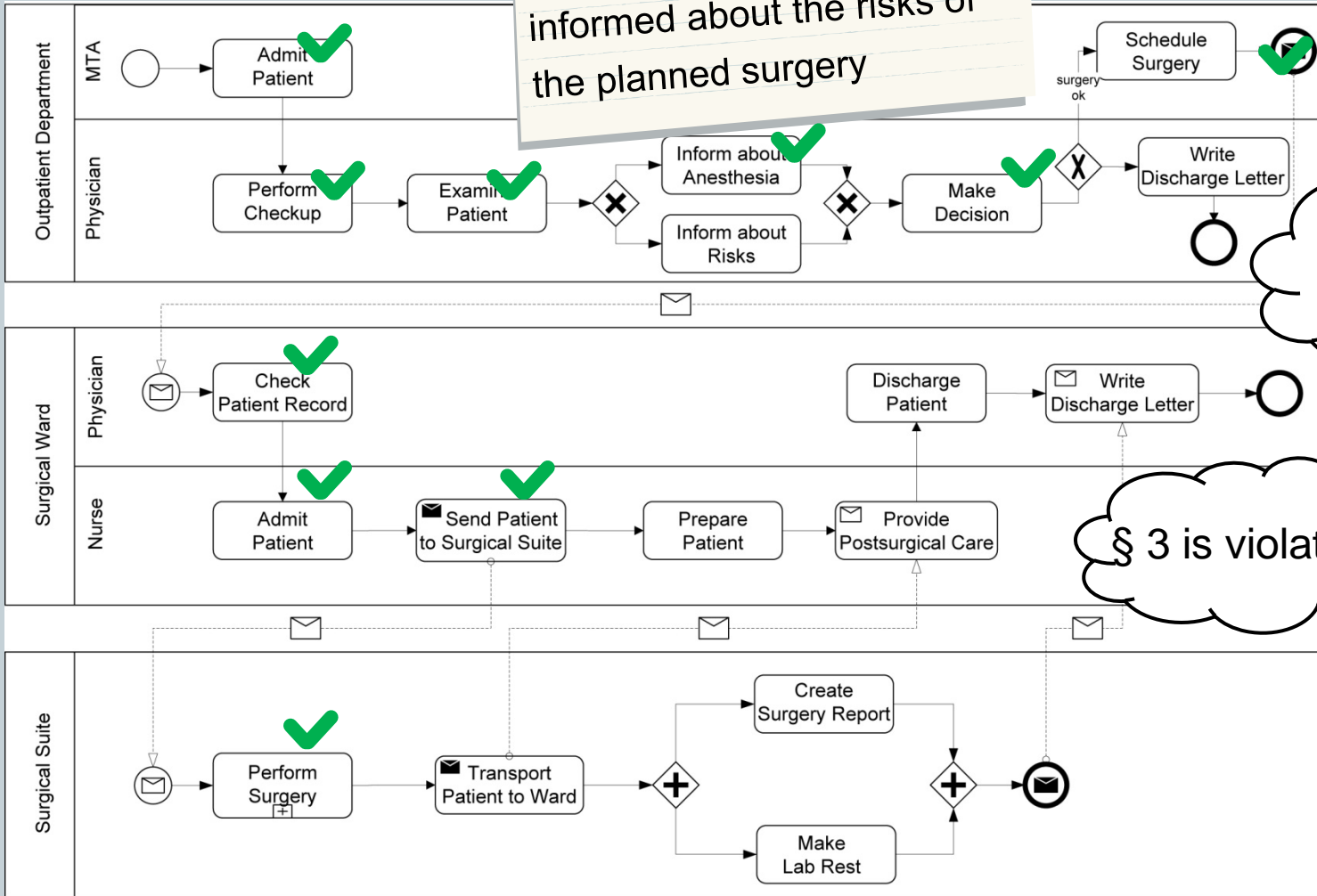
Soundness? ✓



Ulrich, B. Weber
Process-Aware In
tag Berlin Heid

Medical Guideline

§ 3: After the examination, the patient has to be informed about the risks of the planned surgery



Process correct?

§ 3 is violated!



© 2015, B. Weber: Enabling Flexibility in Process-Aware Information Systems @ Springer-Verlag Berlin Heidelberg

Business Processes and Workflows

Summary



MONTEVIDEO, DECEMBER 11TH 2012

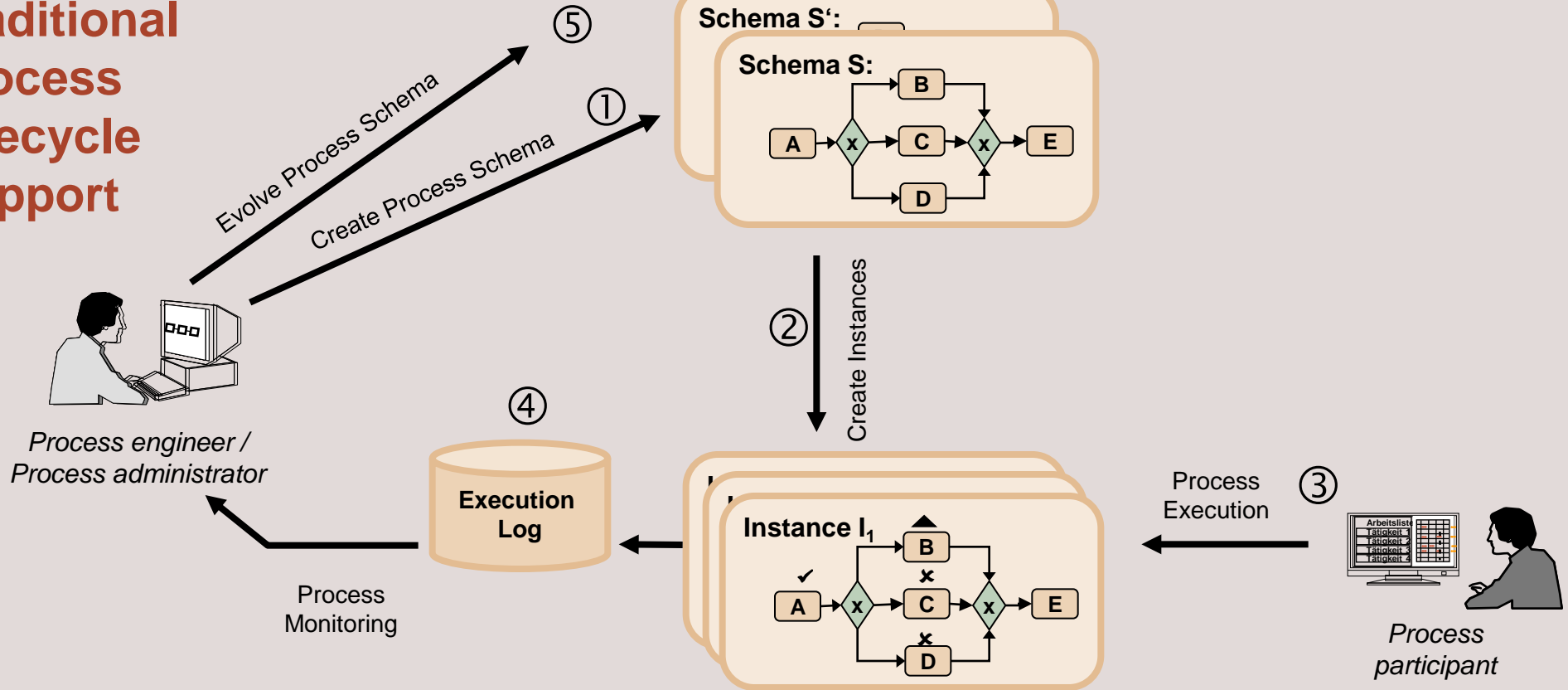


PRESENTED BY
BARBARA WEBER
UNIV. OF INNSBRUCK

Integrated Lifecycle Support for Adaptive and Dynamic Processes (1)

100

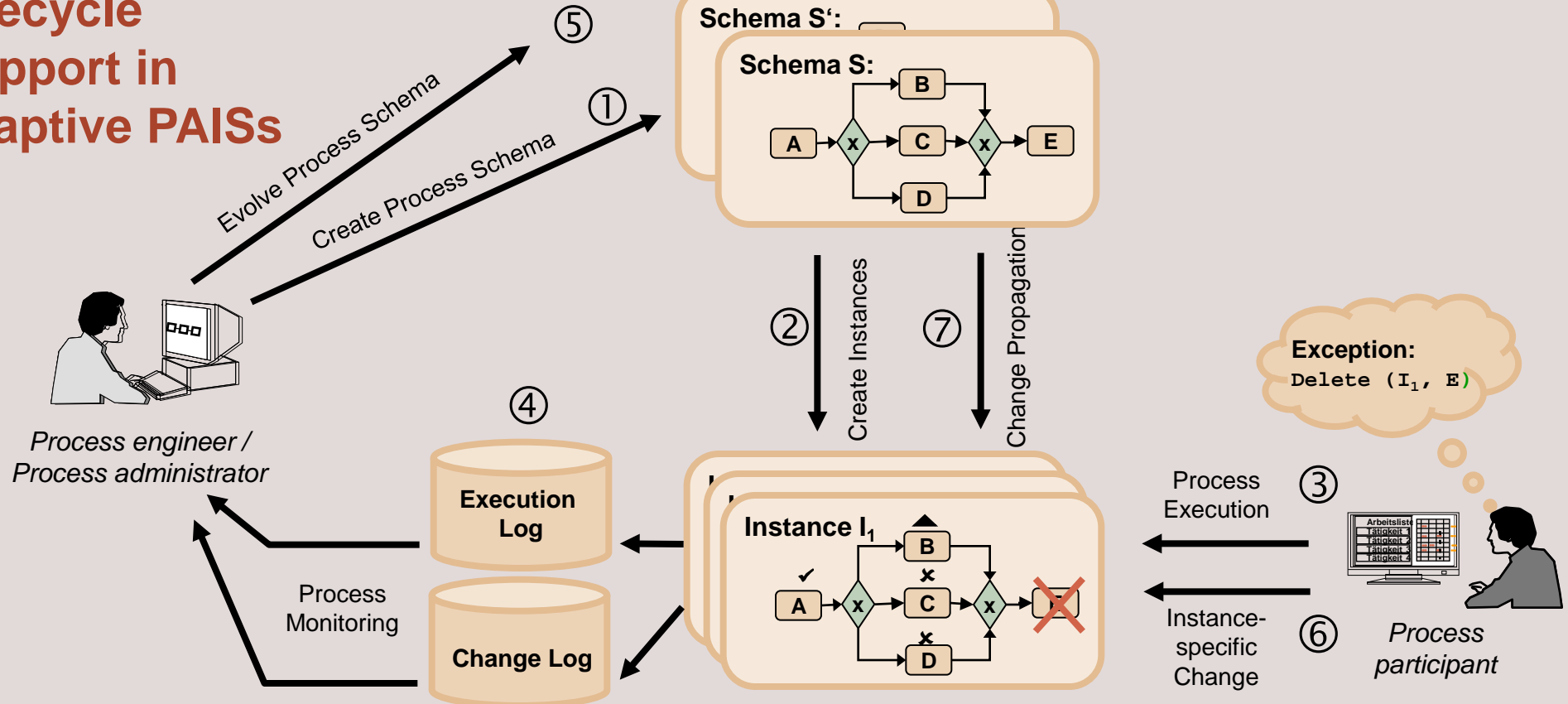
Traditional Process Lifecycle Support



Integrated Lifecycle Support for Adaptive and Dynamic Processes (2)

101

Lifecycle Support in adaptive PAISs



Summary



- **Increasing adoption of PAISs to support business processes at an operational level**
- **Effective business process support imposes several different flexibility needs**
 - **Adaptation, Evolution, Variability, and Looseness**
- **Flexibility needs partially supported by existing commercial systems**

Summary



- **Increasing adoption of PAISs to support business processes at an operational level**
- **Effective business process support imposes several different flexibility needs**
 - Adaptation
 - Evolution
 - Variability
 - Looseness

Summary



- **Adaptation support through**
 - Exception handling (preplanned exceptions)
 - Ad-hoc changes (unforeseen changes)
- **Evolution support through**
 - Versioning of process models
 - Instance migration
- **Variability support through**
 - Process Configuration

Can all be realized with pre-specified process models

Support increasingly available in commercial tools; support in most tools only partially available

Summary



- **Looseness requires different paradigms for representing business processes**
 - Constraint-based process models
 - Data-centric / Object-aware process models

Active area of research

Big interest of commercial vendors



Thank you for your attention !



For more information visit our website <http://bpm.q-e.at/>,
our facebook page www.facebook.com/bpmqe , follow
[bpm_qe](#) on twitter, or send an email to
Barbara.Weber@uibk.ac.at