ulm university universität ulm

Managing Flexibility and Evolution Challenges in Process-aware Information Systems

Scenarios, Technologies, Tools

1

PROF. DR. MANFRED REICHERT
ULM UNIVERSITY

manfred.reichert@uni-ulm.de
www.uni-ulm.de/dbis

Content

- Part 1: Process-aware Information Systems
- Part 2: Flexibility Issues
- Part 3: Flexibility Support for Pre-specified Process Models
- Part 4: Loosely-specified Process Models

Process-Aware Information Systems

A Retail Process

Welcome customer

Offer Clothes

Bill Clothes

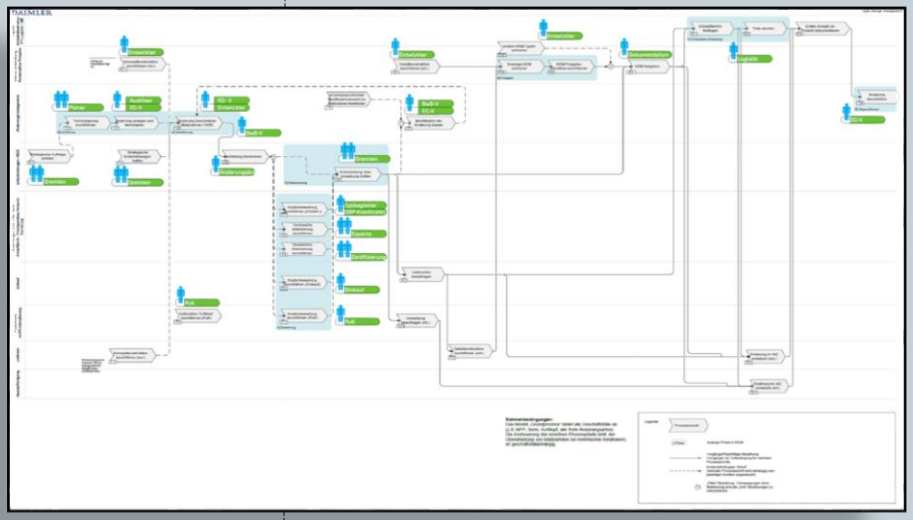
Hand over clothes

Mendling 2006

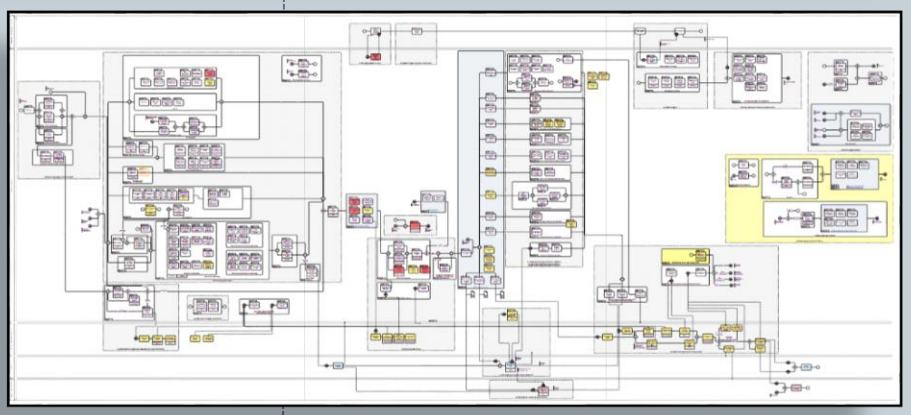
A More Complex Process Scenario

Business Level

Model created by domain expert

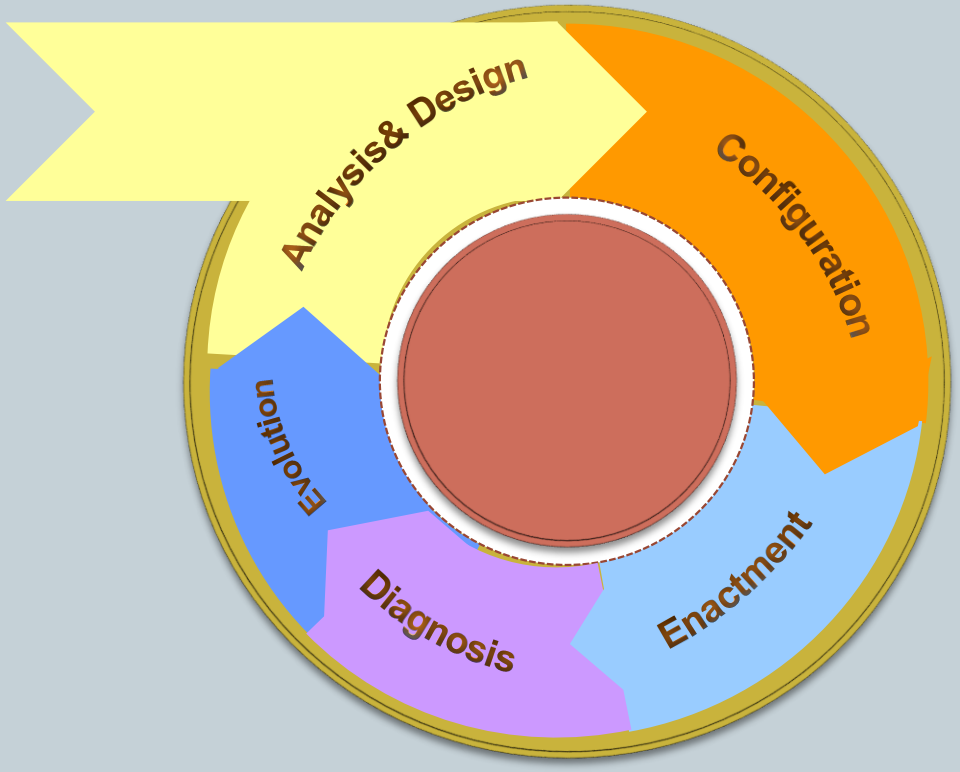


Model created by IT expert

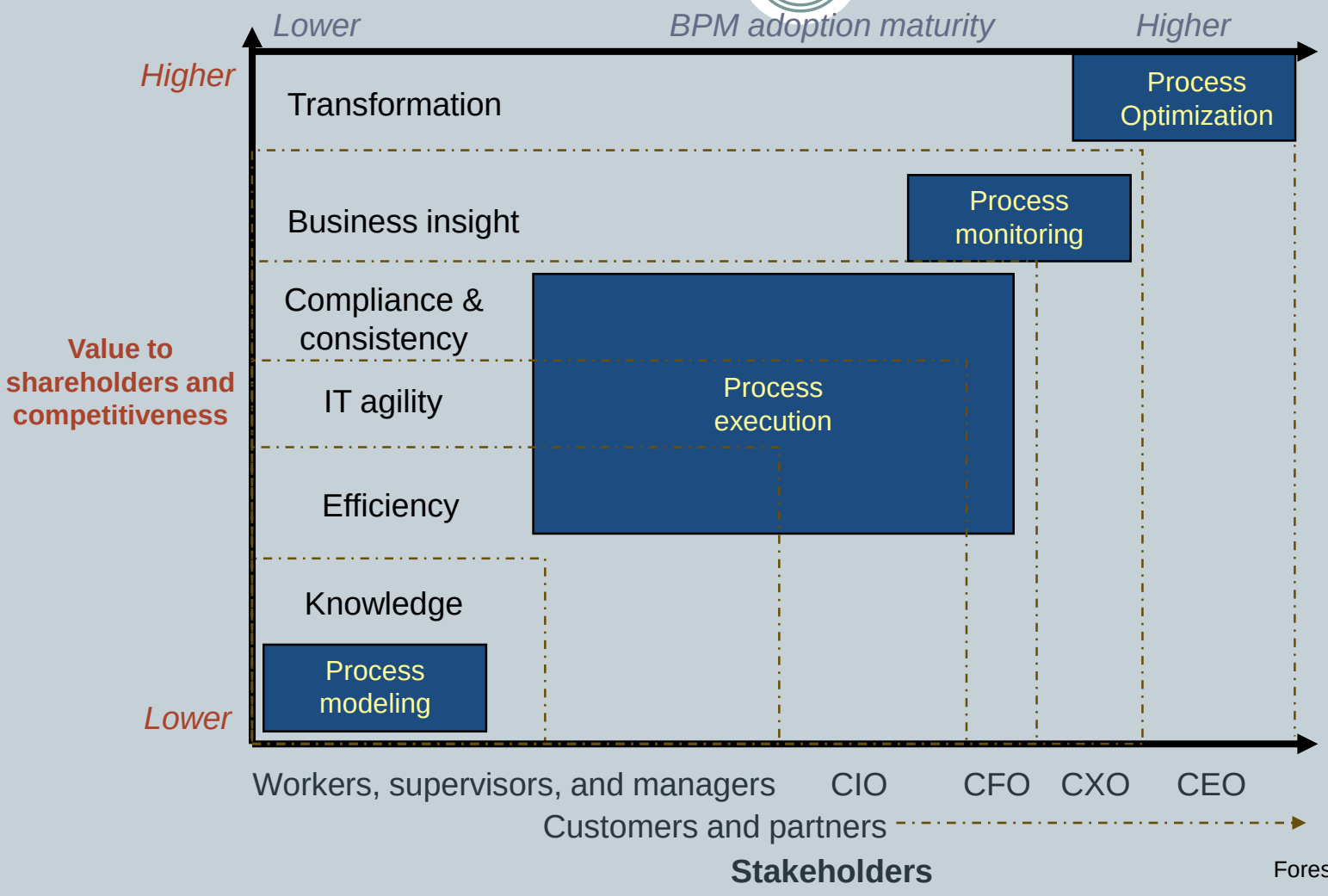


IT Level

Business Process Lifecycle

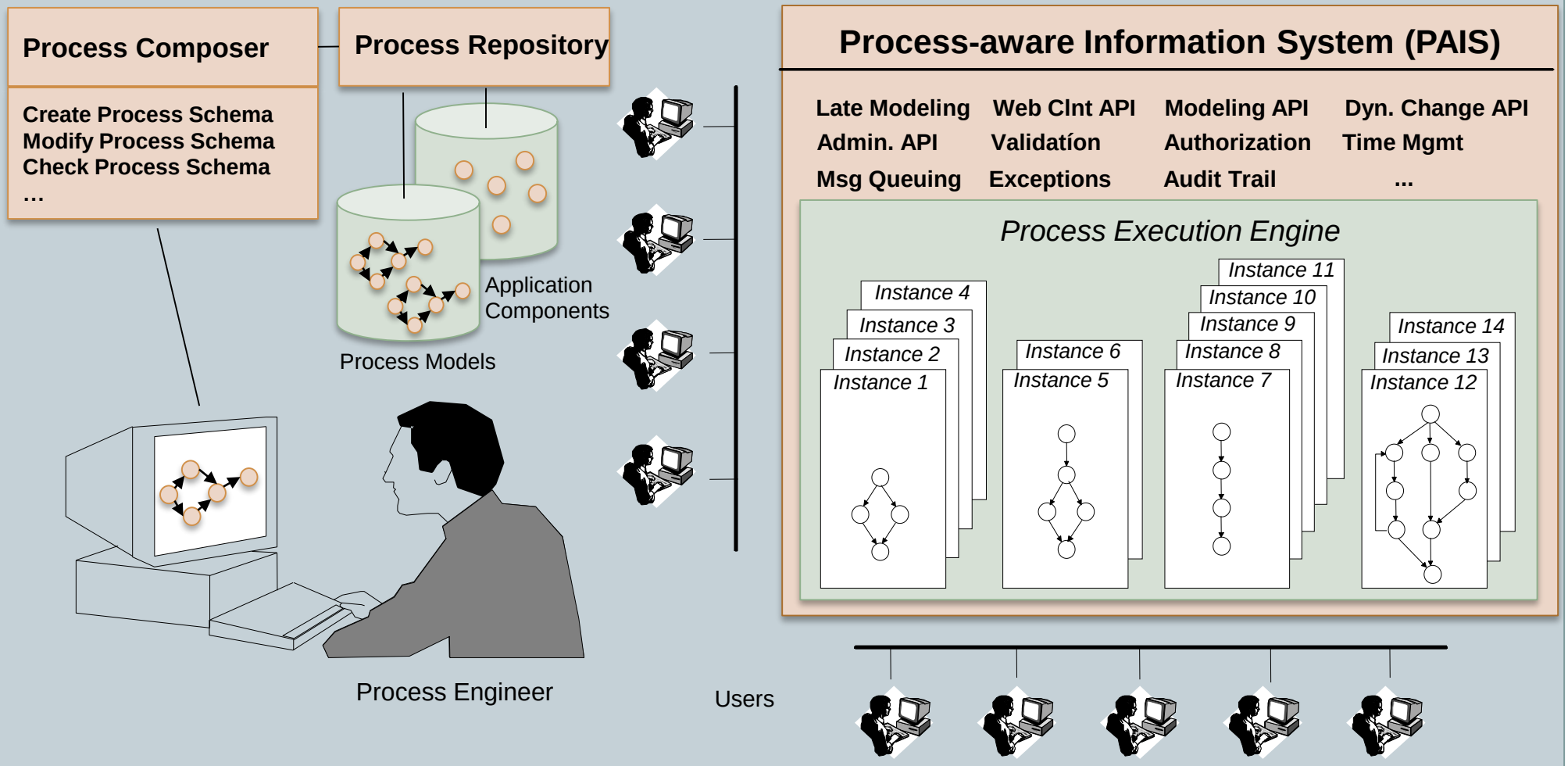


BPM Value Proposition

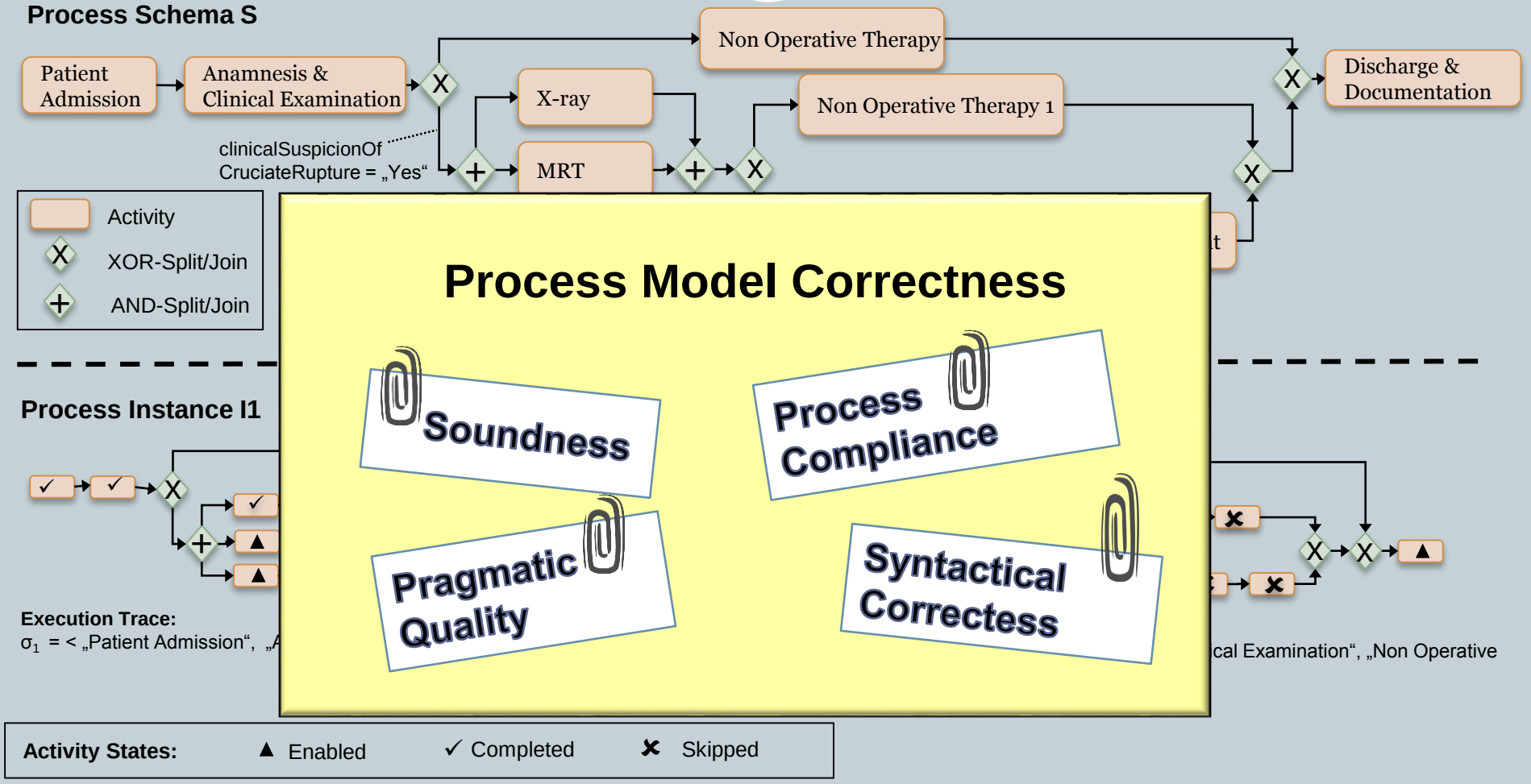


Forester 2007 BPM Market Overview

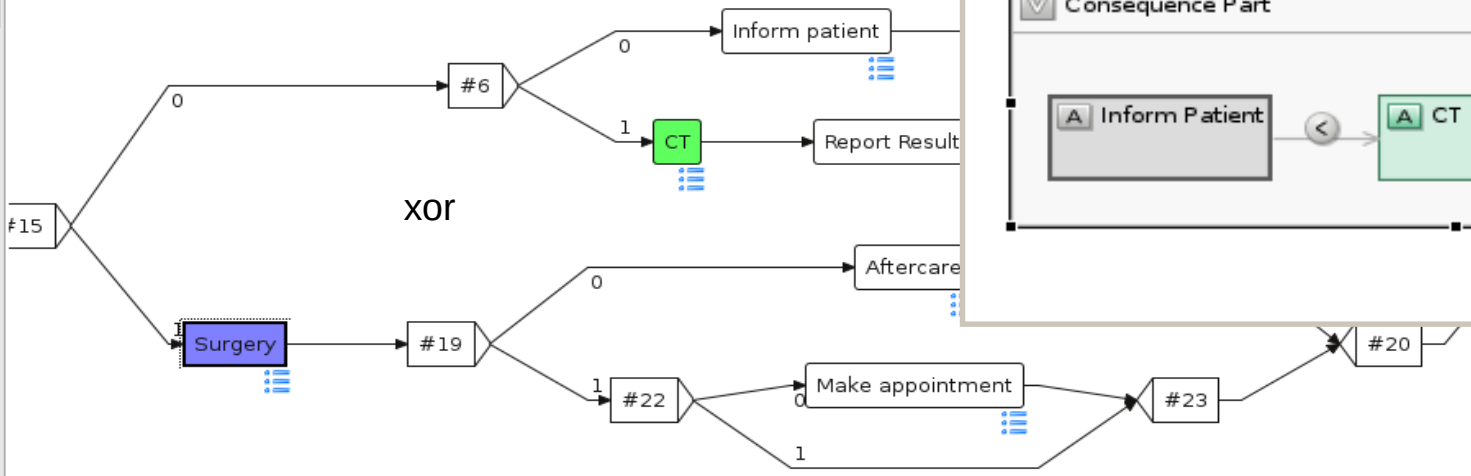
Process-aware Information System

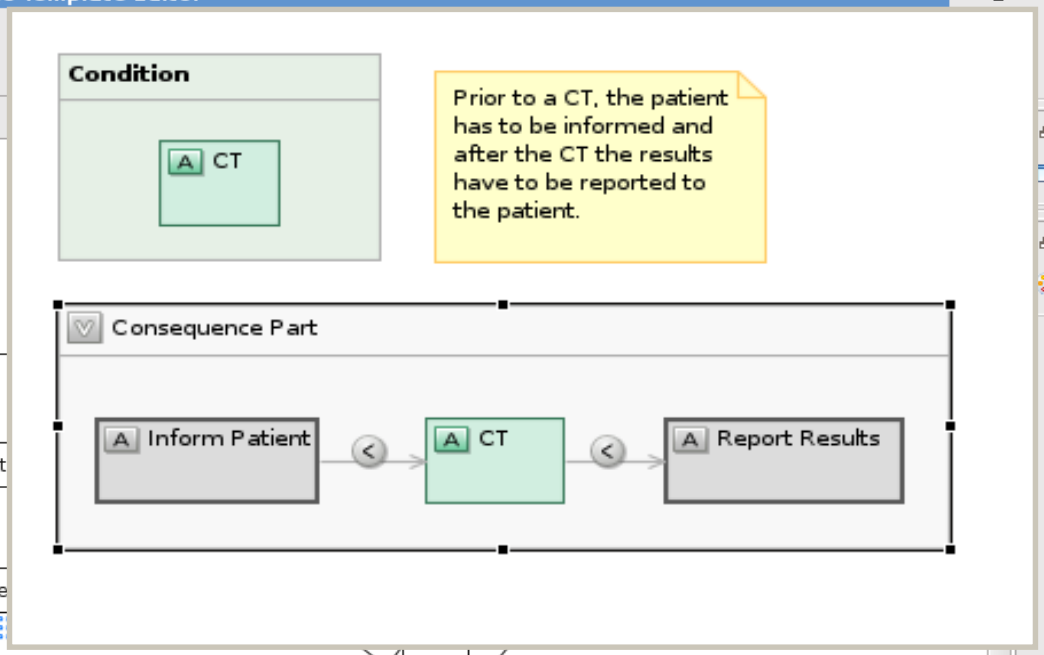


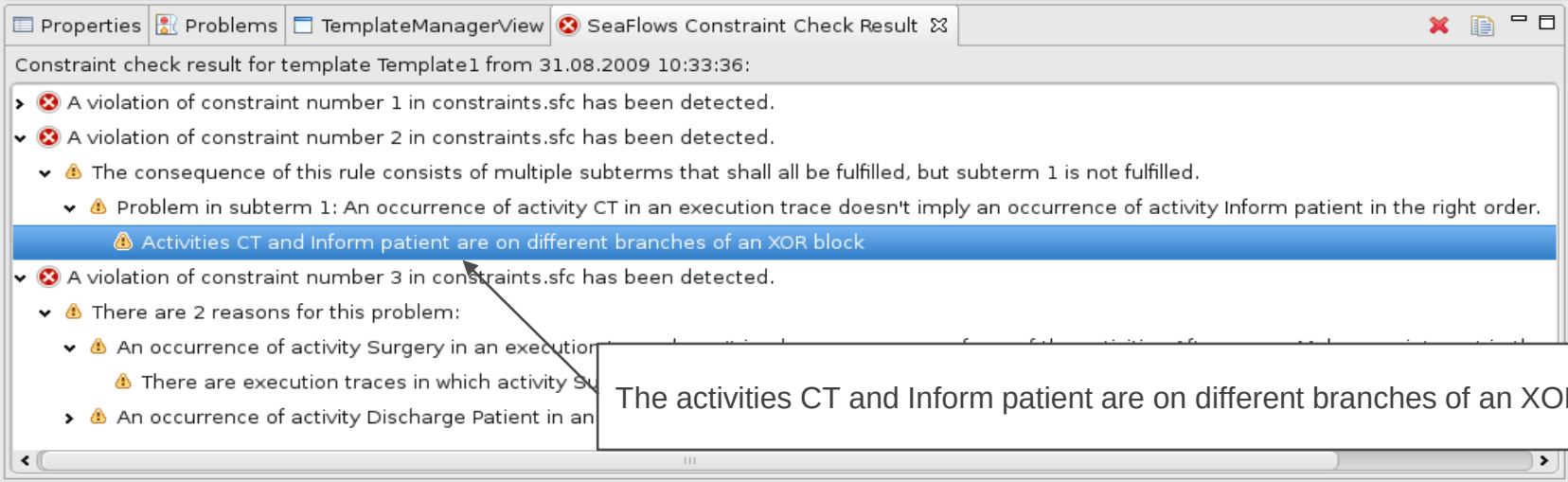
A Process Model and Related Instances



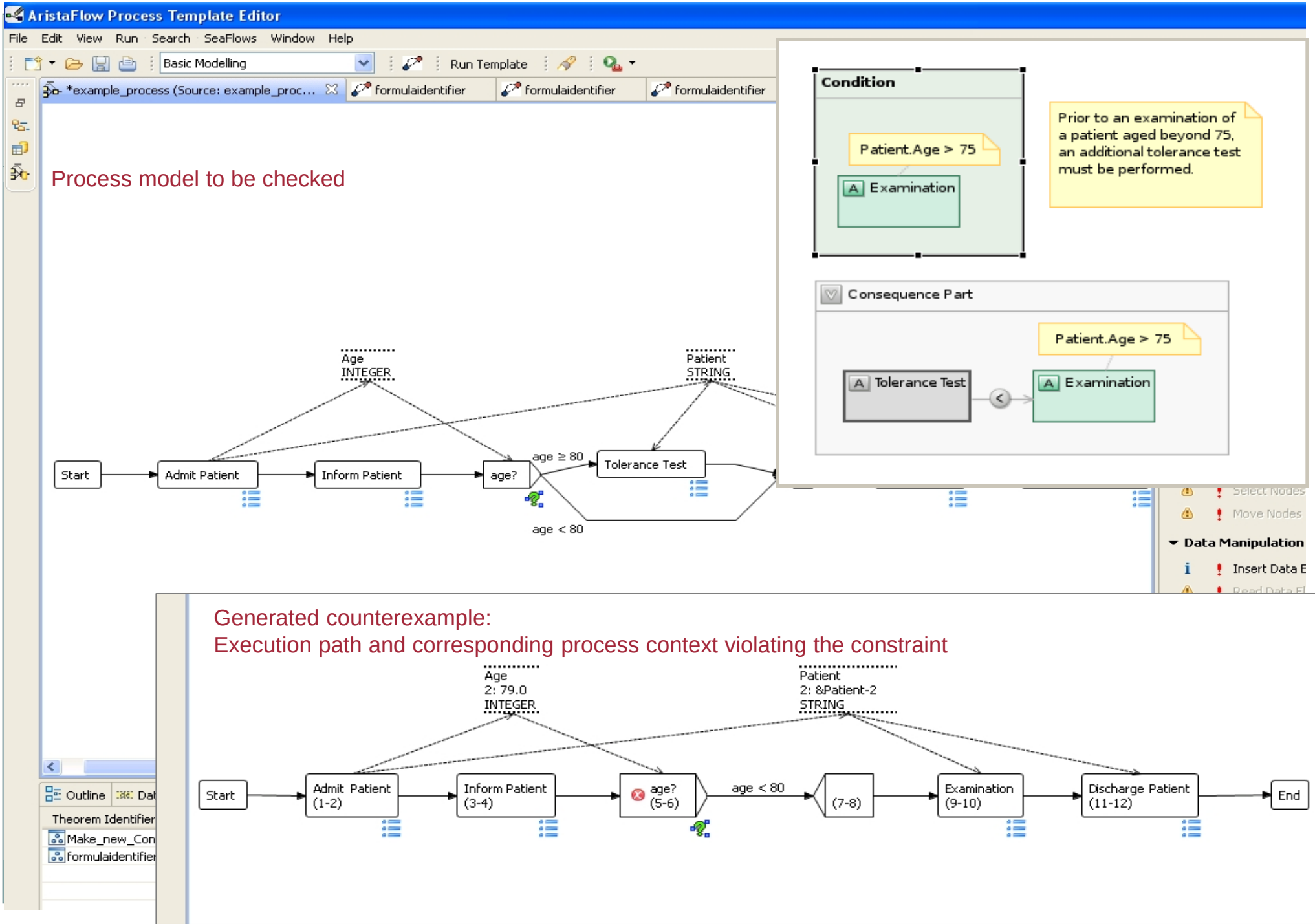
Example



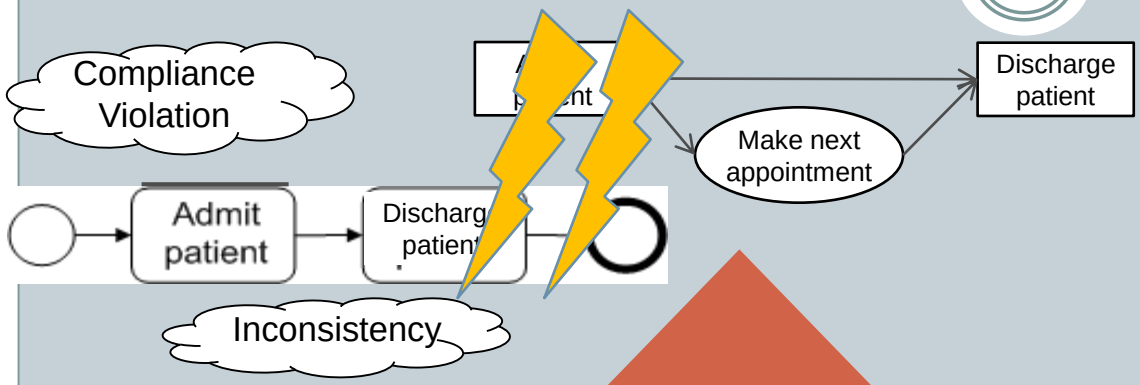


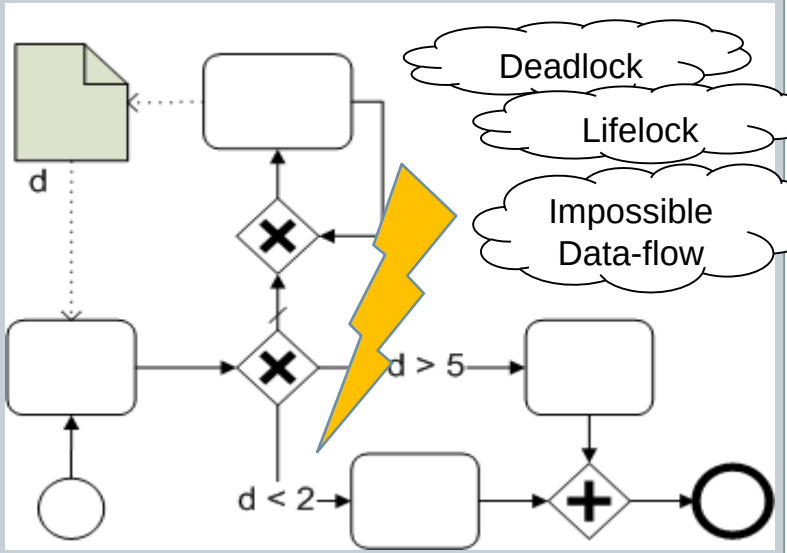


The activities CT and Inform patient are on different branches of an XOR-Block


AristaFlow Process Template Editor
File Edit View Run Search SeaFlows Window Help
Basic Modelling
Run Template
*example_process (Source: example_proc...
formulaidentifier
formulaidentifier
formulaidentifier
Process model to be checked
Condition
Patient.Age > 75
Examination
Prior to an examination of a patient aged beyond 75, an additional tolerance test must be performed.
Consequence Part
Patient.Age > 75
Tolerance Test
Examination
Age
INTEGER
Patient
STRING
Start
Admit Patient
Inform Patient
age?
age ≥ 80
Tolerance Test
age < 80
Select Nodes
Move Nodes
Data Manipulation
Insert Data E
Generated counterexample:
Execution path and corresponding process context violating the constraint
Age
2: 79.0
INTEGER
Patient
2: &Patient-2
STRING
Start
Admit Patient
(1-2)
Inform Patient
(3-4)
age?
(5-6)
age < 80
(7-8)
Examination
(9-10)
Discharge Patient
(11-12)
End
Outline
Theorem Identifier
Make_new_Con
formulaidentifier

Levels of Process Model Correctness



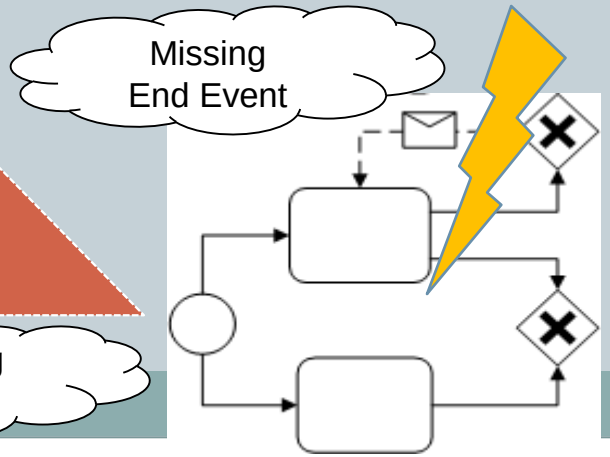


Semantic Correctness
(Business Process Compliance)

Behavioral Correctness

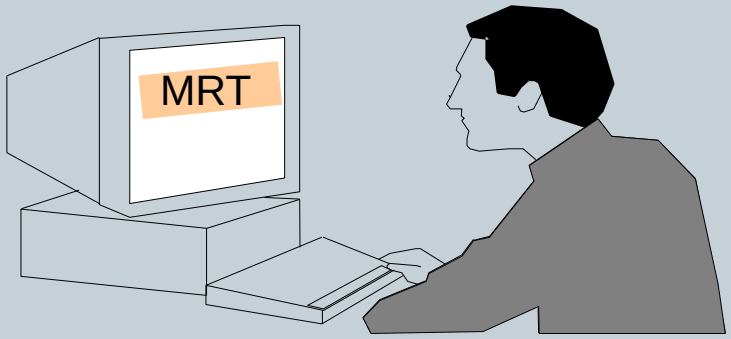
(Soundness)

Syntactical Correctness

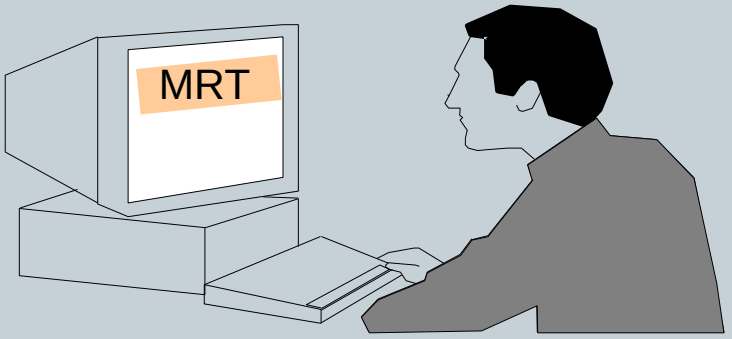


Wrong Flows

User Perspective

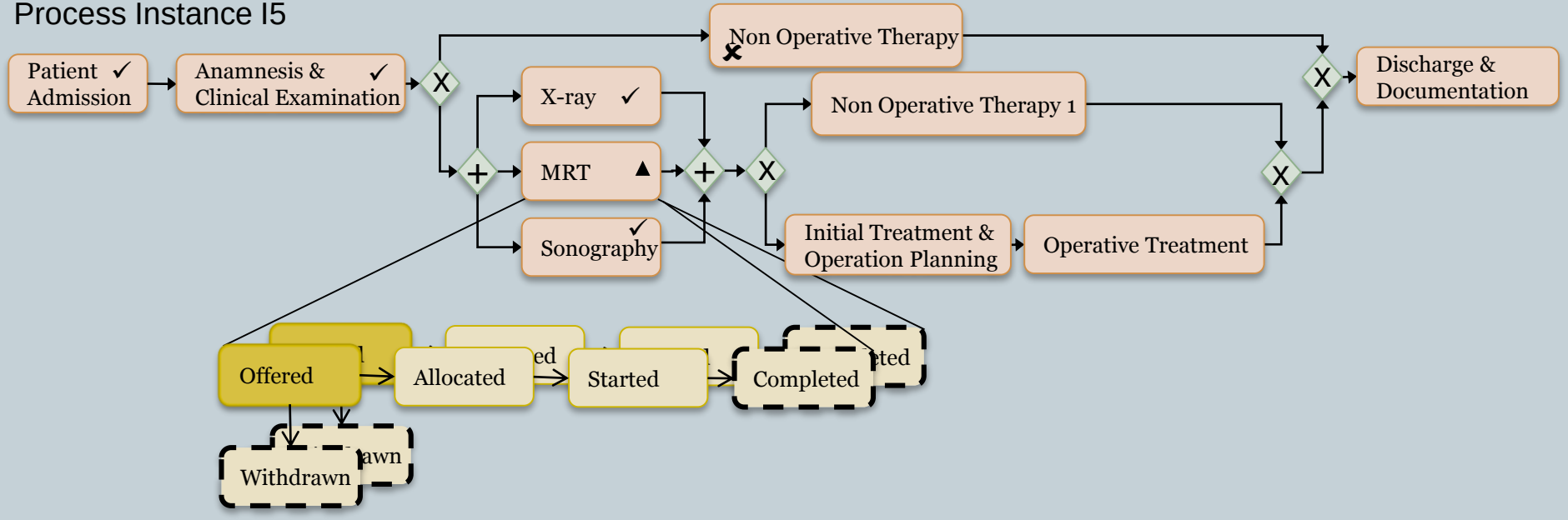


Joe

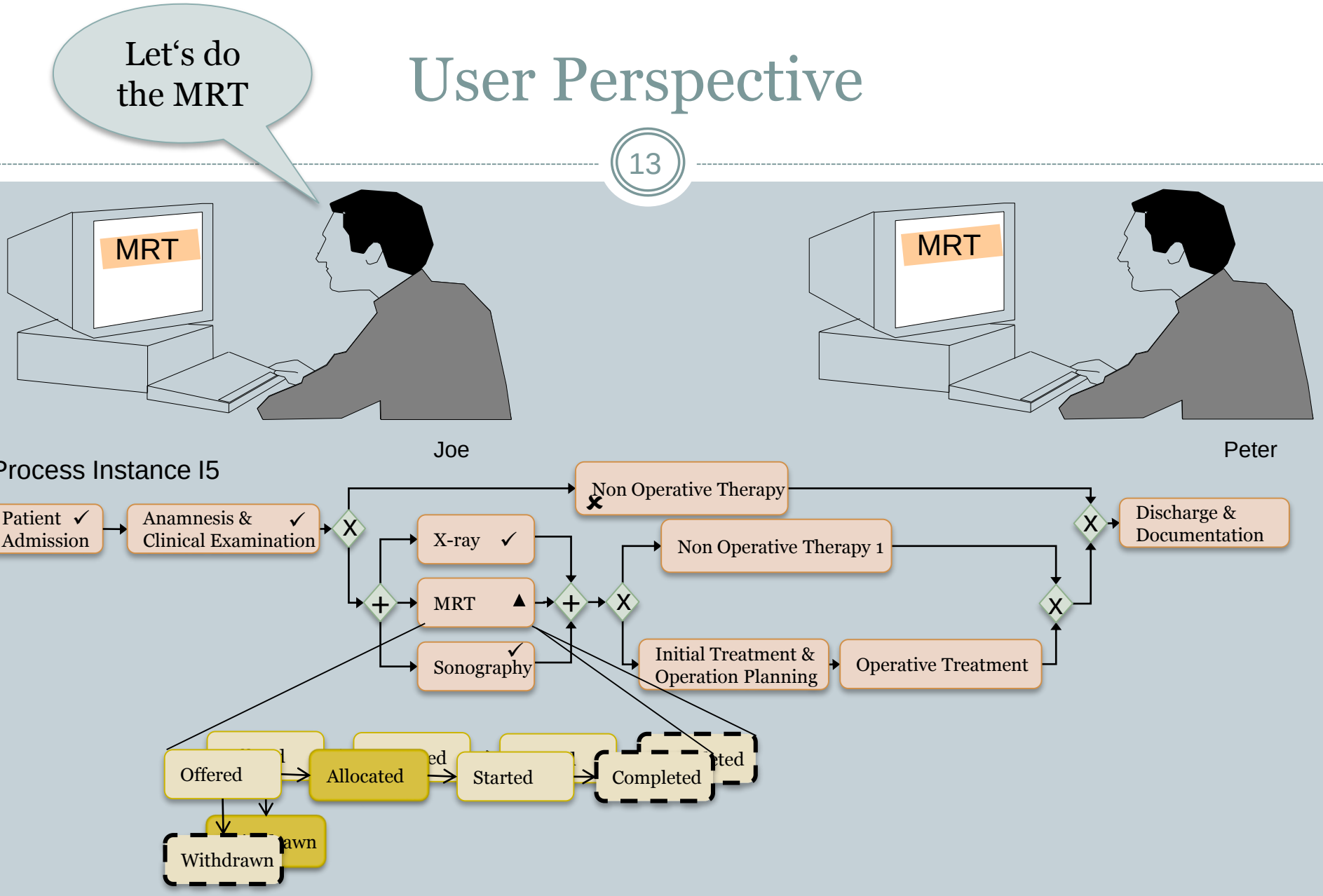


Peter

Process Instance I5



User Perspective

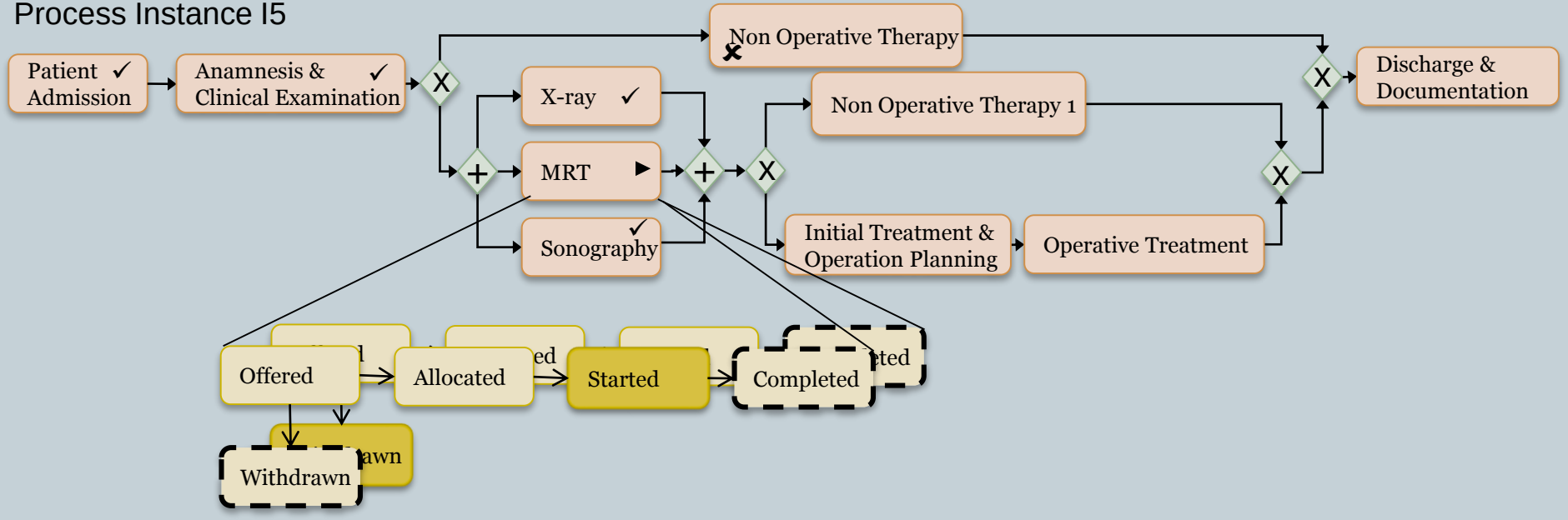


User Perspective

Joe

Peter

Process Instance I5

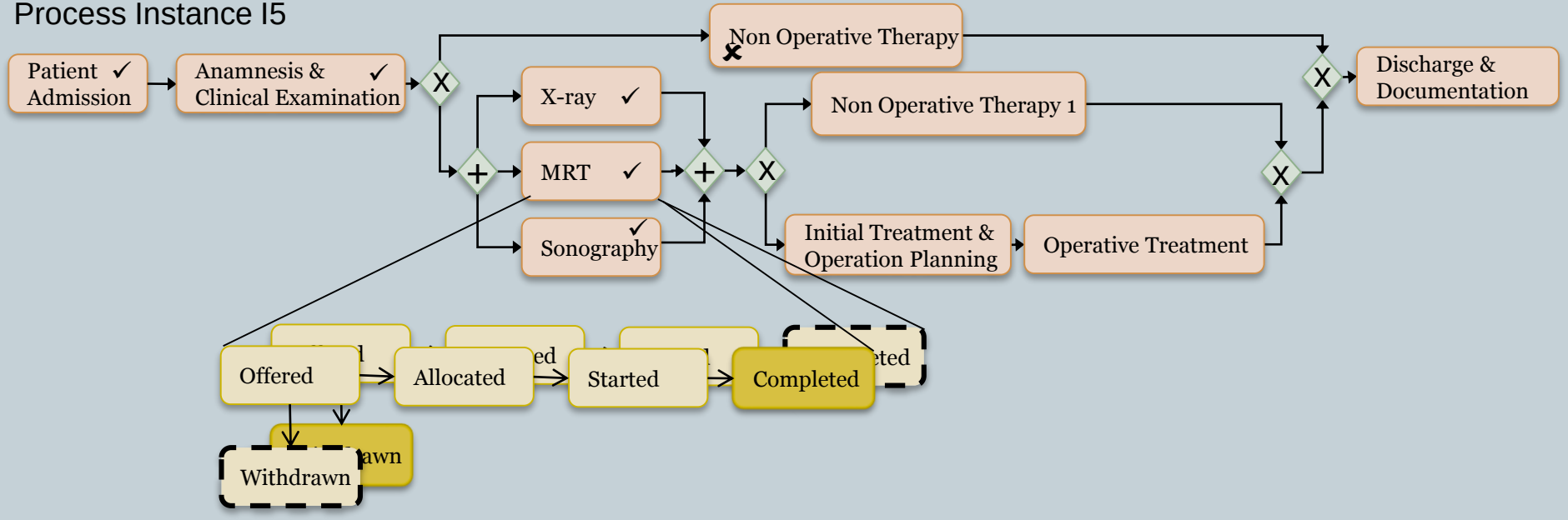


User Perspective

Joe

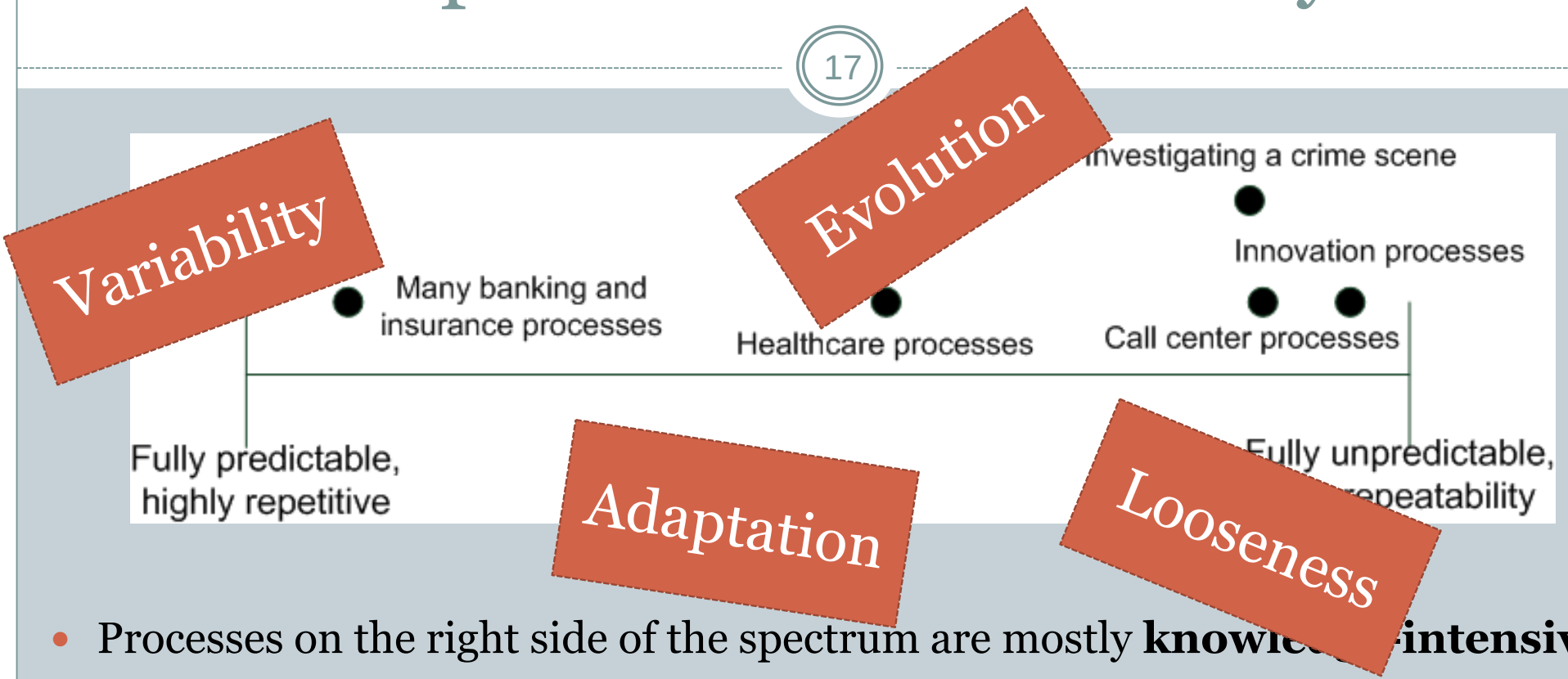
Peter

Process Instance I5



Flexibility Issues

Process Spectrum and Flexibility Needs

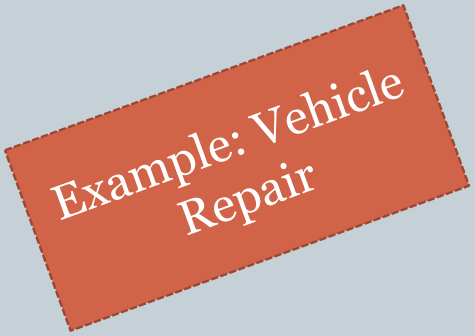


- Processes on the right side of the spectrum are mostly **knowl intensive**
  - **Unpredictability:** Course of action depends on situation-specific parameters
  - **Non-repeatability:** Two process instances hardly look the same
  - **Emergence:** Future course of action depends on knowledge gained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

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



Looseness

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

- Drivers
  - Unpredictability
  - Non-Repeatability
  - Emergence

Example: Treatment Processes in a Hospital

Adaptation

- Ability to adapt the process and its structure to temporary events

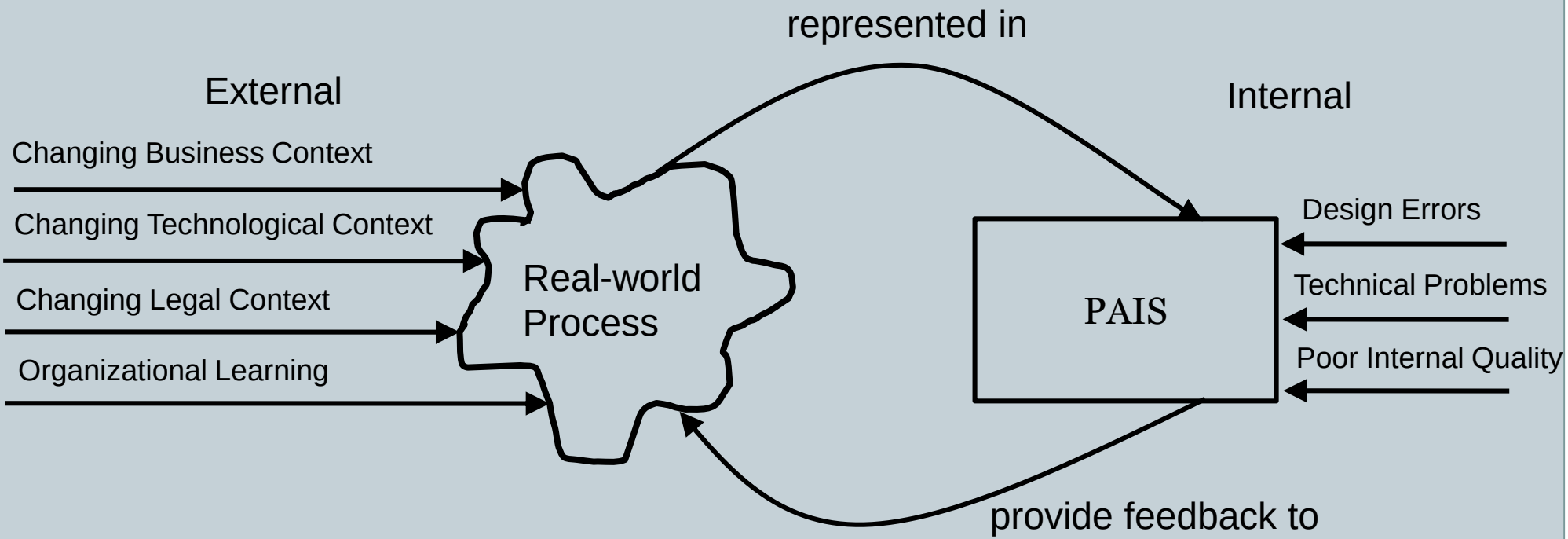
- Drivers
  - Special Situations
  - Exceptions

- Anticipation of Adaptation
  - Planned
  - Unanticipated

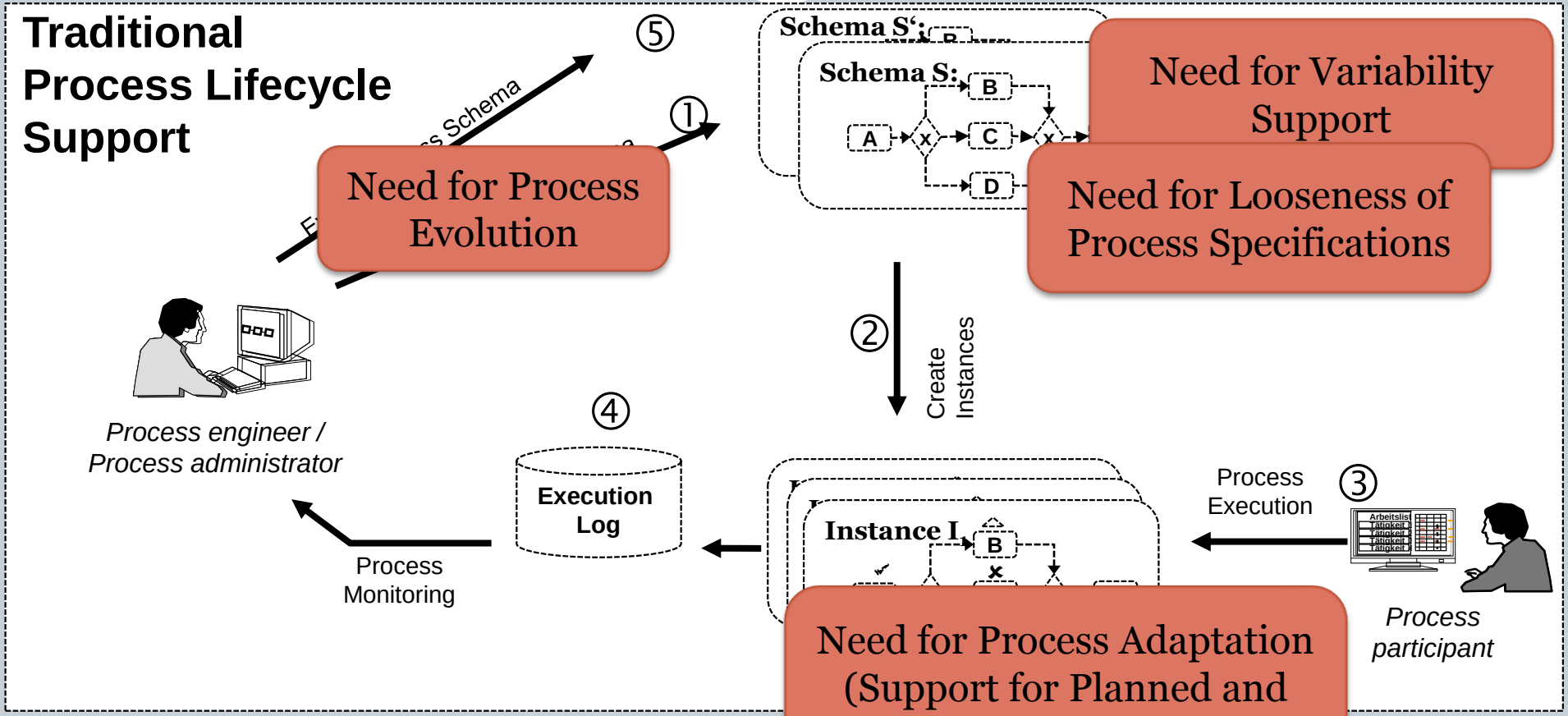
Example: Examination Procedures in a Hospital

Evolution

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



Flexibility Issues along the Process Lifecycle



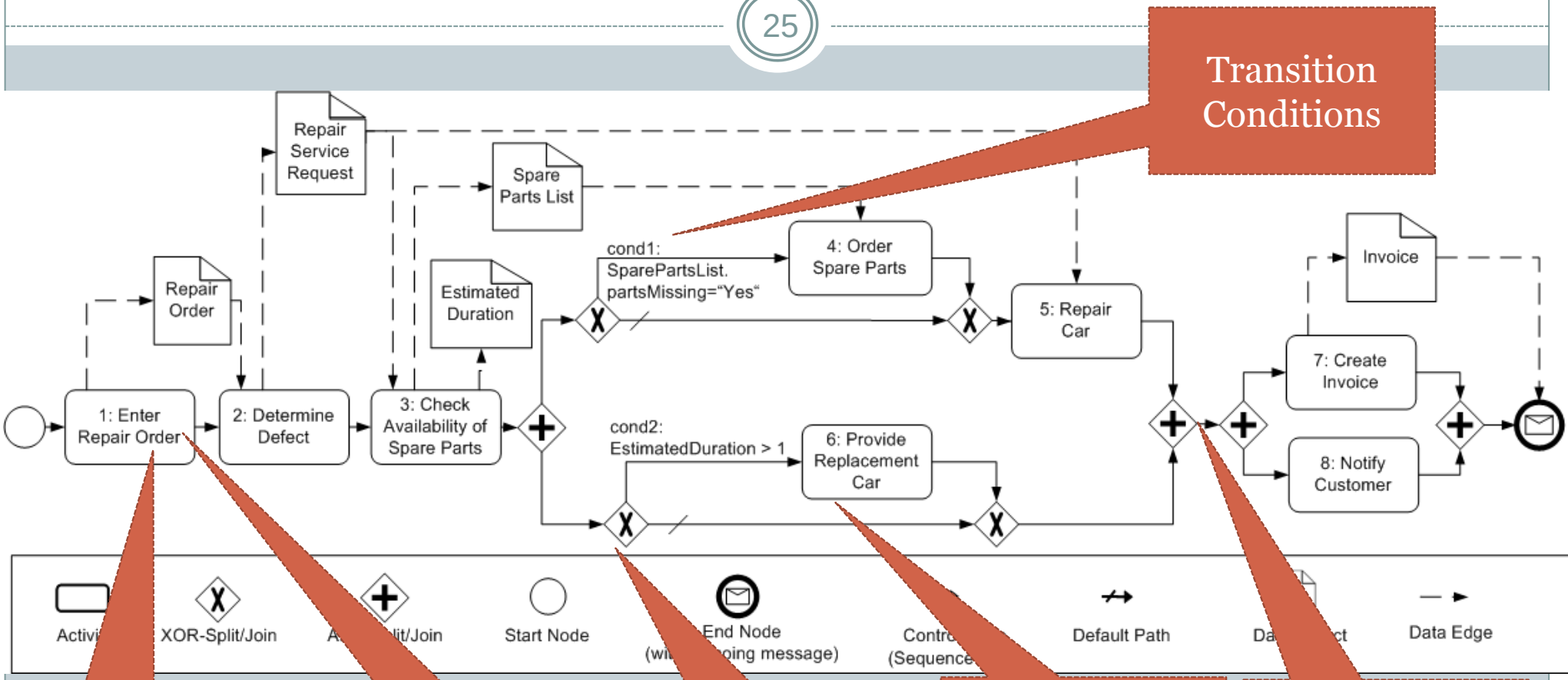
[WRW+09]

Flexibility Needs and Technological Requirements

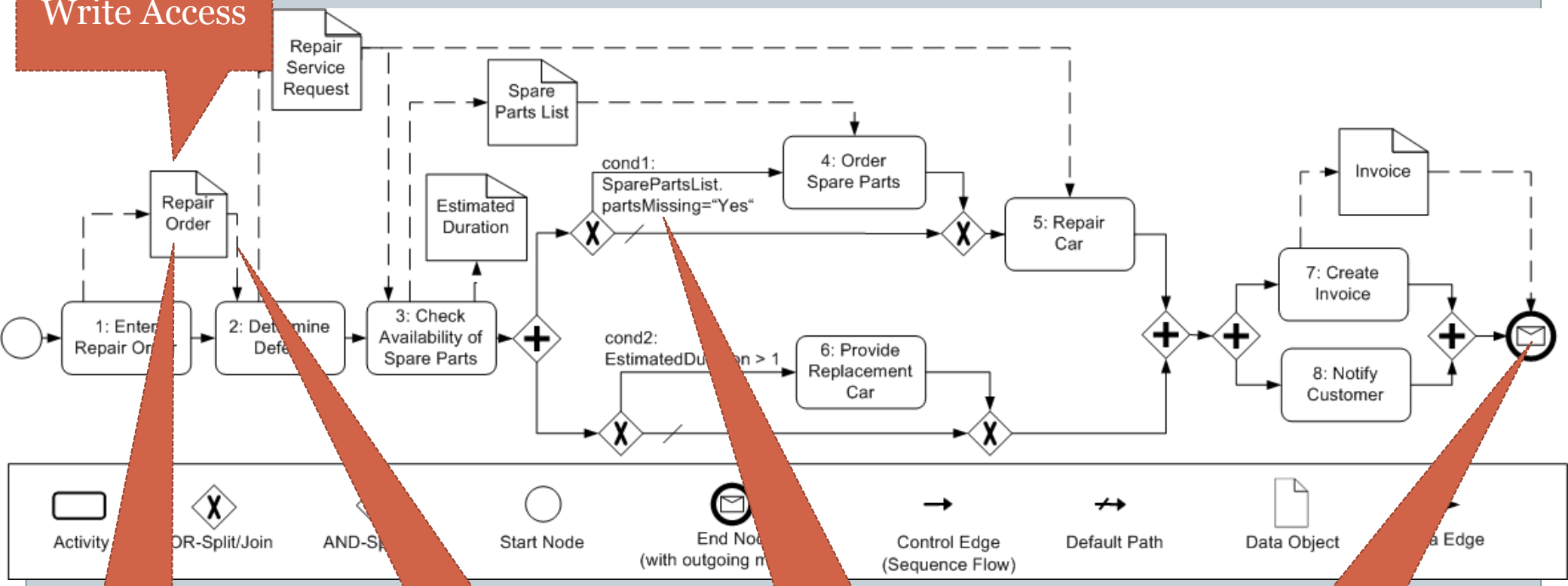
Flexibility Need	Dimension	Technological Requirement
Variability		Configuration
Looseness		Loosely-specified processes
Adaptation	Planned Unplanned	Exception Handling Ad-hoc Changes
Evolution	Deferred Evolution Immediate Evolution Poor Internal Quality Organizational Learning	Versioning Process Instance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

Flexibility-By-Design

Basic Control Flow Concepts & Patterns



Basic Data Flow Concepts & Patterns



Data Edge – Write Access

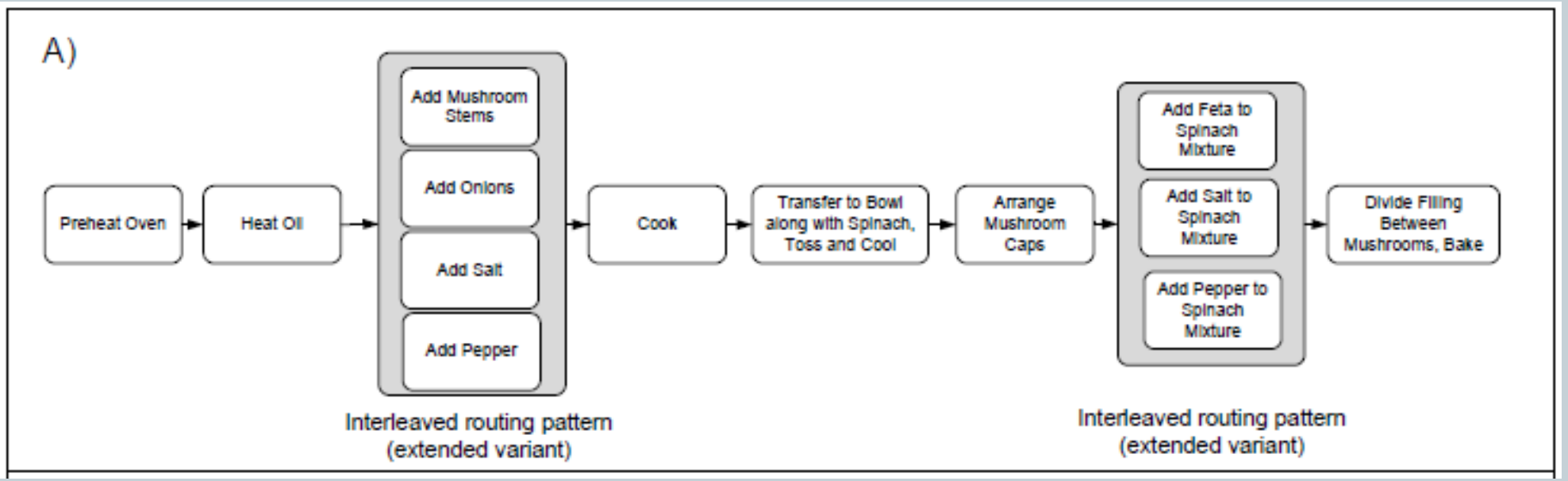
Data Object

Data Edge – Read Access

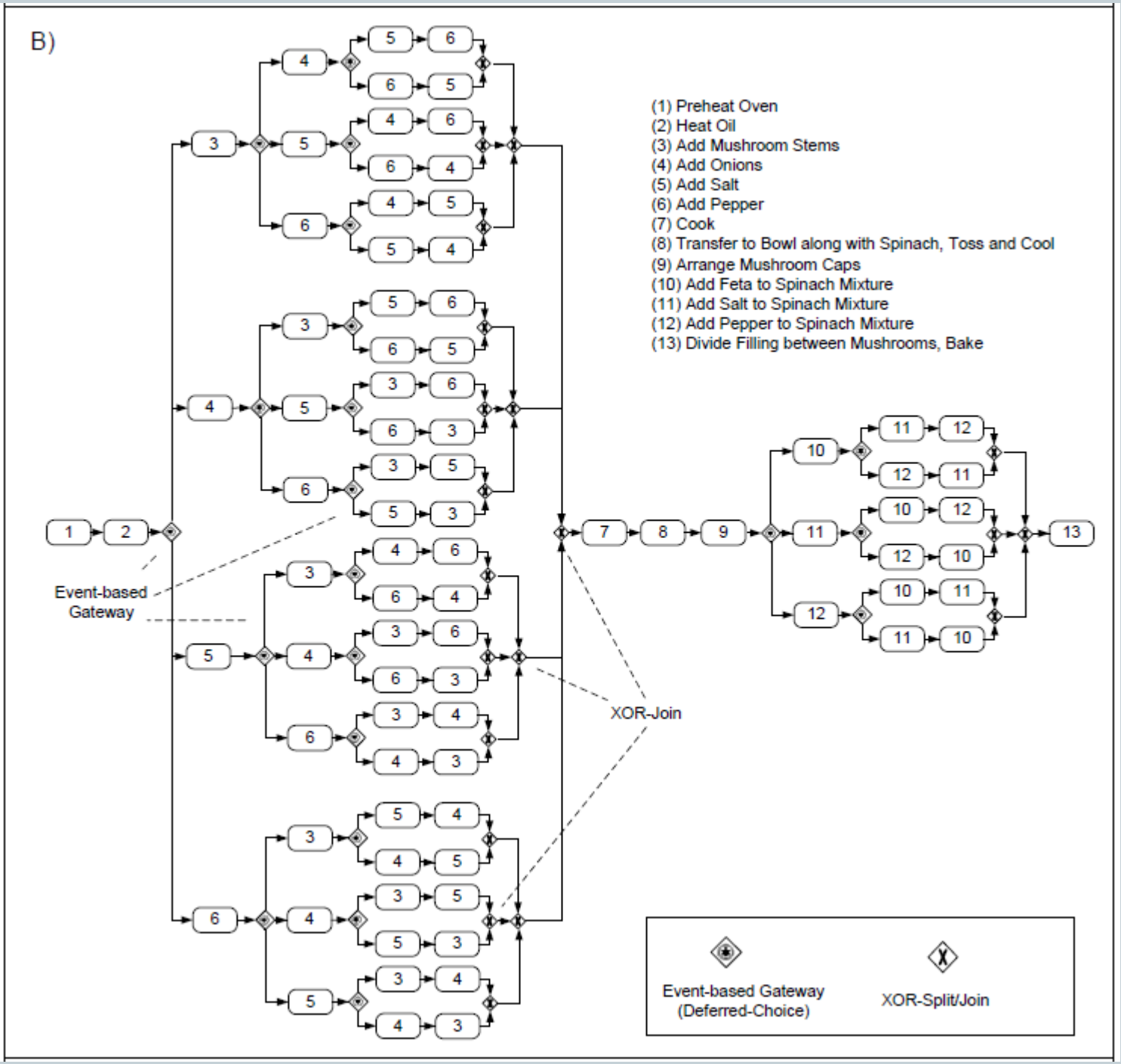
Transition Condition references *SparePartsList*

End Message with Data Obejct *Invoice*

Expressiveness and Flexibility-by-Design



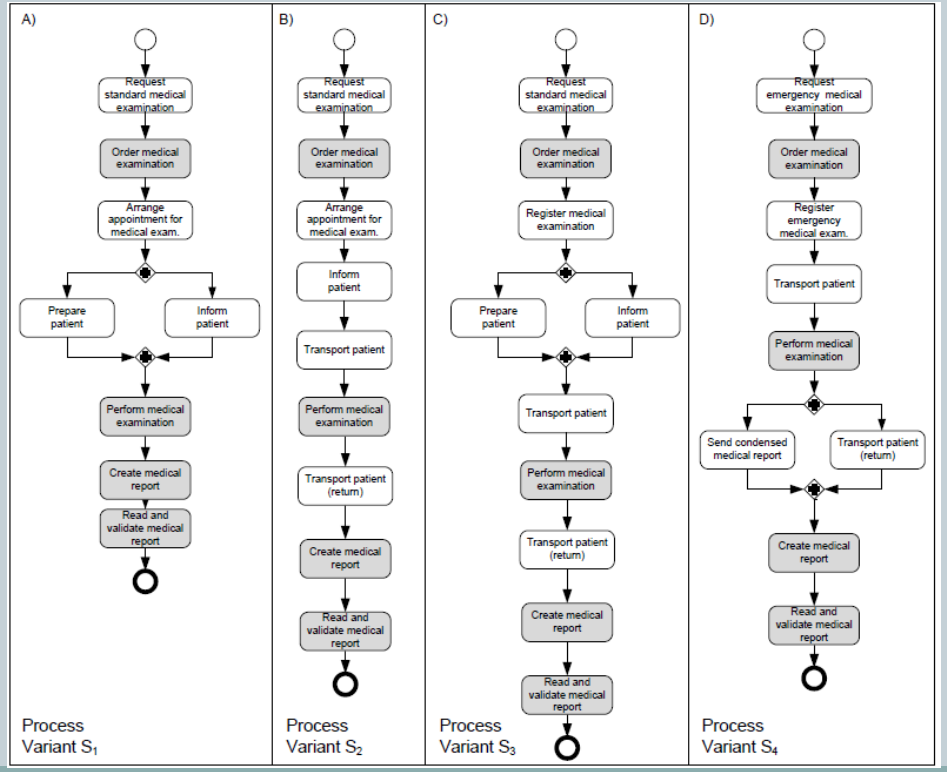
(Missing) Expressiveness and Flexibility by Design



Configurable Process Models

Variety of related variants

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

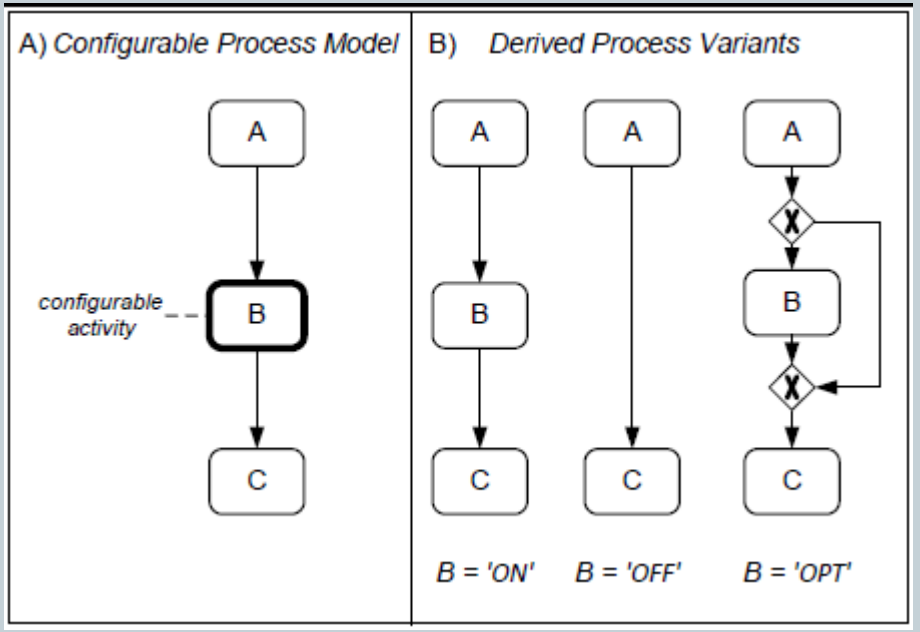


Configurable Process Models

- 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, Provop
- Configurable nodes represent **variation points** associated with configuration alternatives
- Possible combinations of configuration alternatives can be restriceted through constraints

Configurable Activities

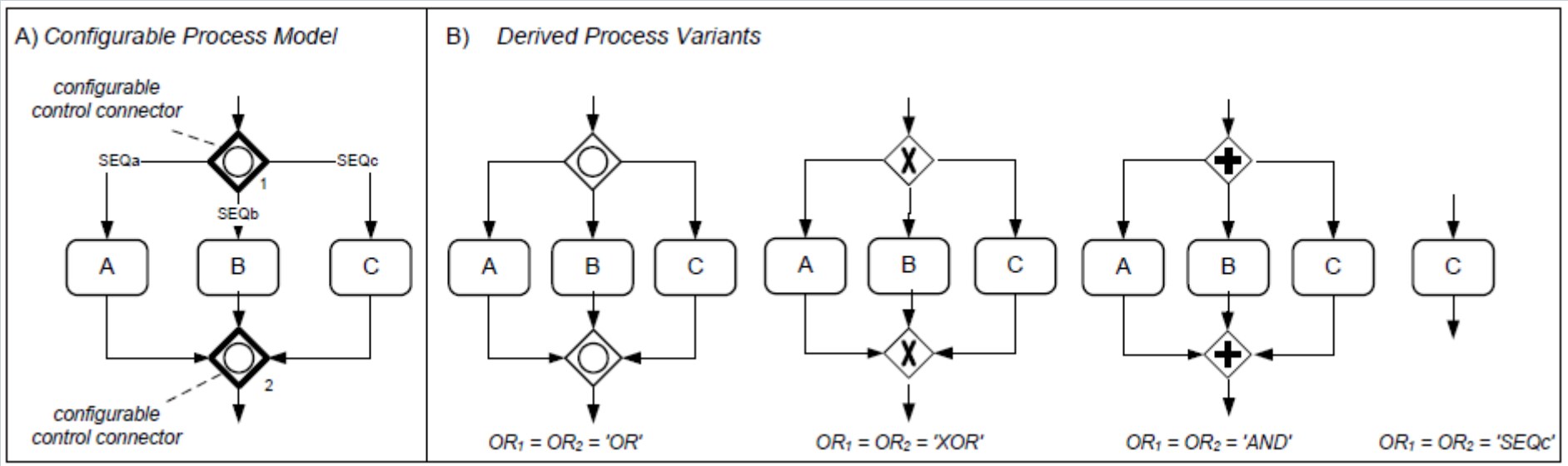
- Included (ON)
- Excluded (OFF)
- Conditional (OPT)



Configurable Control Connectors

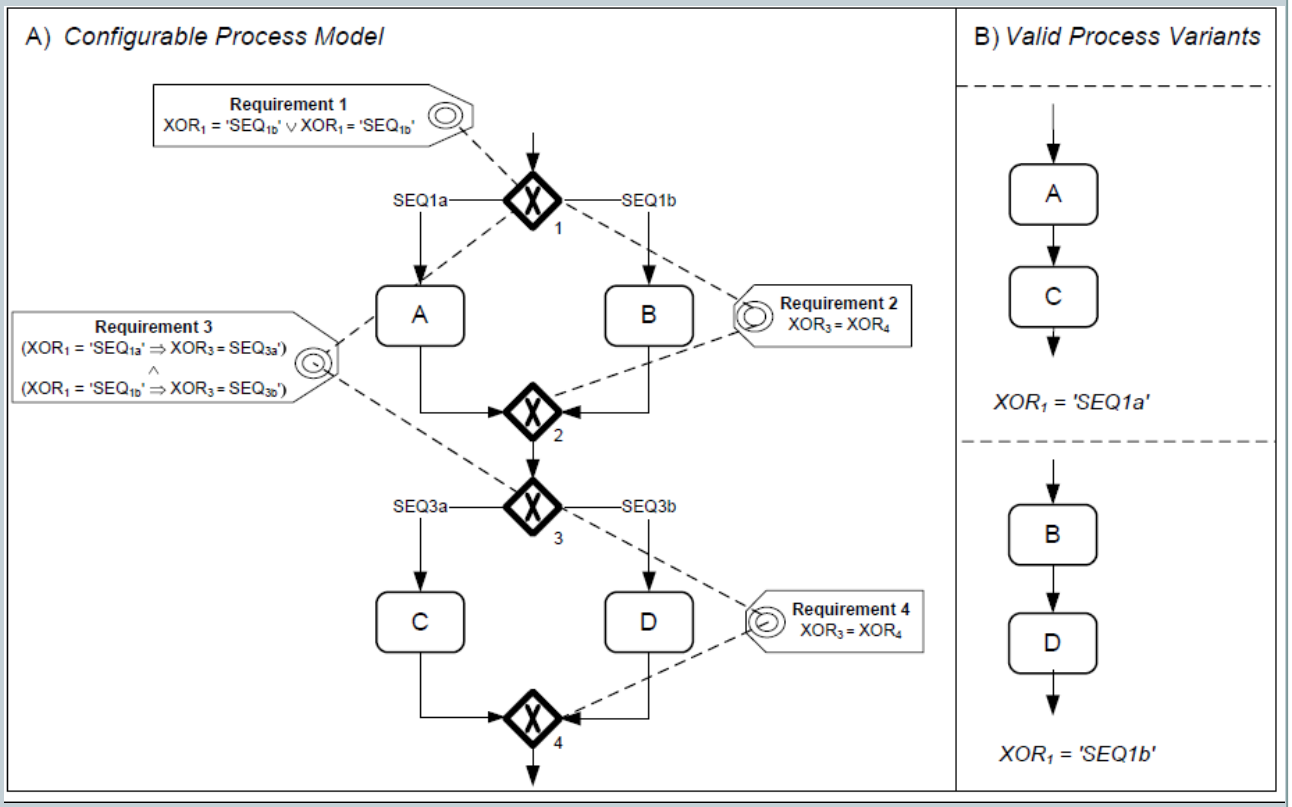
- **Configurable OR**
- Configurable XOR
- Configurable AND

Can be configured to a connector equally restrictive or less restrictive

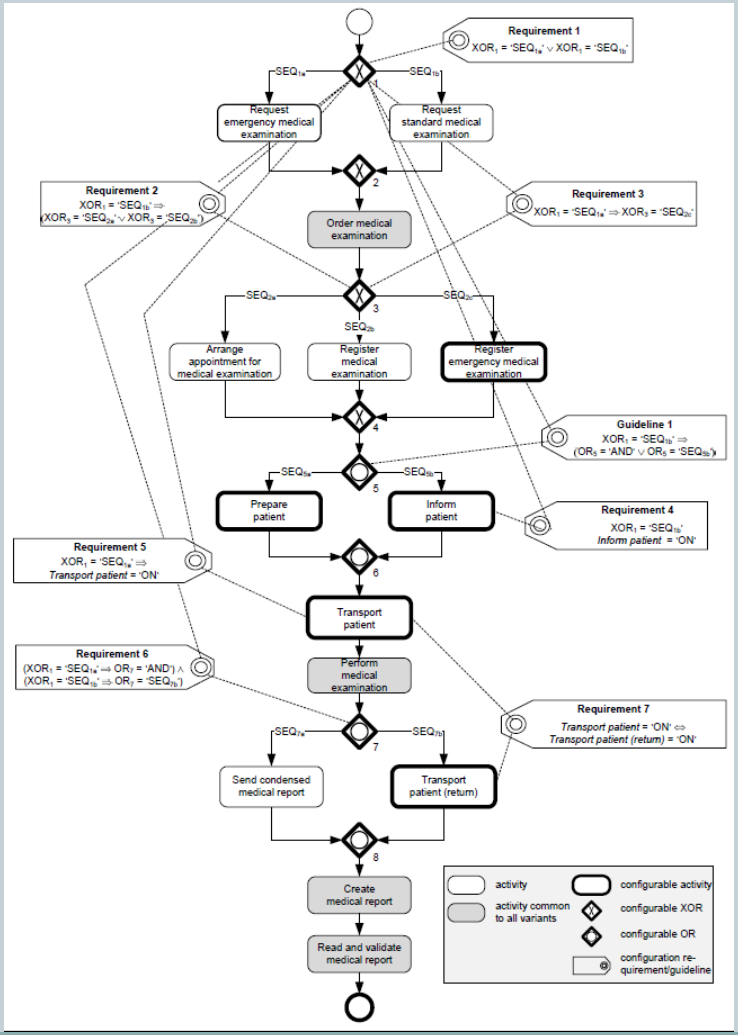


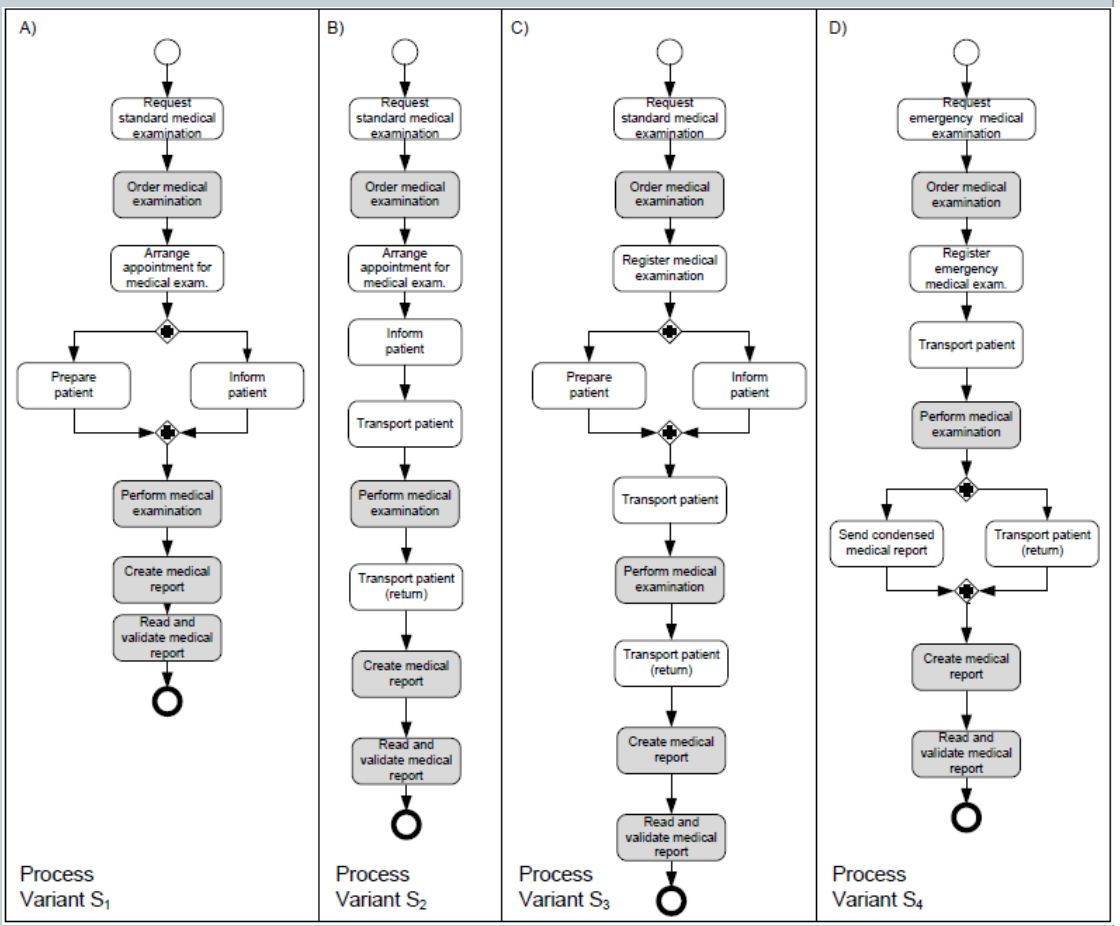
Configuration Requirements and Guidelines

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

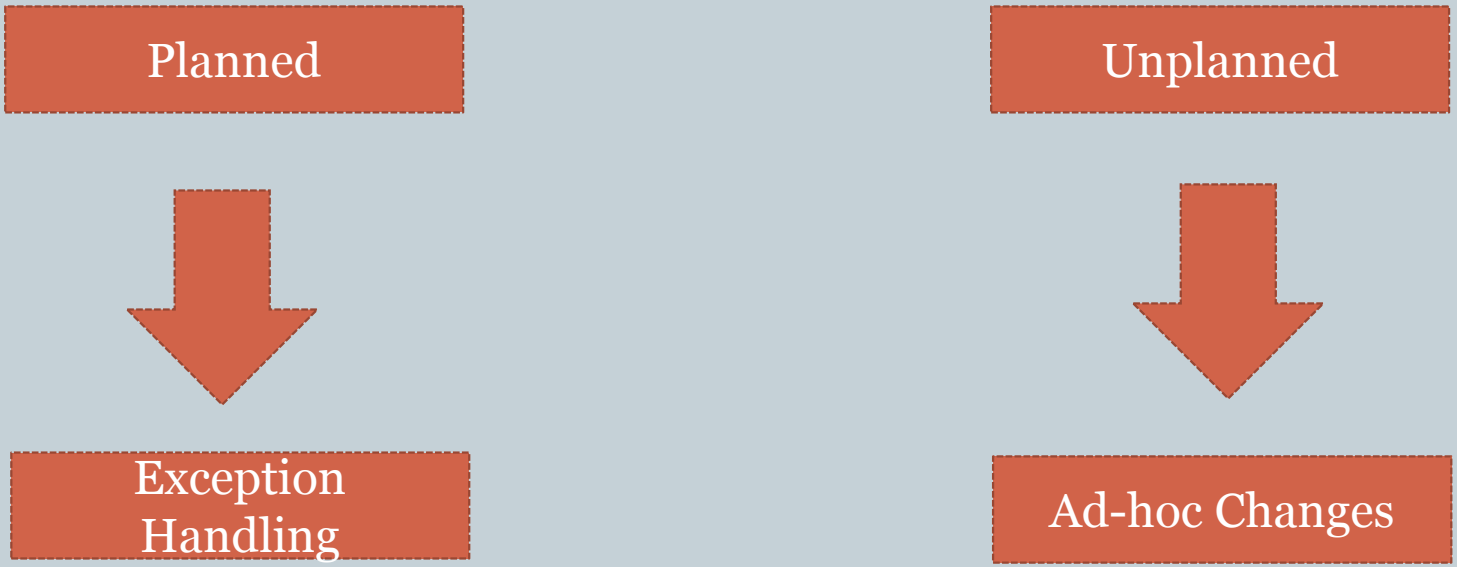


Example of a Configurable Model





Exception Handling



Handling Planned Exceptions in PAIS

I will skip techniques for handling planned execptions in this lecture and refer to my text book instead!

Sources

Activity

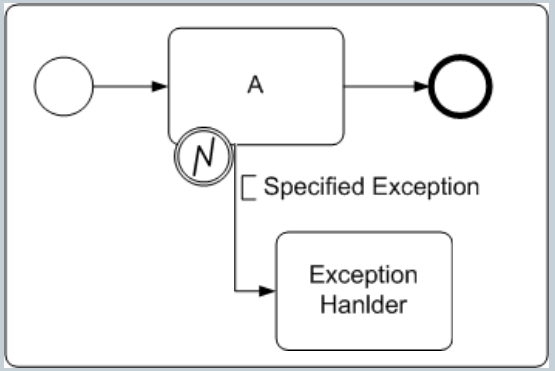
Technical | Semantical

Deadline Expiry

Resource Unavailability

Inconsistency real-world / PAIS

Constraint Violations



a suitable handler is chosen

dler

ves

nordered

Add Behavior

Deferred Fixing | Immediate Fixing

Retry | Exception-driven Rework

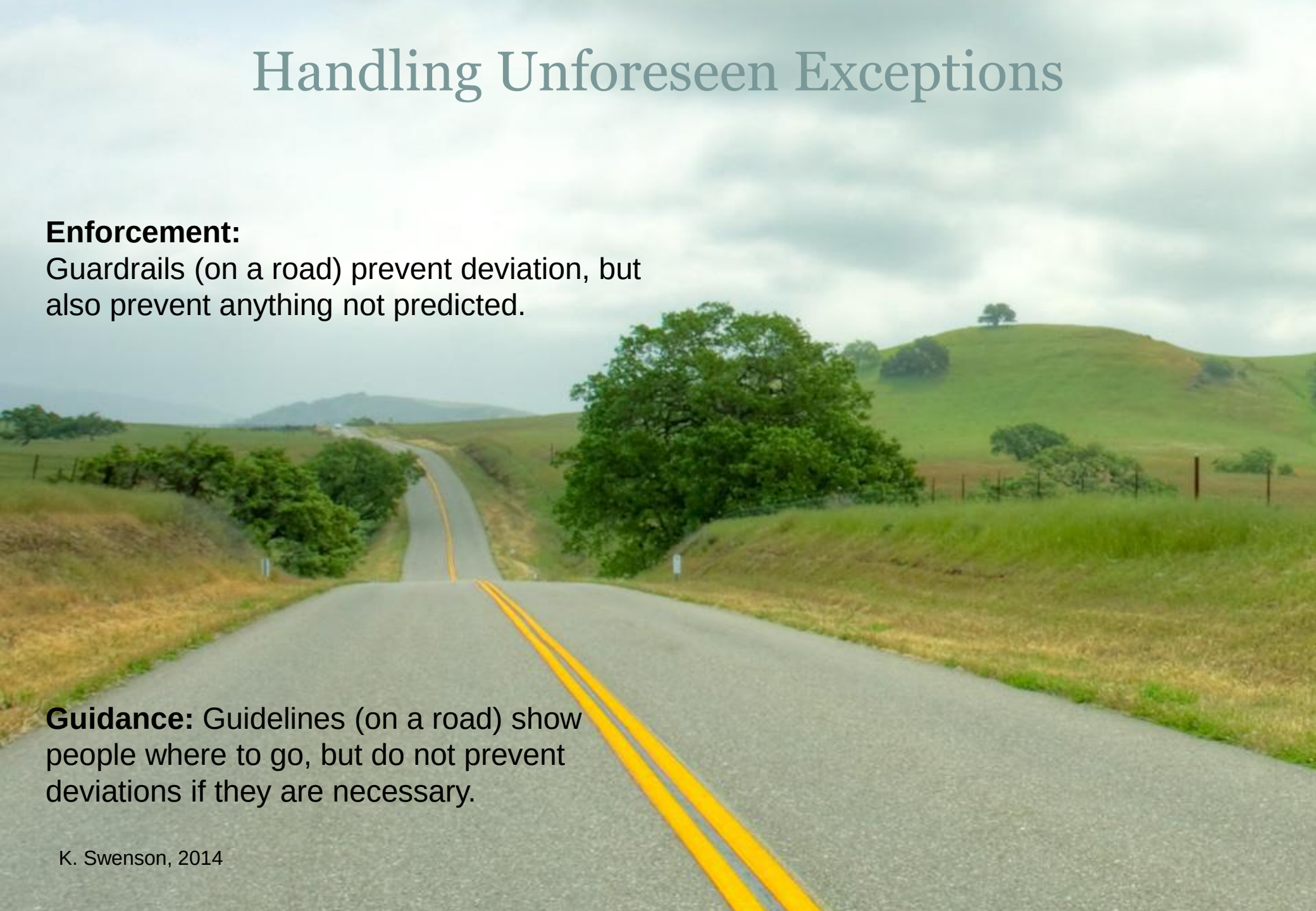
Cancelling Behavior

Reject | Compensate

Resource Patterns

Delegate | Escalate | ...

Handling Unforeseen Exceptions



Enforcement:
Guardrails (on a road) prevent deviation, but also prevent anything not predicted.

Guidance: Guidelines (on a road) show people where to go, but do not prevent deviations if they are necessary.

K. Swenson, 2014

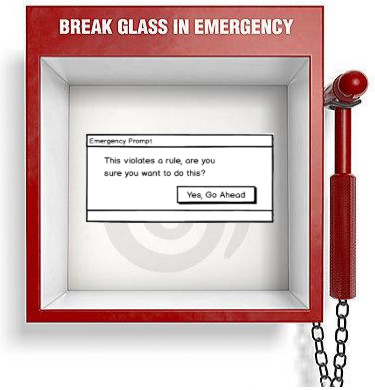
Handling Unforeseen Exceptions

"Planning is helpful. If you don't know what you want, you'll seldom get it. But, no matter how well you plan, you will fare better if you expect the unexpected. The unexpected, by nature, comes unseen, unthought, unenvisioned. All you can do is plan to go unplanned, prepare to be unprepared, make going with the flow part of your agenda, for the most successful among us envision, plan, and prepare, but cast all aside as needed, while those who are unable to go with the flow often suffer, if they survive."

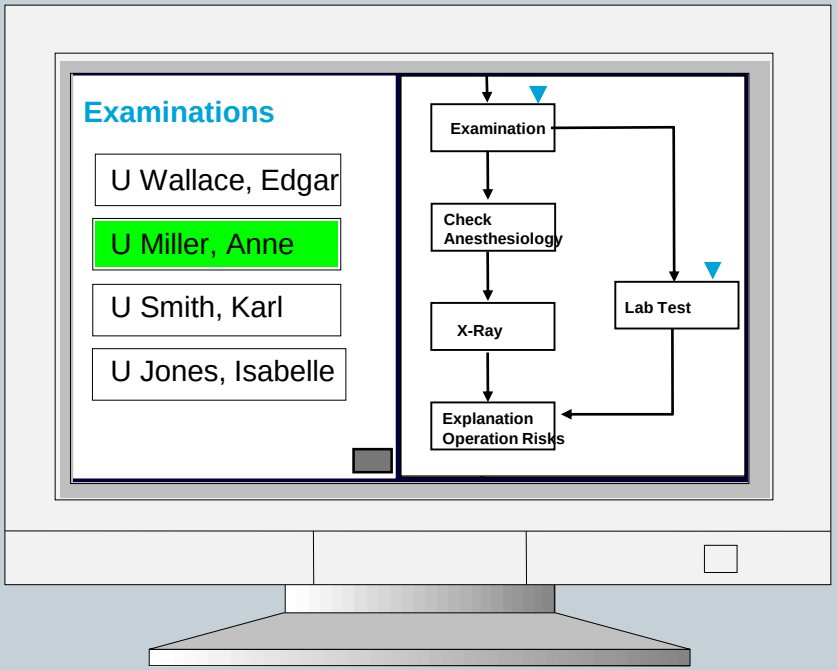
David W. Jones

This patient has a combination of symptoms that requires us to do something that has never been tried before!

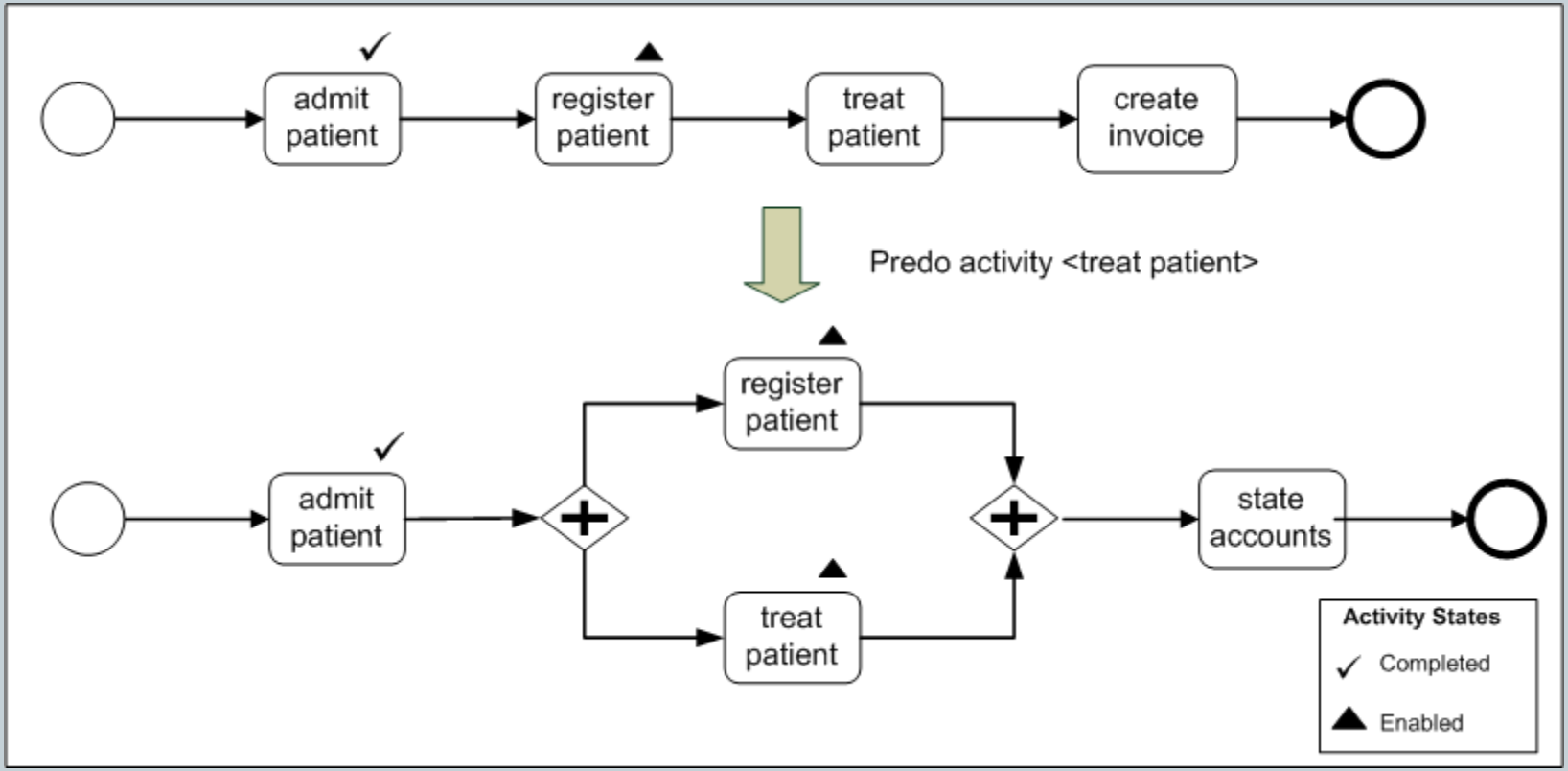
I'm sorry Dr. House, I can't allow you to do that. It would make the process invalid.



User View on an Ad-hoc Process Change



Behavioral Changes Require Structural Process Model Adaptations



Structurally Adapting Pre-Specified Process Models

Change Primitives

- Add node
- Remove node
- Add edge
- Remove edge
- …

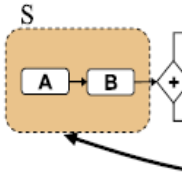
High-Level Change Operations

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

Structurally Adapting Pre-Specified Process Models

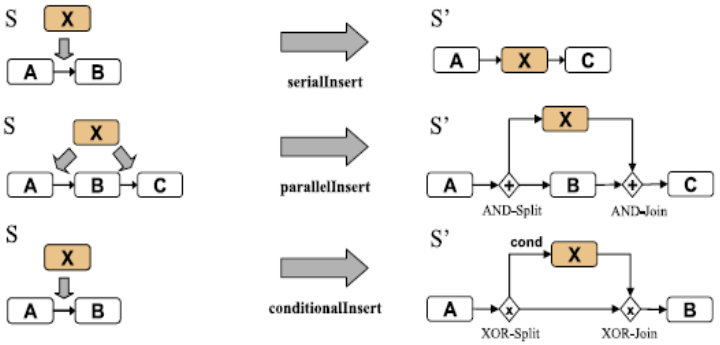
Pattern AP5: SWAP Process Fragment

Description	Two existing process fragments are swapped in process schema S.
Example	Regarding a particular delivery process the order in which requested g
	delivered to two customers has to be swapped
Problem	
Implementation	
Related Patterns	



Pattern AP1: INSERT Process Fragment

Description	A process fragment X is added to a process schema S.
Example	For a particular patient an allergy test has to be added to his treatment process due to a drug incompatibility.
Problem	In a real world process a task has to be accomplished which has not been modeled in the process schema so far.
Design Choices (in addition to those described in Fig. 6)	C. How is the new process fragment X embedded in the process schema? 1. X is inserted between two directly succeeding activities (`serial insert`) 2. X is inserted between two activity sets (`insert between node sets`) a) without additional condition (`parallel insert`) b) with additional condition (`conditional insert`)



Implementation	This adaptation pattern can be realized by transforming the high level insertion operation into a sequence of low level change primitives (e.g., add node, add edge).

Pattern PP3: Late Composition of Process Fragments

Description	At build-time a set of process fragments is defined from which the schema of a concrete process instance can be composed during run time. This can be achieved by dynamically l by specifying the control dependencies between them on the fly.

cal examinations are accomplished in a hospital. The exact
lied to a particular patient and the order in which they are performed are
: individually depending on his/her medical problems.

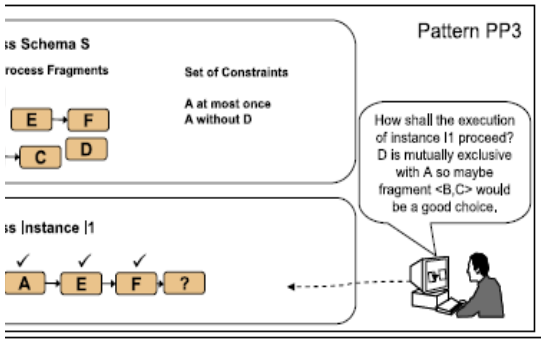
ants of how process fragments can be composed. To reduce the number
e specified by the process engineer during build time, process instances
sed from a given set of fragments.

: building blocks for late modeling?

agments from the repository can be chosen.

based subset of the process fragments from the repository can be

s or process fragments can be defined.

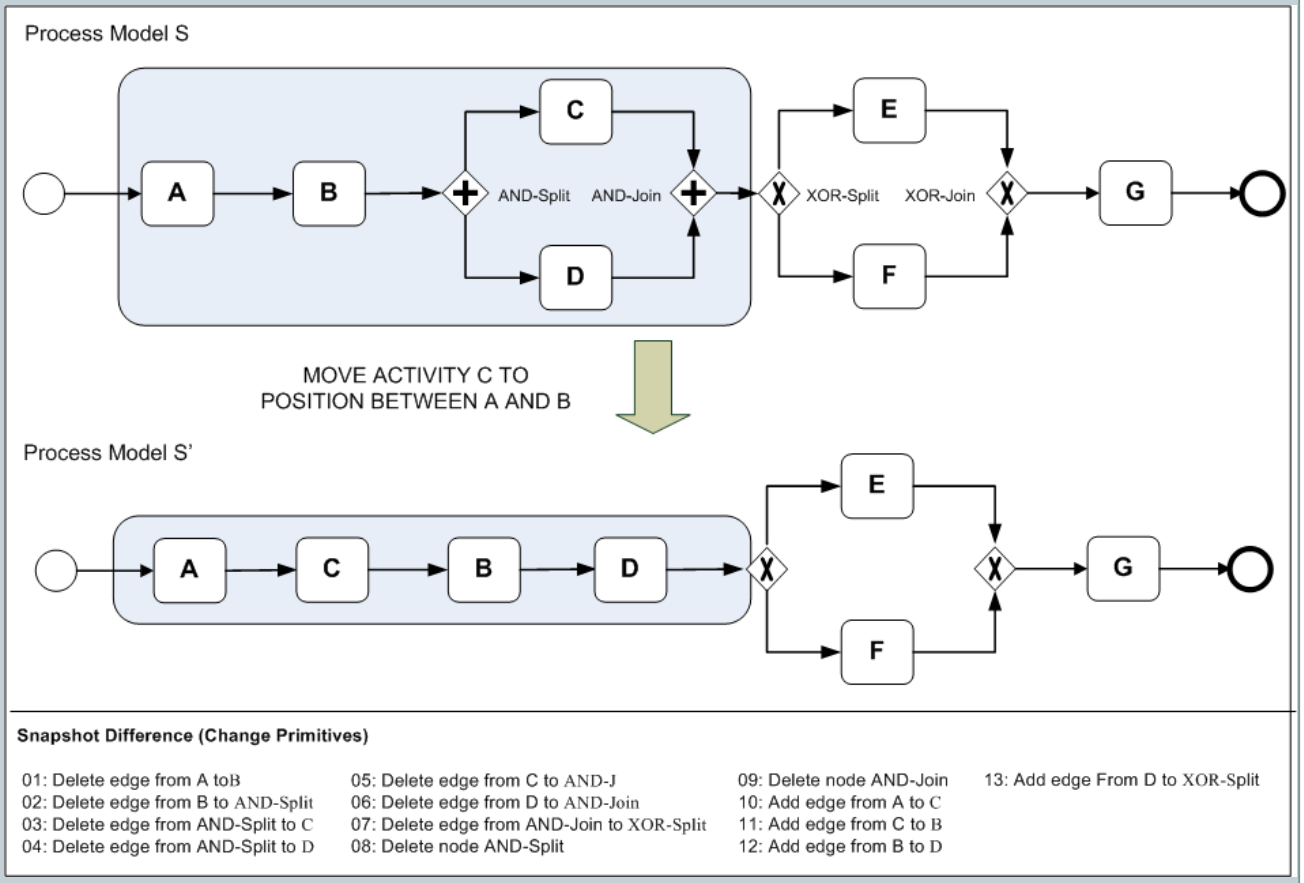


Structurally Adapting Pre-Specified Process Models

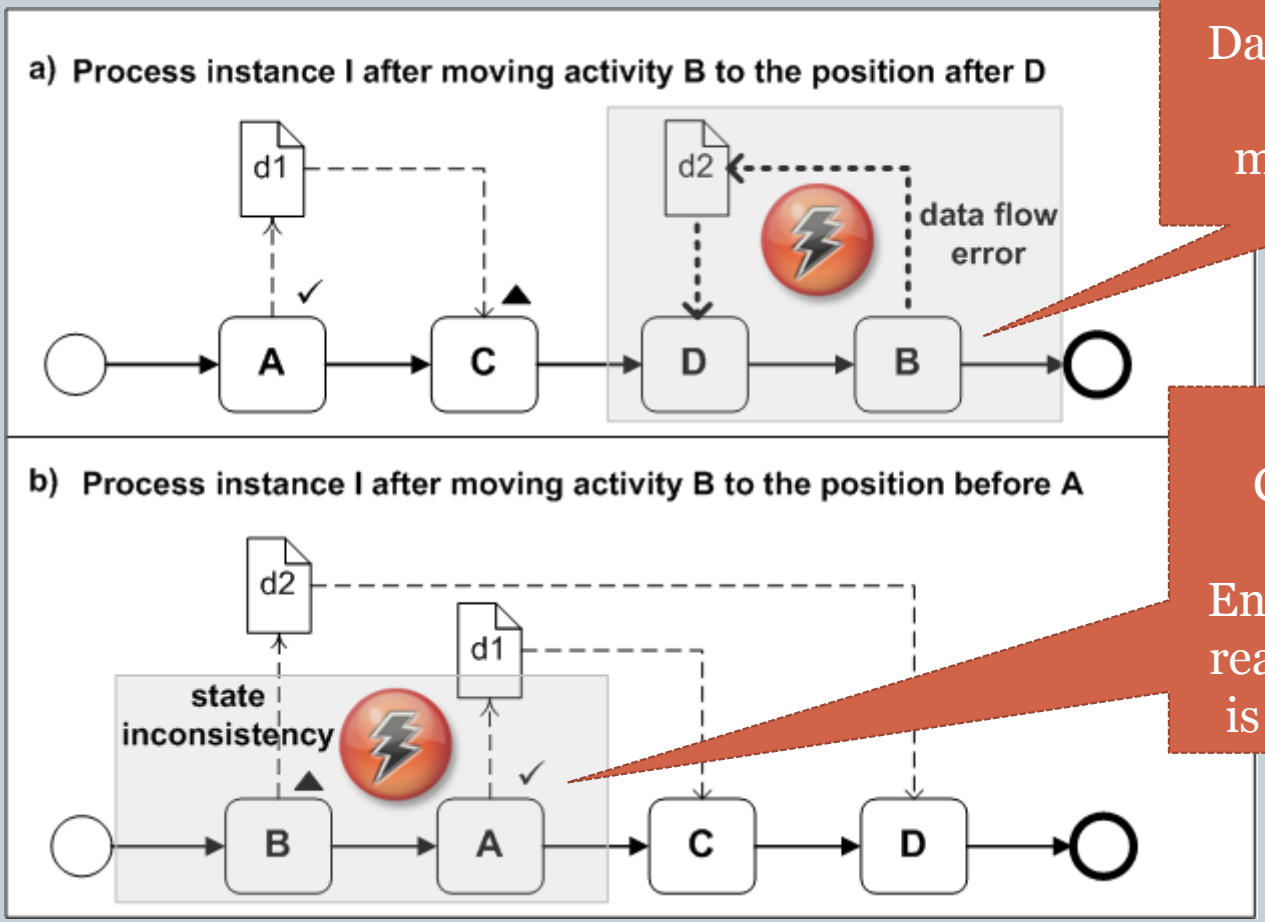
Adaptation Patterns

vs.

Change Primitives



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

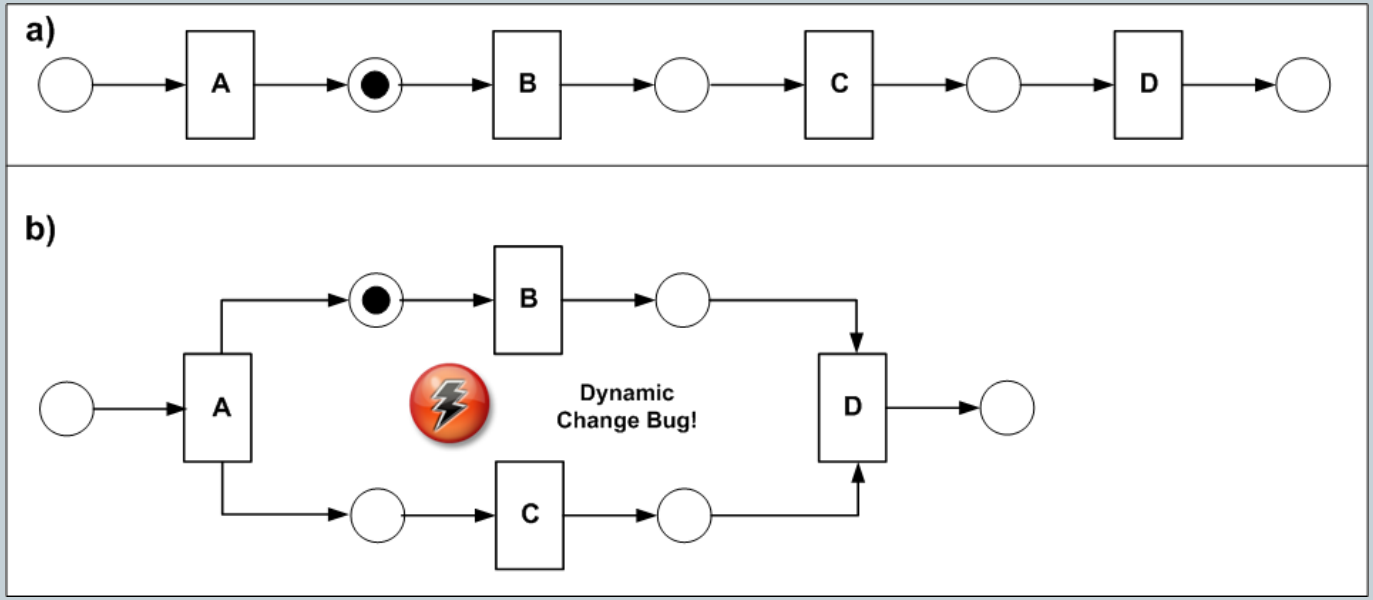


Data flow error caused by missing data

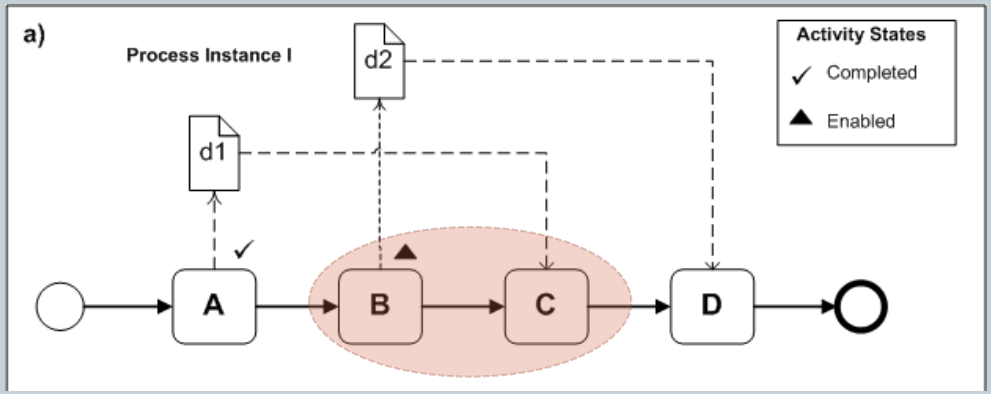
No Proper Completion ensured. End node can be reached while B is still enabled

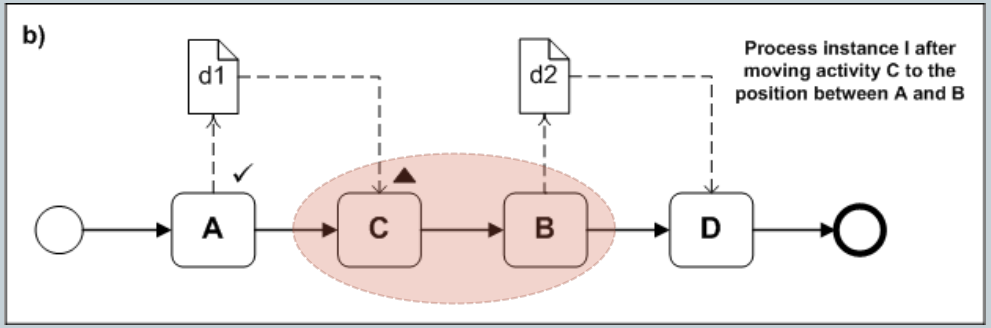
Behavioral Changes Require Adaptations of the Process Instance State

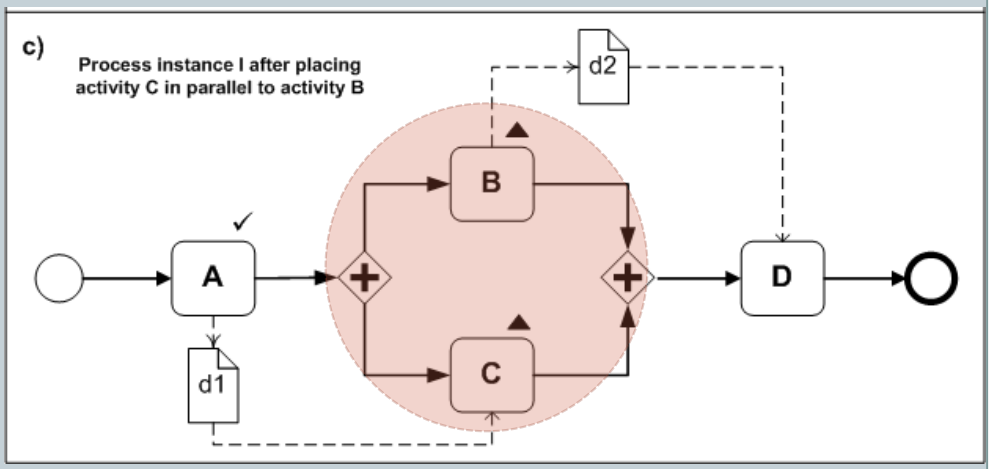
Dynamic Change Bug



Behavioral Changes Require Adaptations of the Process Instance State

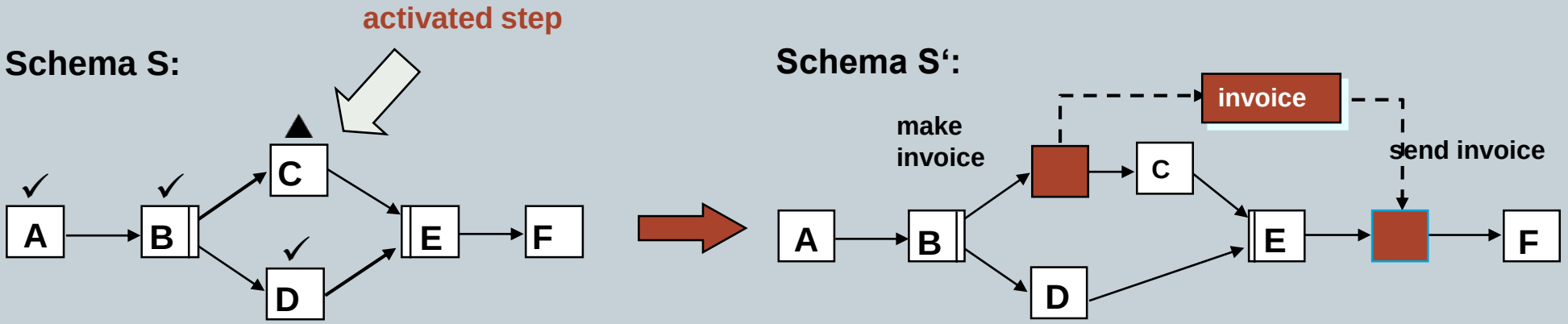






Behavioral Changes Require Adaptations of the Process Instance State

Ensuring Dynamic Correctness



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

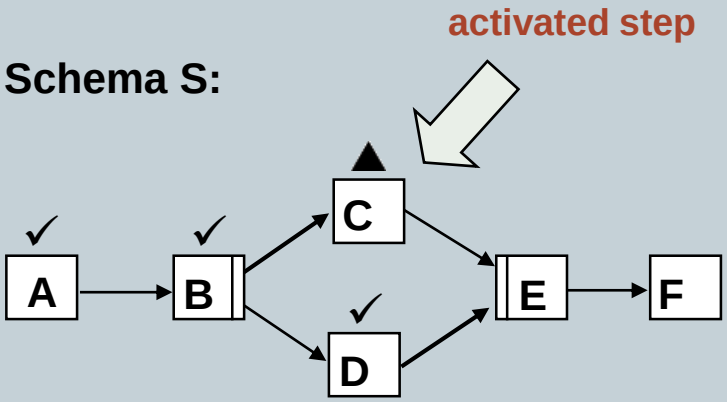
Need for general correctness criterion

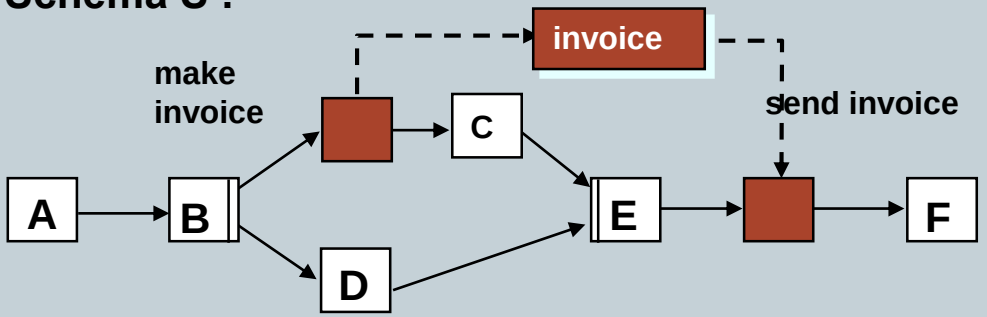
⇨State Compliance

[ReDa98, RRW08a, RRD04a, RRD04b]

Behavioral Changes Require Adaptations of the Process Instance State

Ensuring Dynamic Correctness





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

More complicated: loop backs

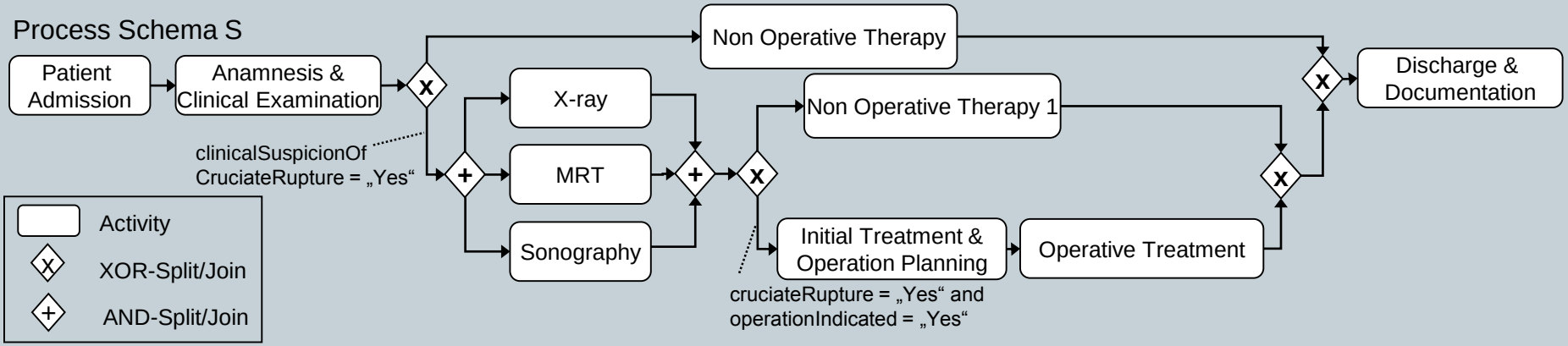
Further challenges: - How to efficiently check for compliance?
- How to efficiently migrate process instances?

[RRD04a, RRD04b]

Correctness of Process Instance Changes

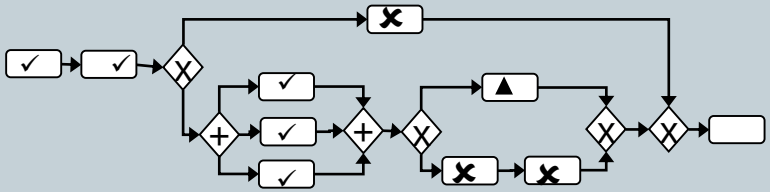
Process Type Level

Process Schema S



Process Instance Level

Process Instance I3

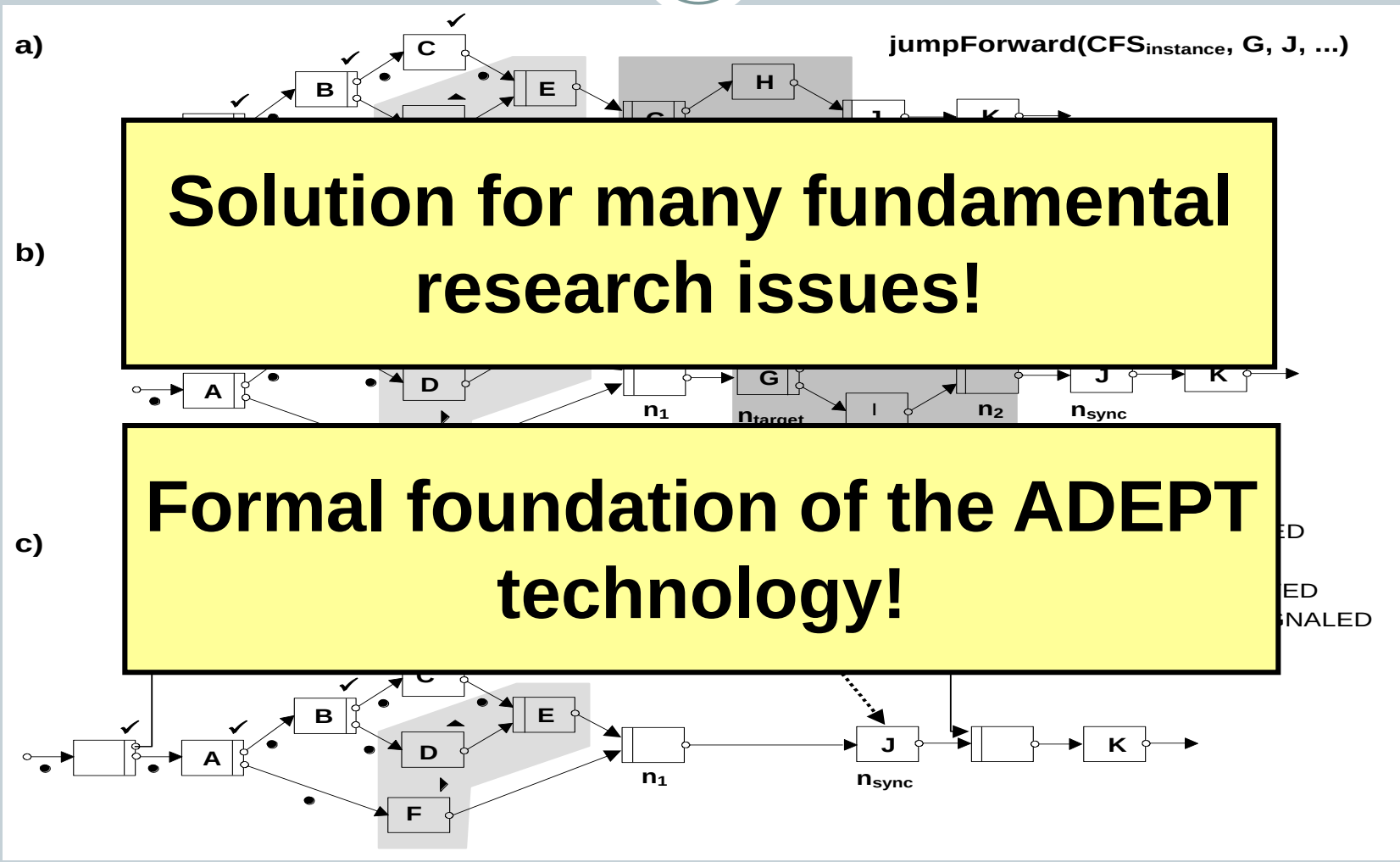


Execution Trace:
σ_3 = < „Patient Admission“, „Anamnesis & Clinical Examination“, „MRT“, „X-ray“, „Sonography“>

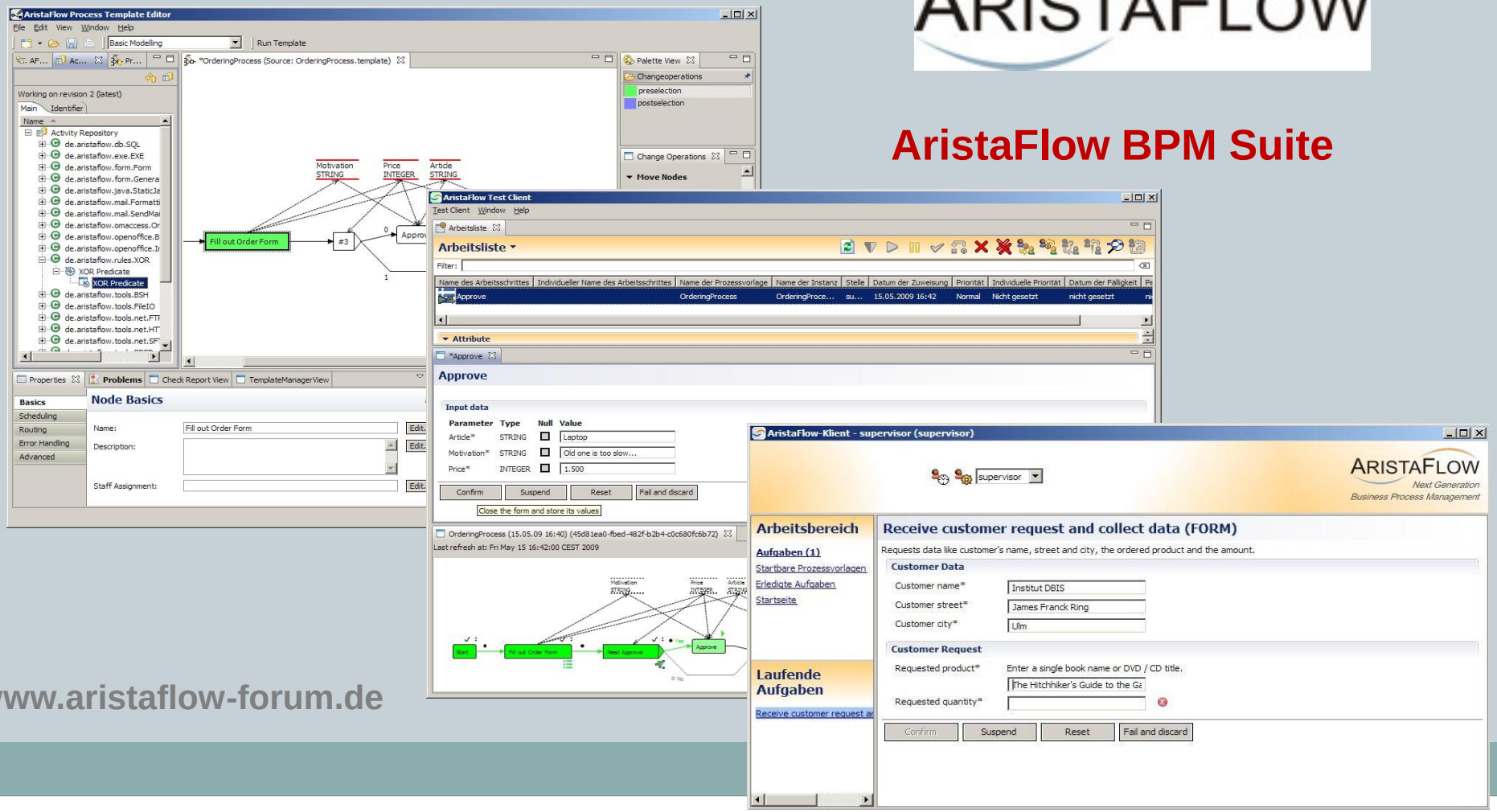
I3 is not state compliant with change Delete (I3, MRT)

[ReDa98]

The ADEPT Framework



… and its transfer to industrial practice

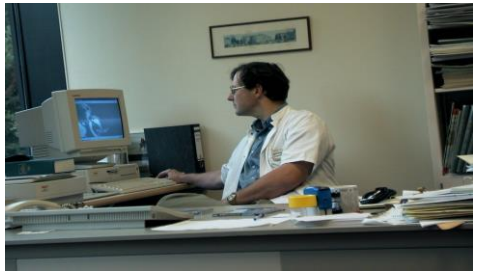


ARISTAFLOW

AristaFlow BPM Suite

www.aristaflow-forum.de

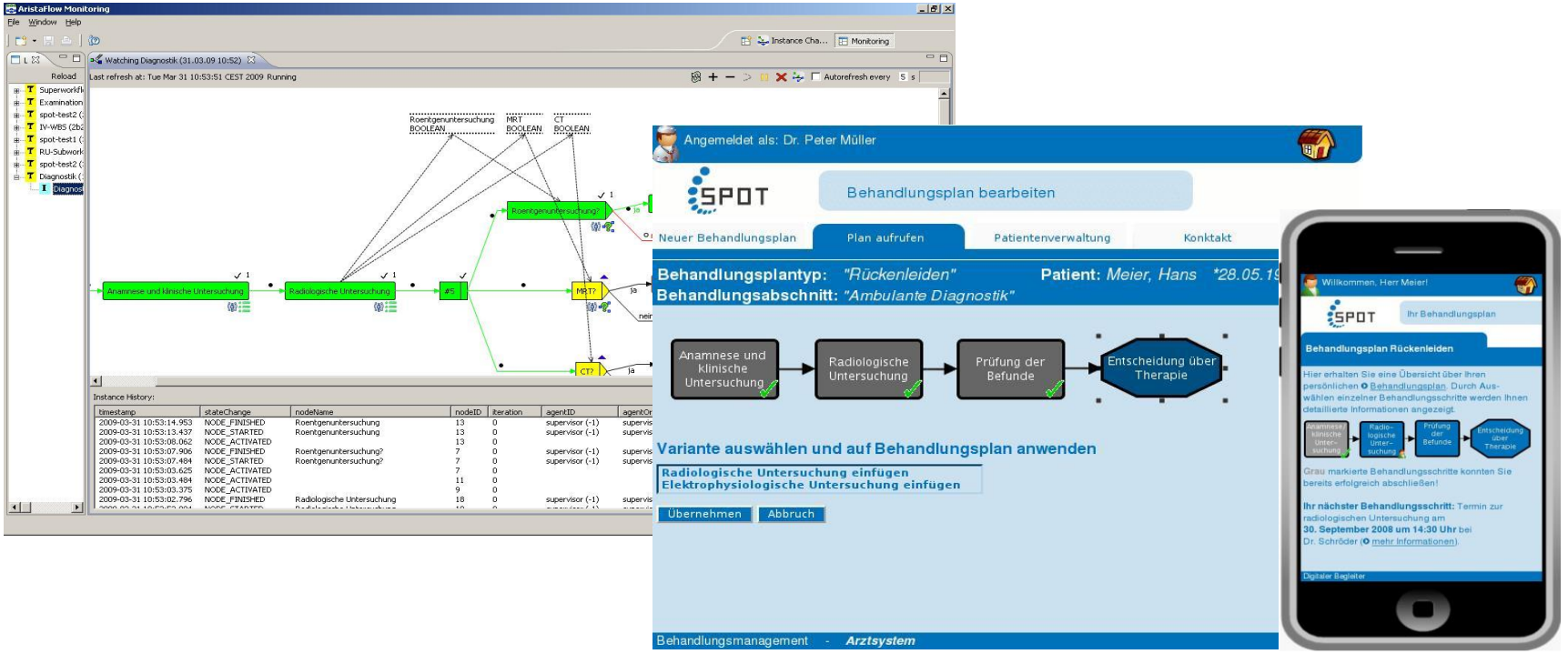
Application Example I



Flexible Support of Clinical Pathways with AristaFlow

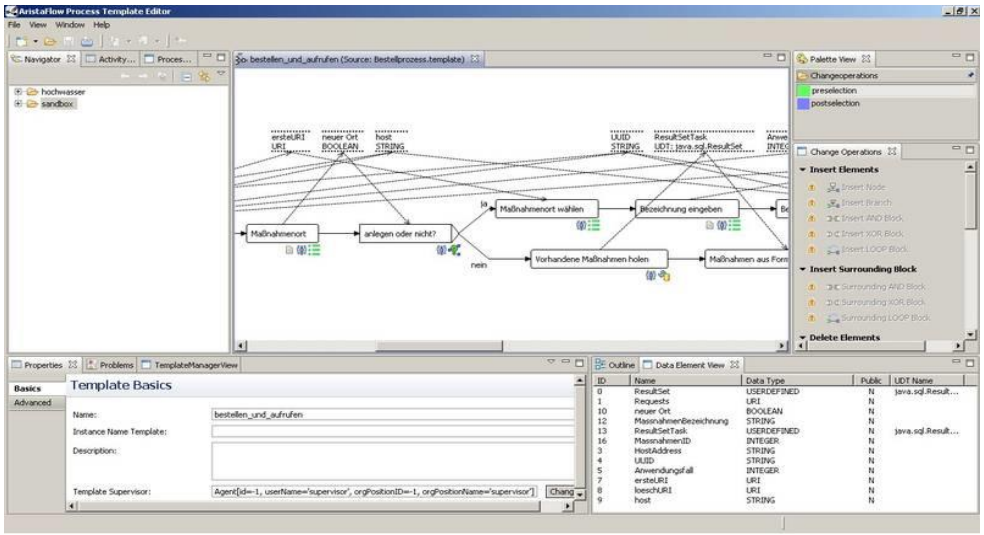
Partners:

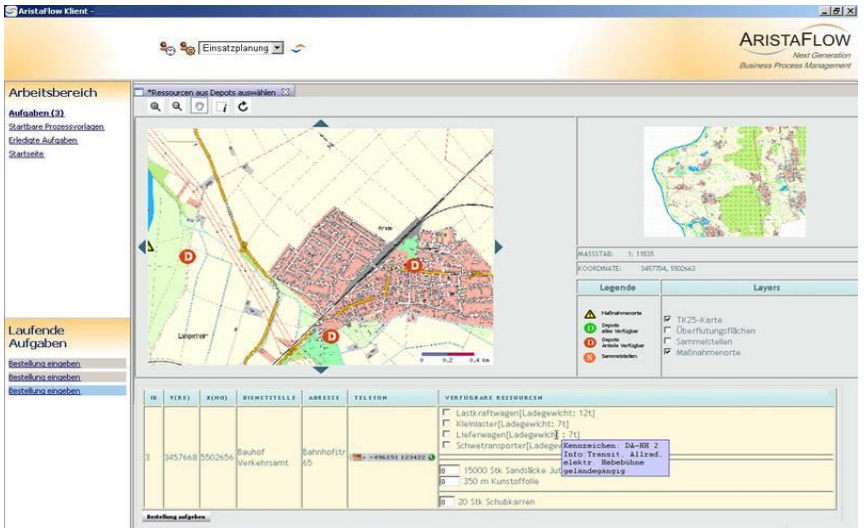
Jan Neuhaus, Claudia Reuter
Fraunhoferinstitut Dortmund



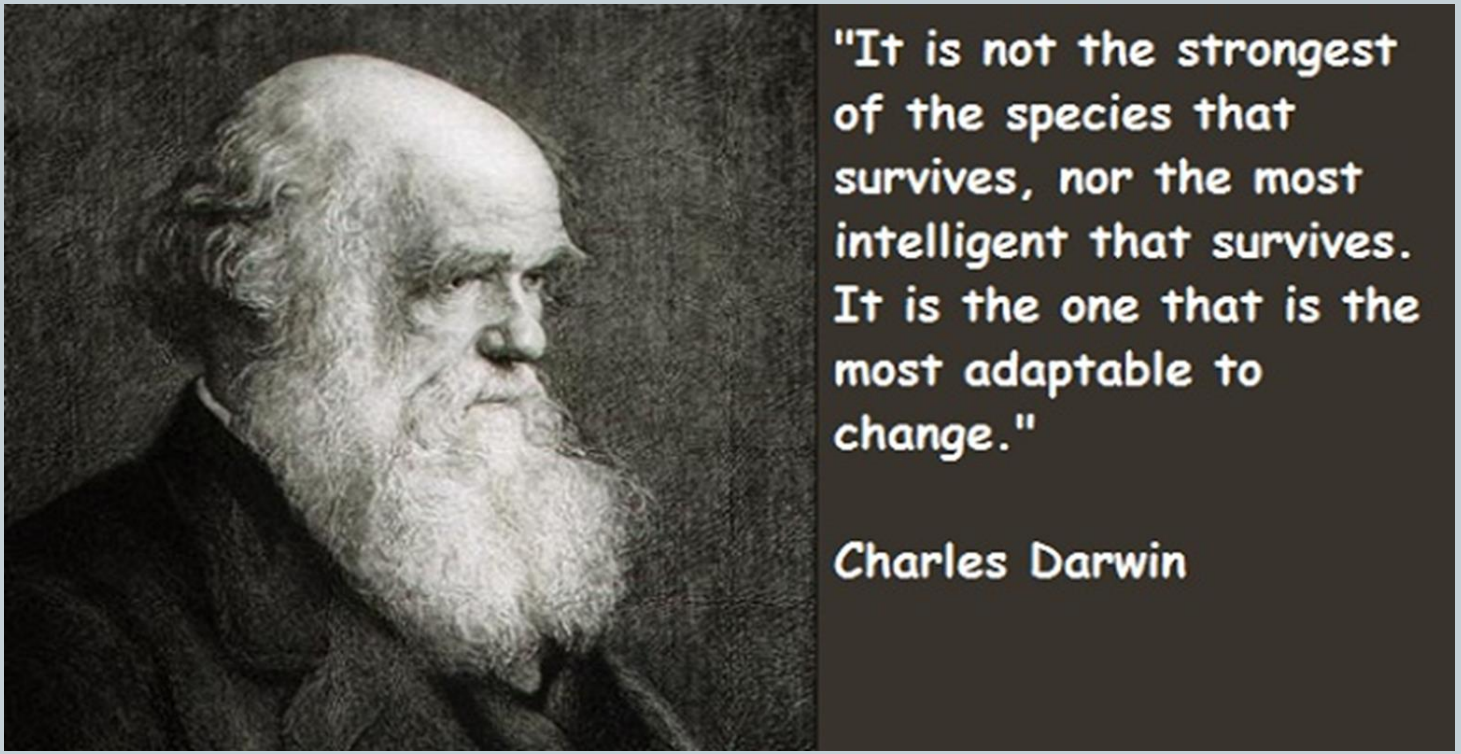
Application Example I

Process-aware, Cooperative Emergency Management for Water Infrastructures
Partner: TU Darmstadt



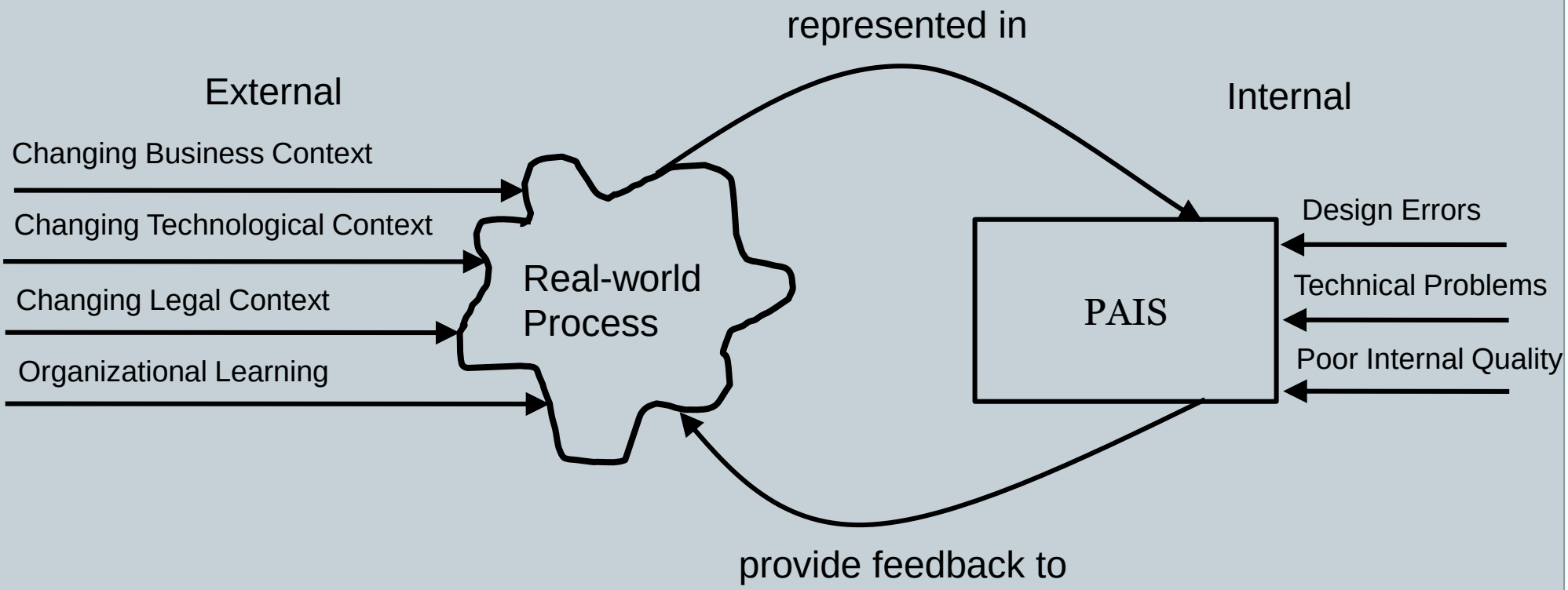


Process Evolution

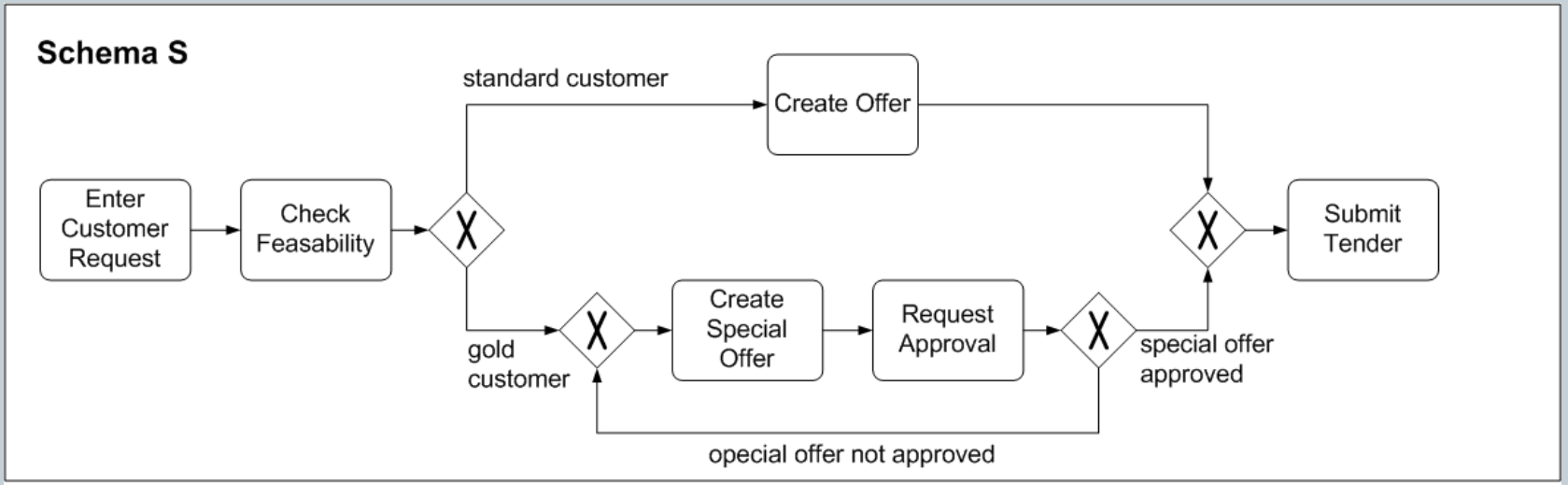


Process Evolution

Drivers



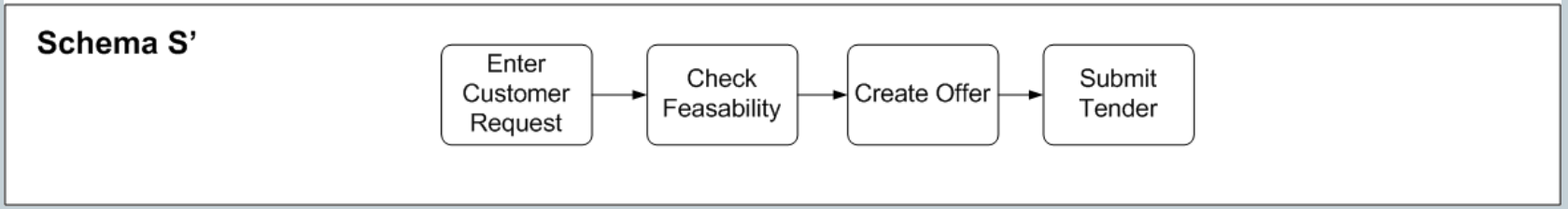
Process Evolution



S evolves to S'

by applying change Δ_S with

Δ_S = <Delete(S, Create Special Offer), Delete(S, Request Approval)>



Change Support Features
Schema Evolution, Version Control and Instance Migration

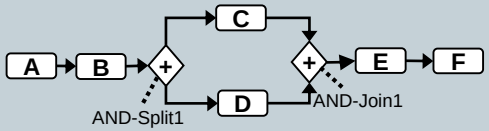
- Schema Evolution
  - Changes at the process type level
- How to deal with running instances when adapting the original process schema?
  - Scenario 1: No version control
  - Scenario 2: Co-existence of instances of old / new schema
  - Scenario 3: Change propagation and instance migration

Scenario 1: No Version Control

- Schema is overwritten and instances are migrated

Type change overwrites schema S

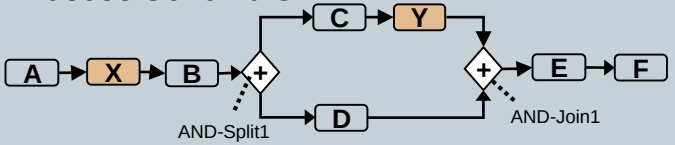
Process Schema S



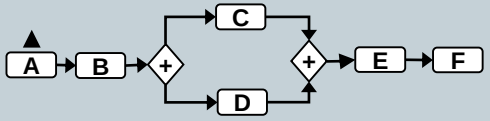
Insert X between A and B
Insert Y between C and AND-Join1

Schema Evolution

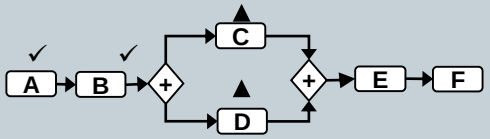
Process Schema S'



Process Instance I1

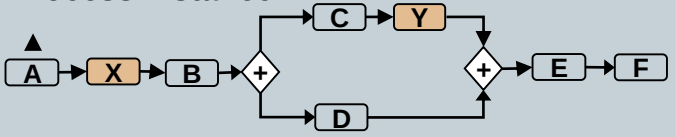


Process Instance I2

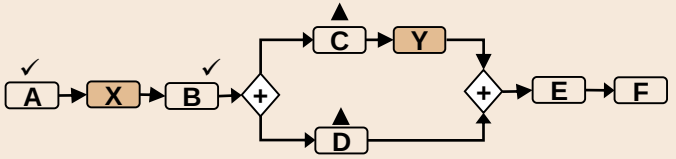


Change is propagated to all running process instances

Process Instance I1



Process Instance I2



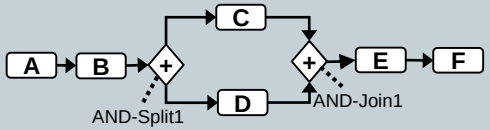
Inconsistent state

Scenario 2: Version Control

- Co-existence of instances of different schema versions

Type change results into a new version of schema S

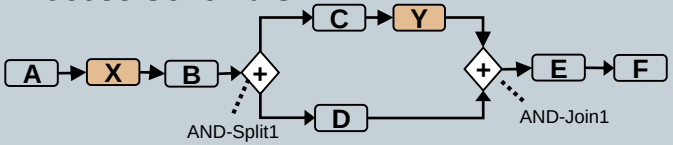
Process Schema S



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

Schema Evolution

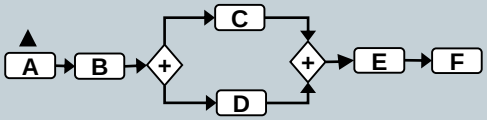
Process Schema S'



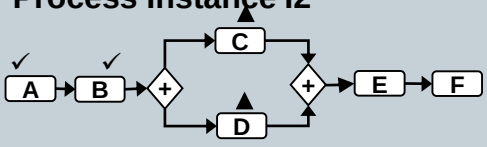
Old instances remain with schema S

Instances created from S (before schema evolution)

Process Instance I1

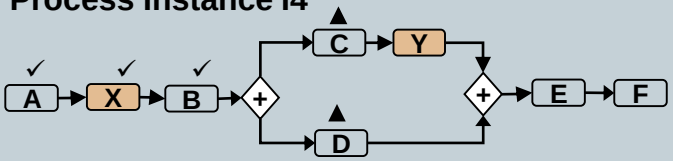


Process Instance I2

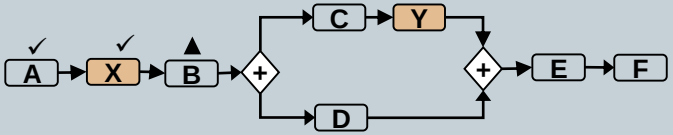


Instances created from S' (after schema evolution)

Process Instance I4



Process Instance I5

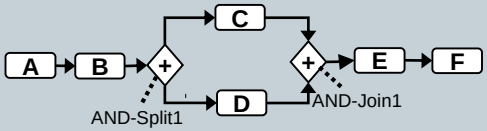


Scenario 3: Instance Migration

- Compliant instances are migrated to the new schema

Type change results into a new version of schema S

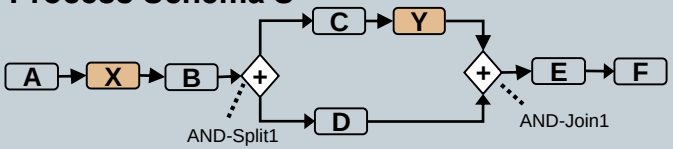
Process Schema S



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

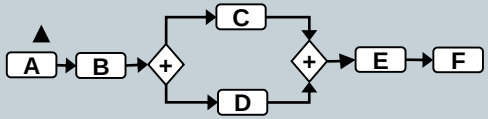
Schema Evolution

Process Schema S‘

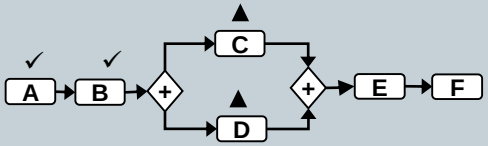


Migration of compliant process instances to S’

Process Instance I1

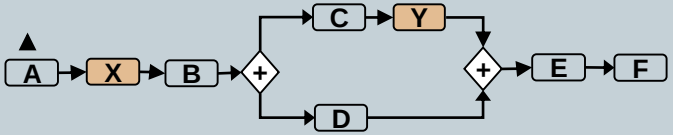


Process Instance I2



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

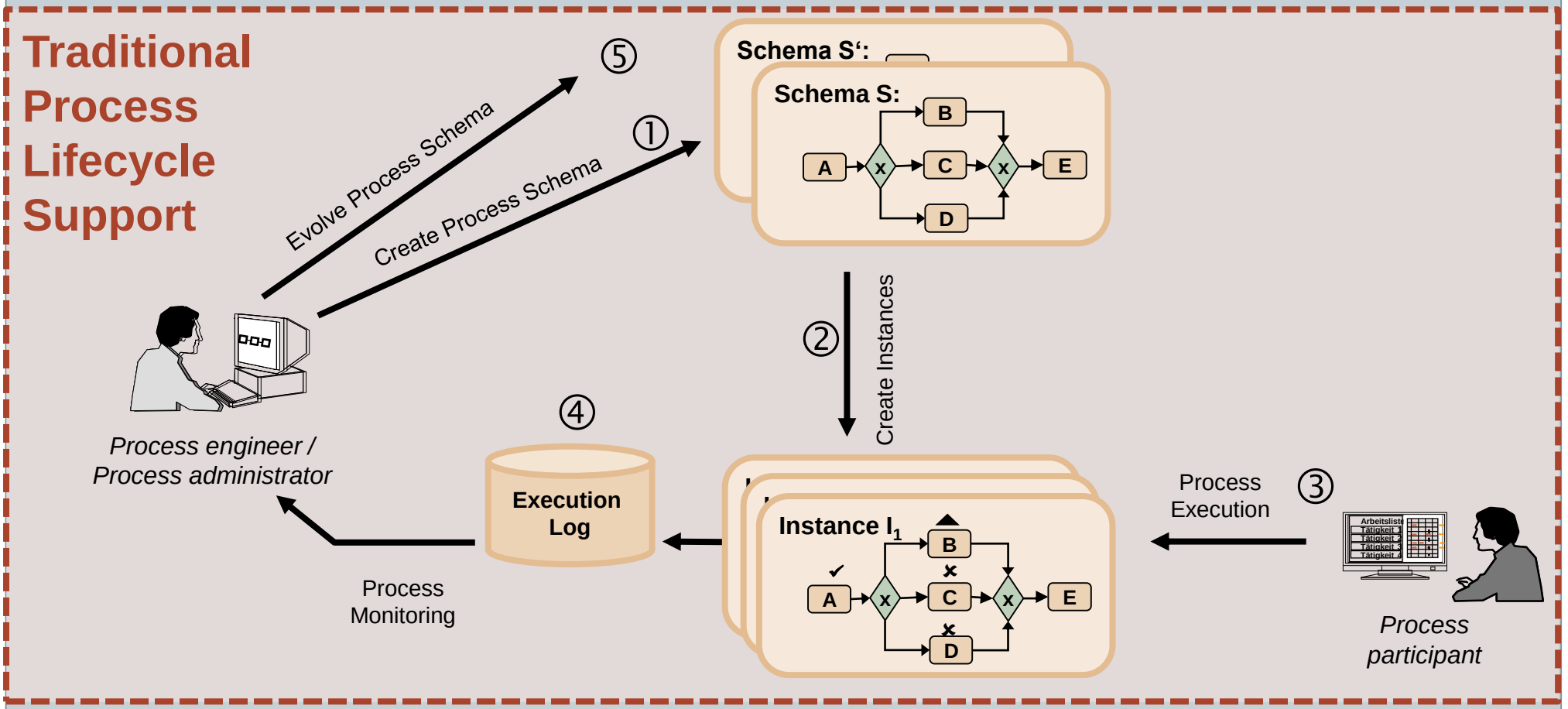
Process Instance I1



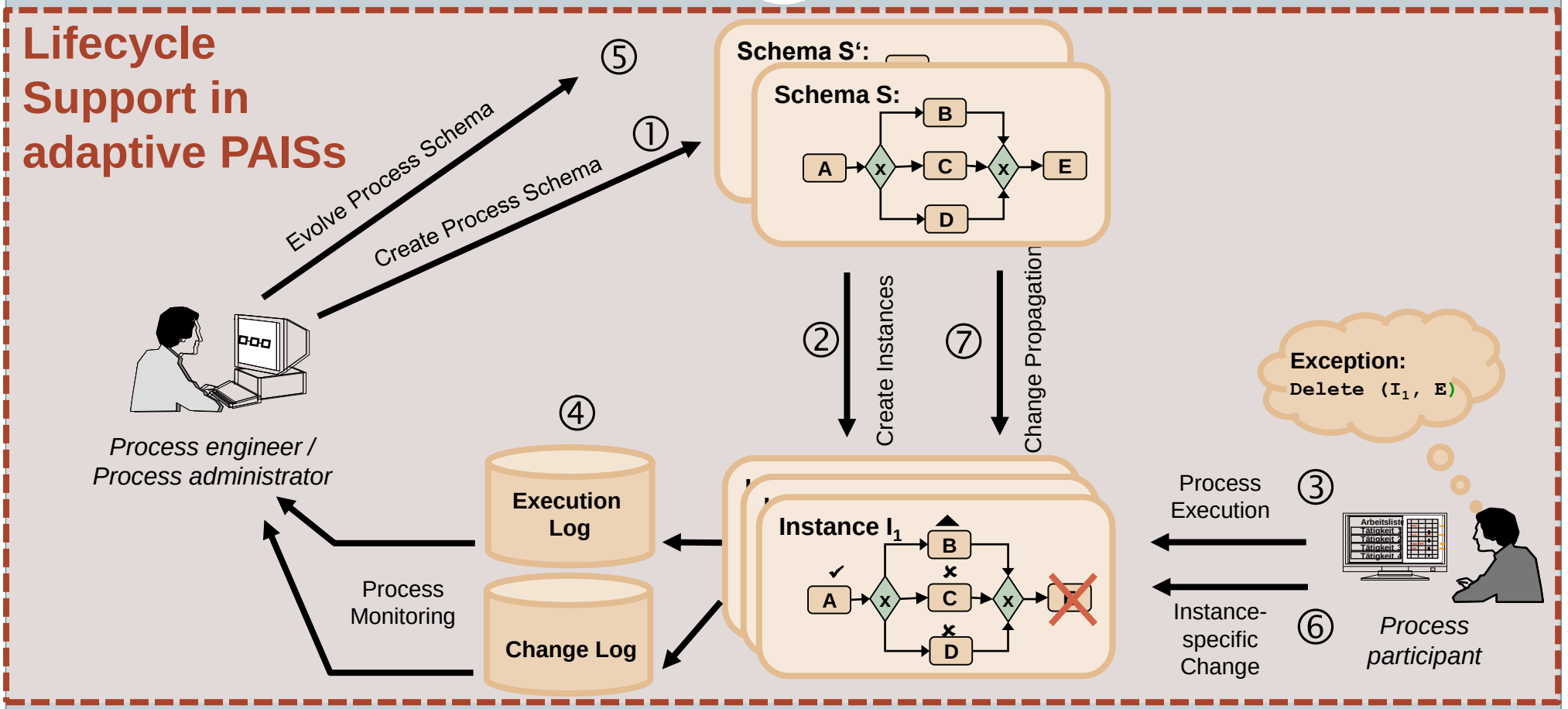
Process Instance I_2 not compliant with S’

[RRD04a]

Integrated Lifecycle Support for Adaptive and Dynamic Processes (1)

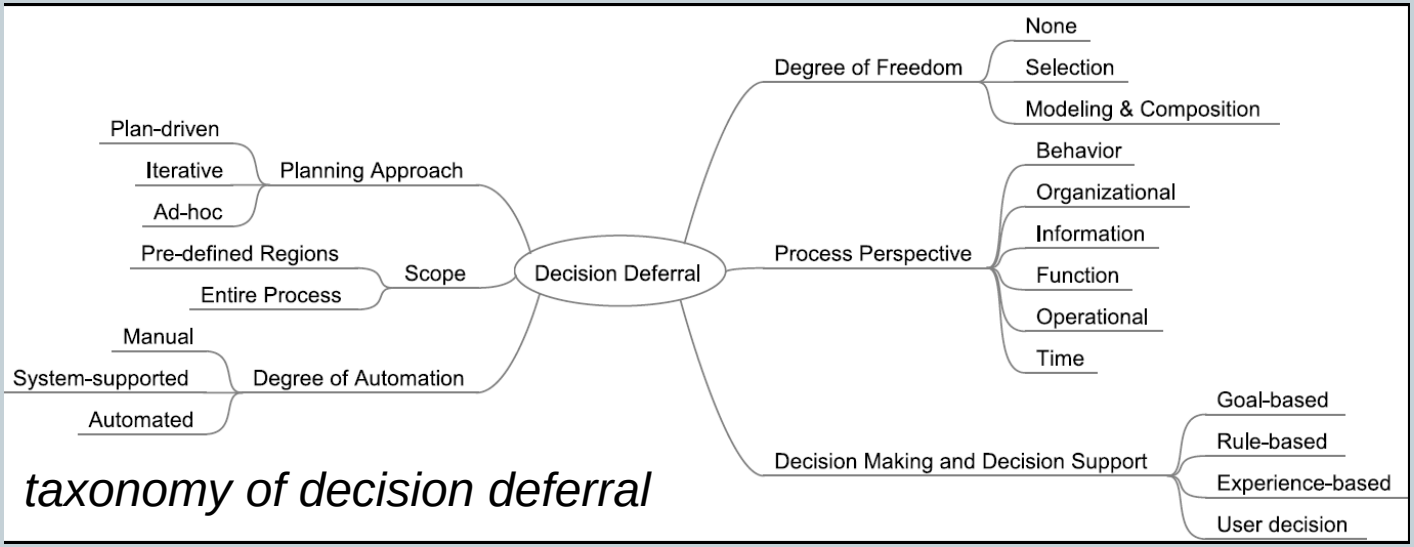


Integrated Lifecycle Support for Adaptive and Dynamic Processes (2)



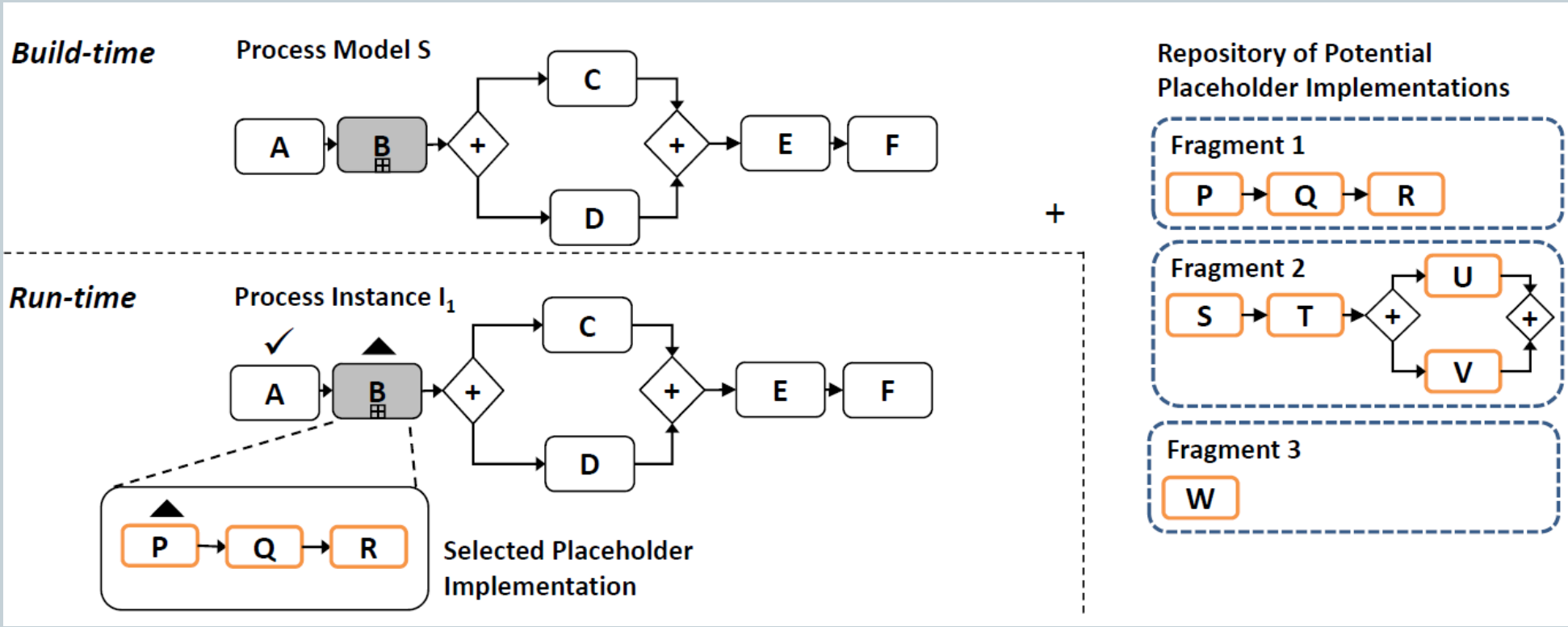
Loosely Specified Processes

- To deal with unpredictability, non repeatability and emergence loosely specified processes keep (parts) of the process unspecified during build-time

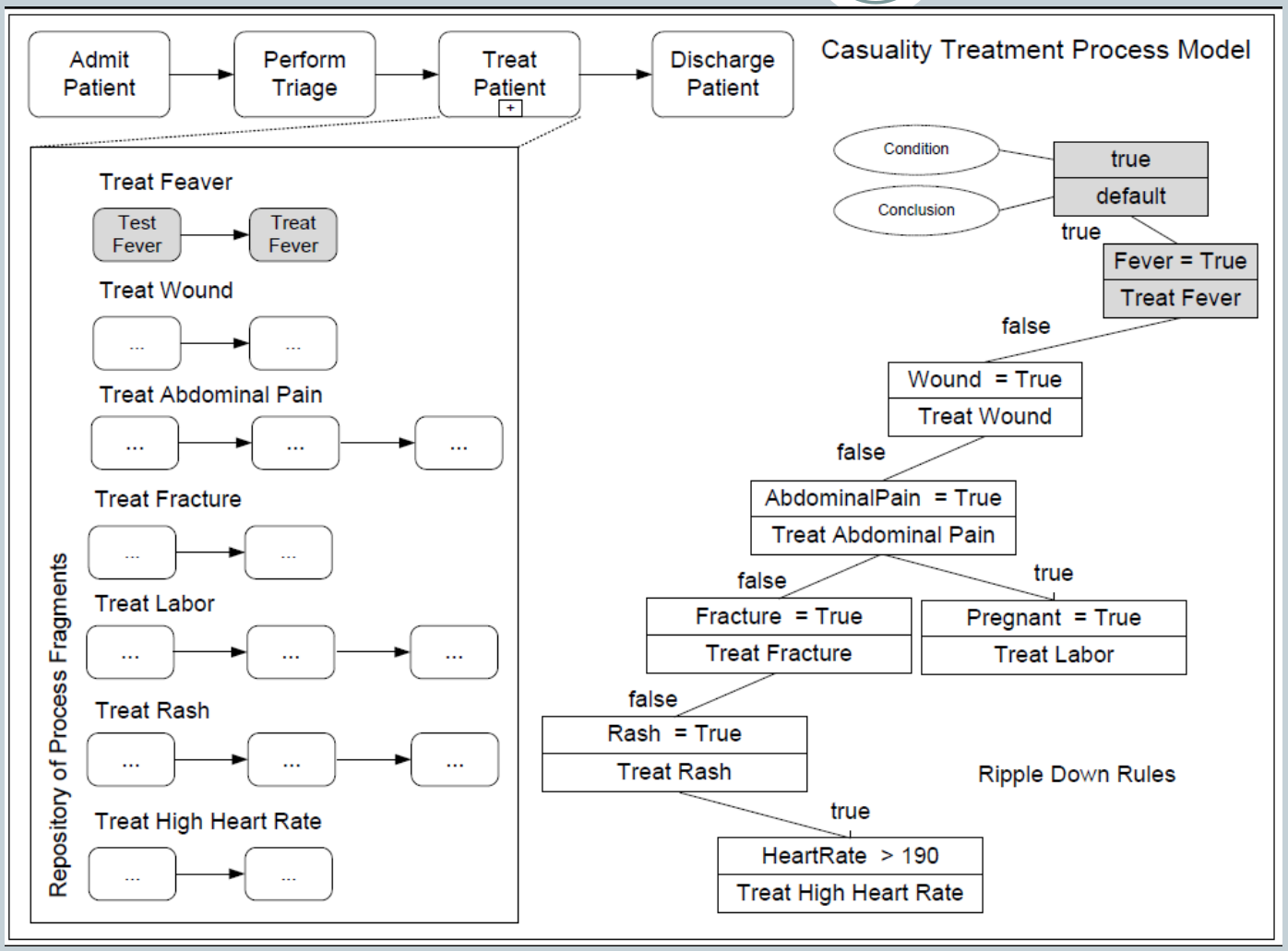


taxonomy of decision deferral

Late Selection Pattern

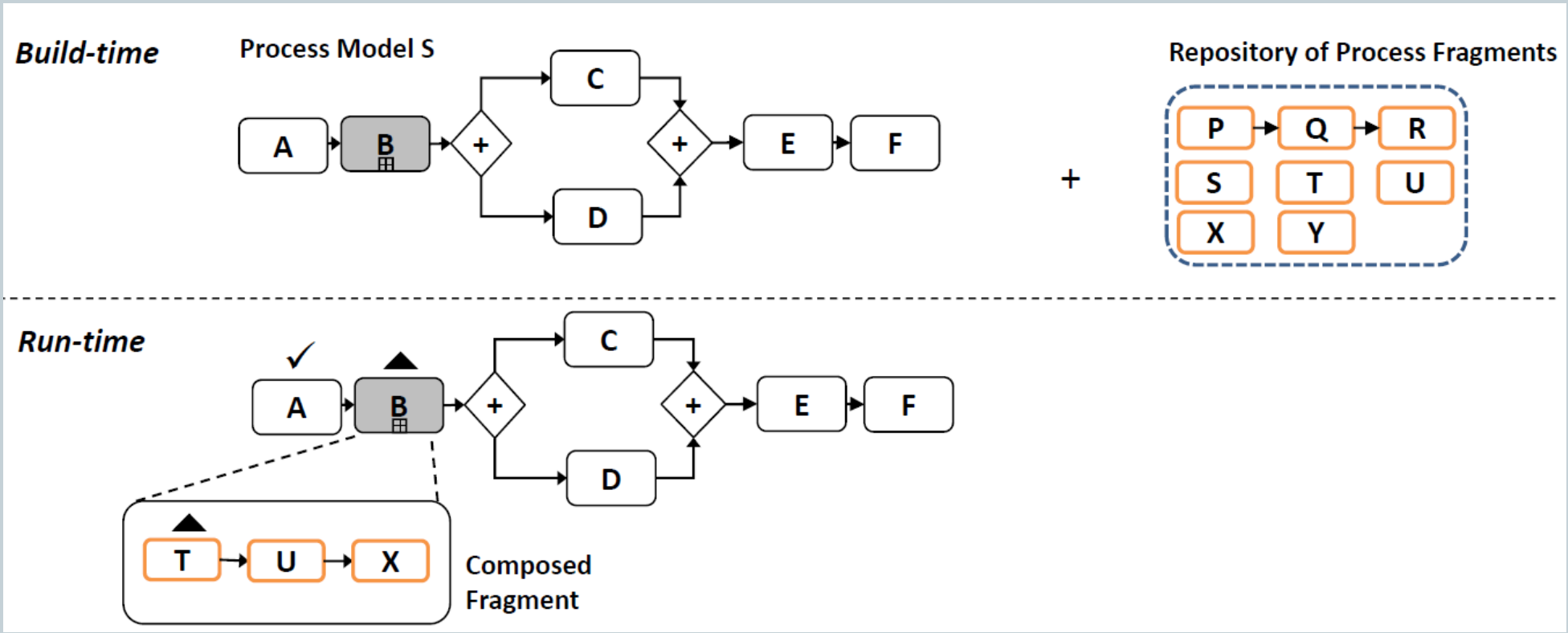


Late Selection – The Worklets Approach

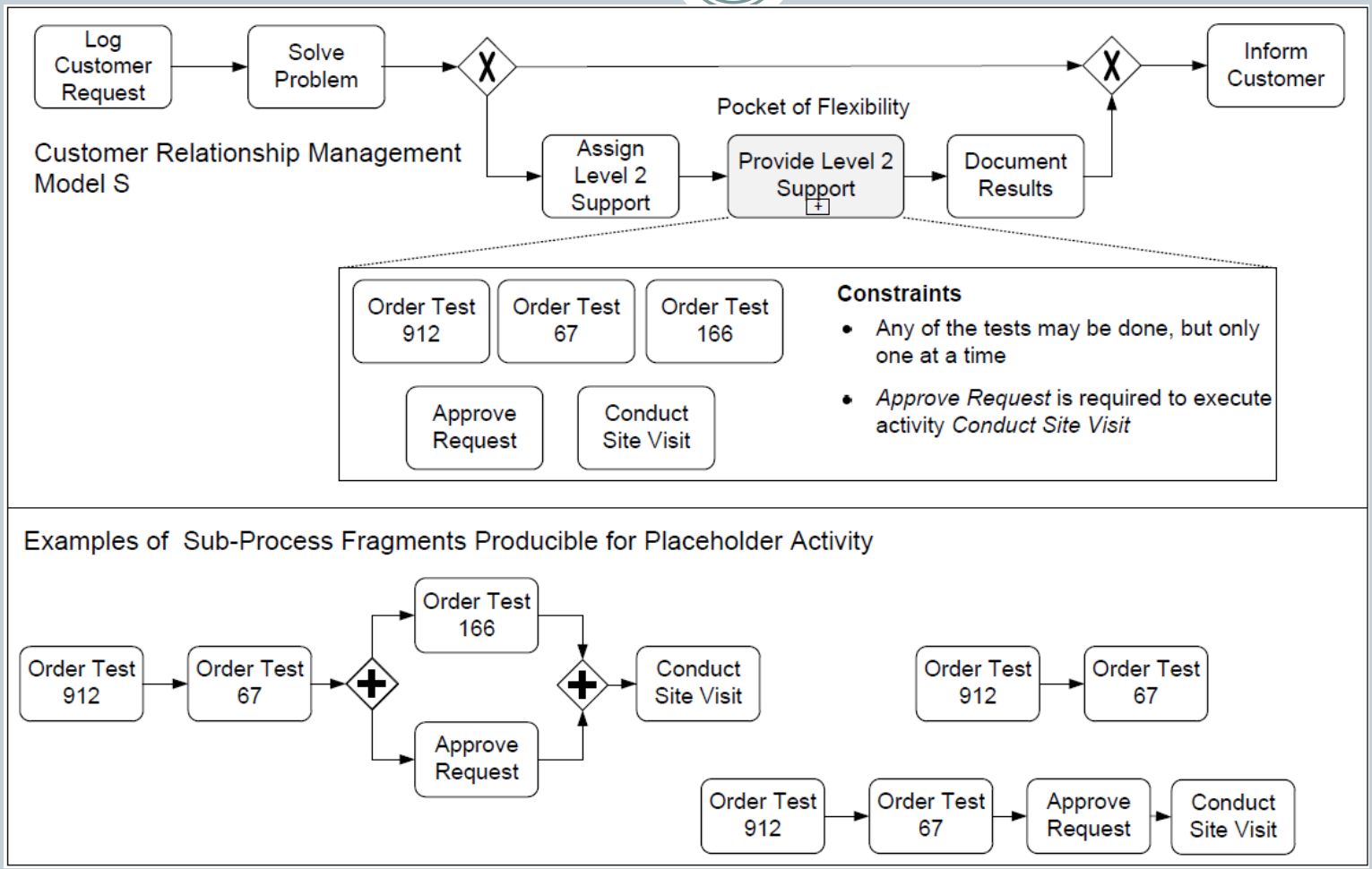


[AHE+06]

Late Modeling

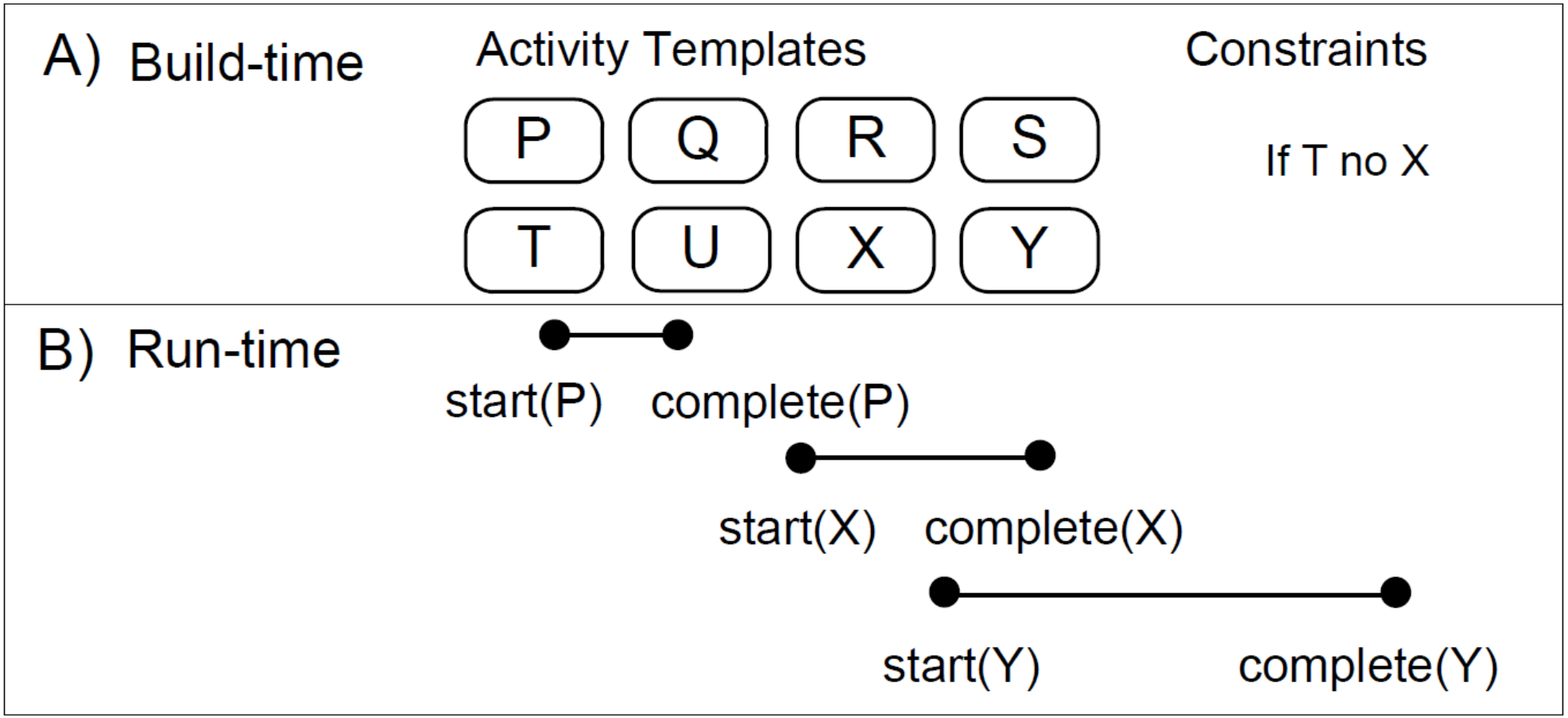


Late Modeling – Pockets of Flexibility



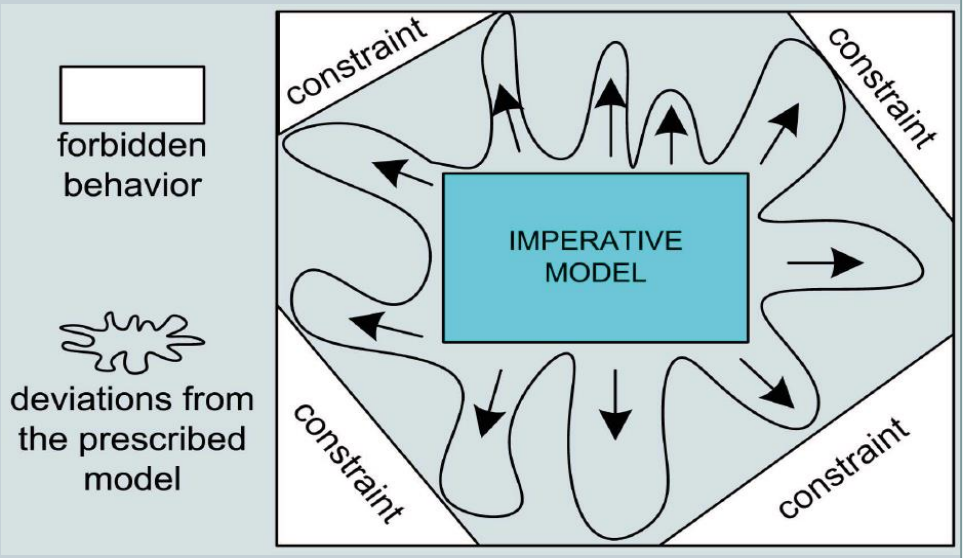
[SSO01, SSO05]

Ad-hoc Composition - Declare



Declarative Processes

- Instead of describing exactly how a business process should be executed, declarative processes
  - describe the activities to be executed
  - constraints prohibiting undesired behavior



[PSSA07]

Modeling Declarative Processes

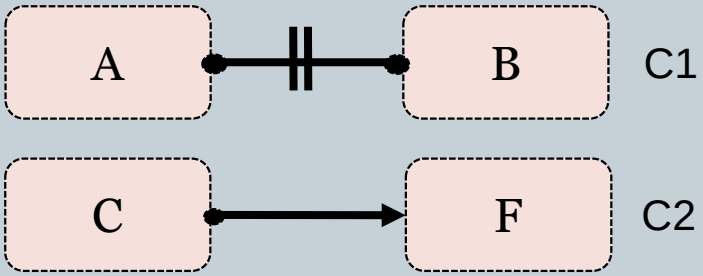
Declarative Process Model S

Activities A

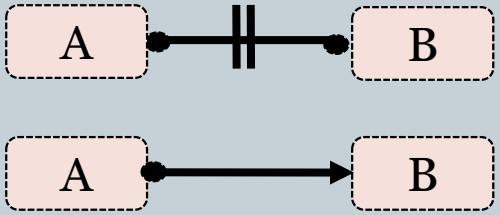
A B C

D E F

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:

σ_1 = < A, A, D, E, A>
σ_2 = < B, C, F, E, B>
σ_3 = < B, E, F>

Execution trace not producible on S:

σ_4 = < A, C, E, A> C2 ×
σ_5 = < B, D, C> C2 ×
σ_6 = < A, D, B, F, E> C1 ×

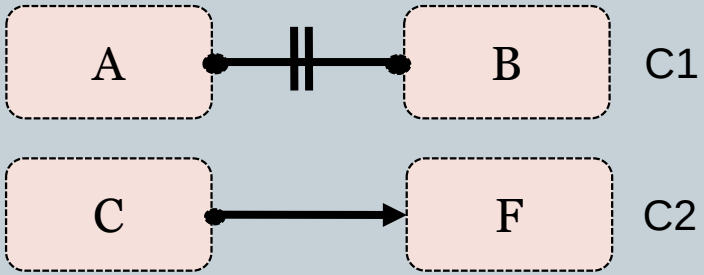
Executing Declarative Processes

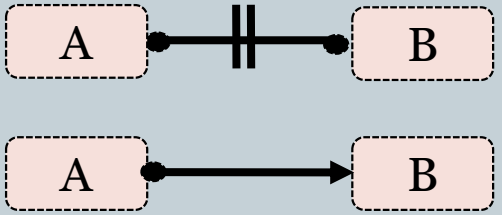
Declarative Process Model S

Activities A

A B C D E F

Constraints C



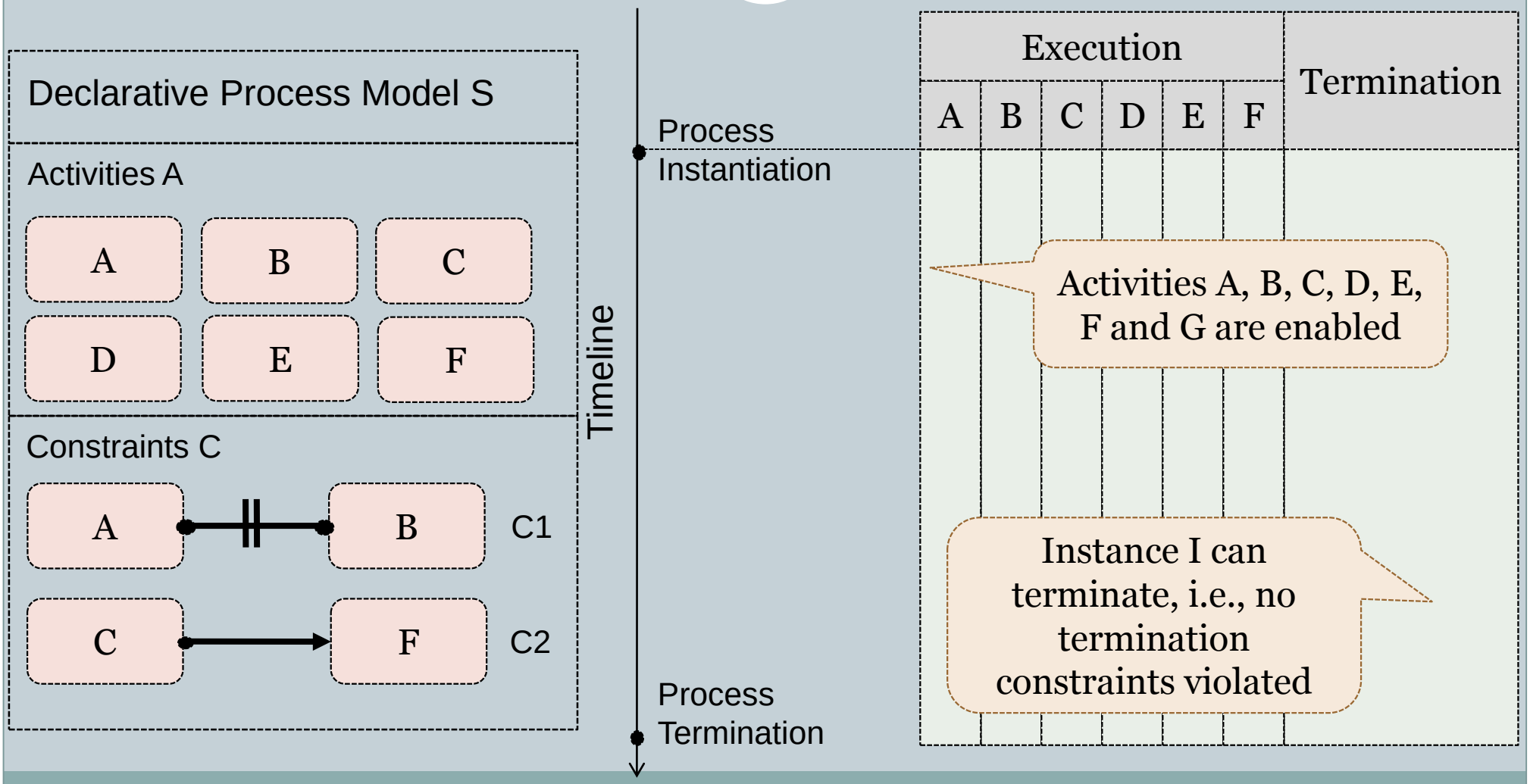


NOT CO-EXISTENCE
A and B are mutually exclusive

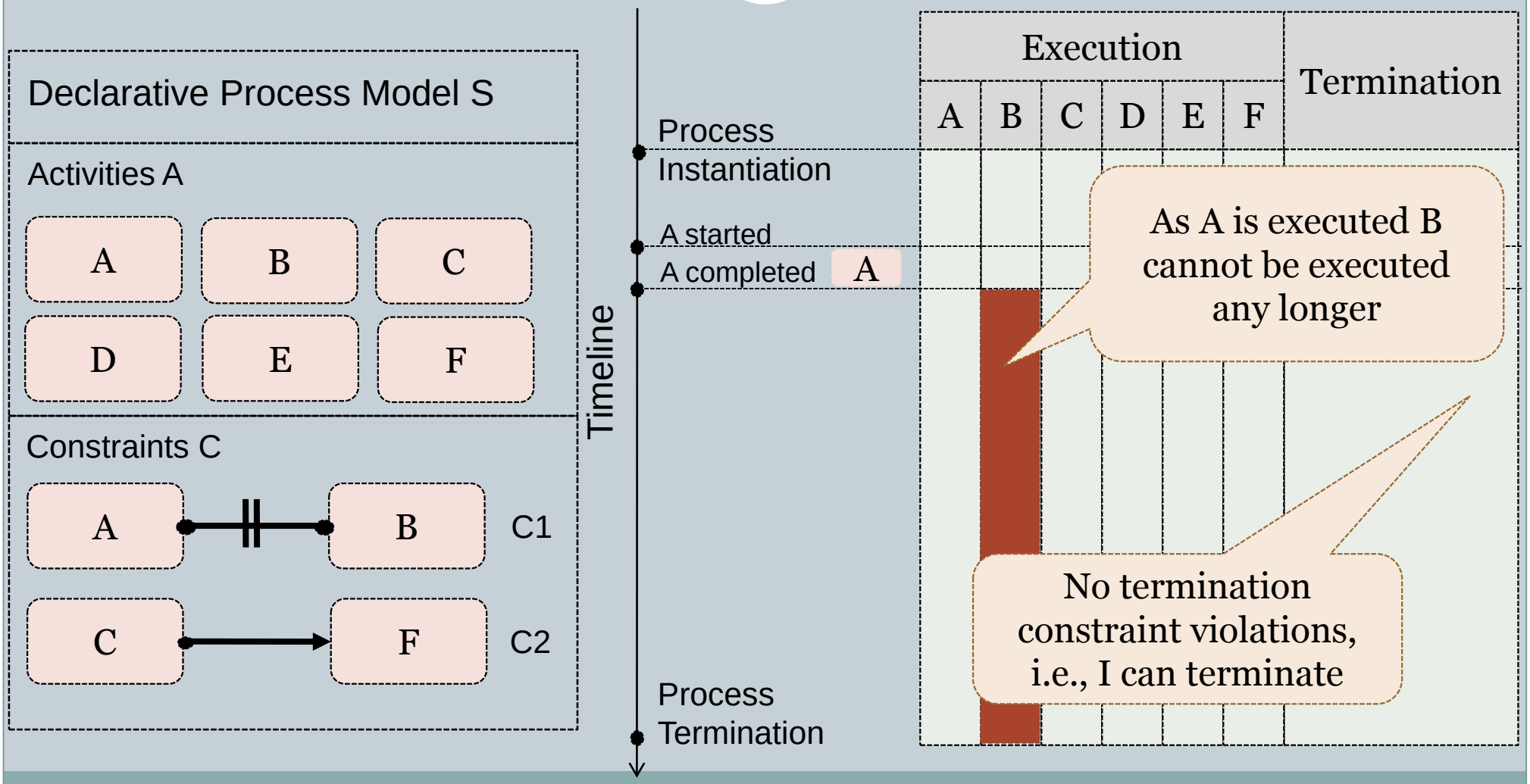
RESPONSE
If A is executed, B needs to executed afterwards

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

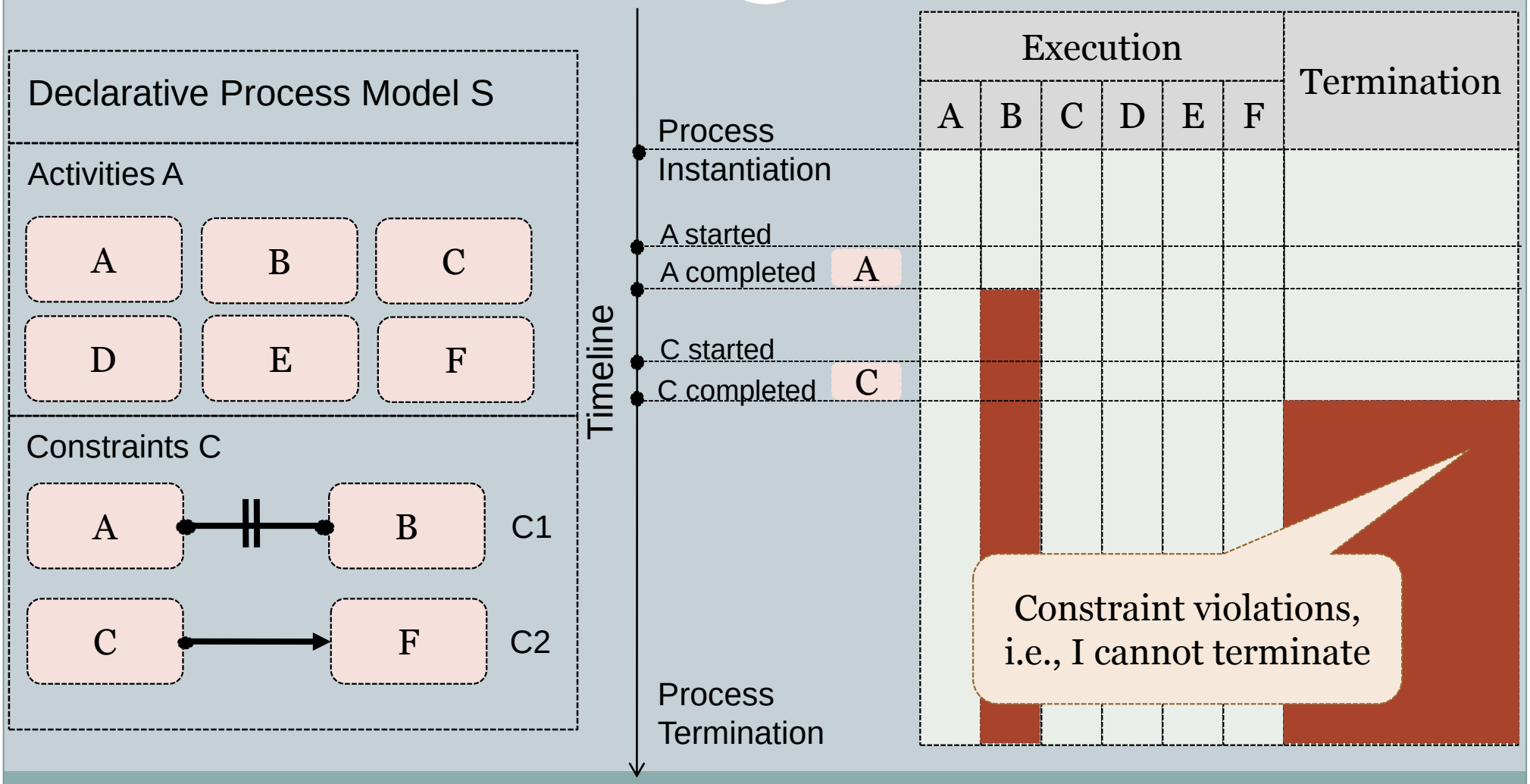
Executing Declarative Processes



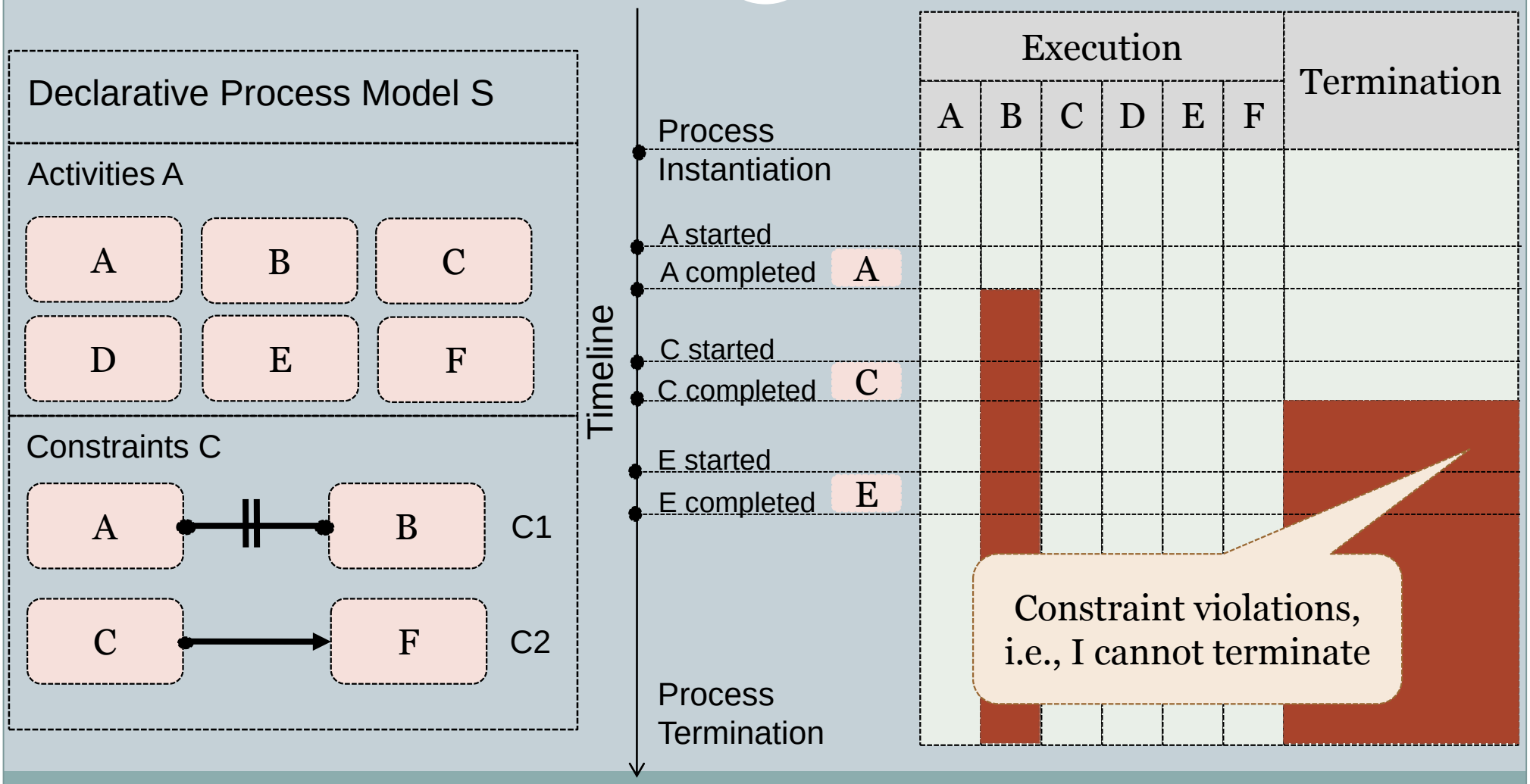
Executing Declarative Processes



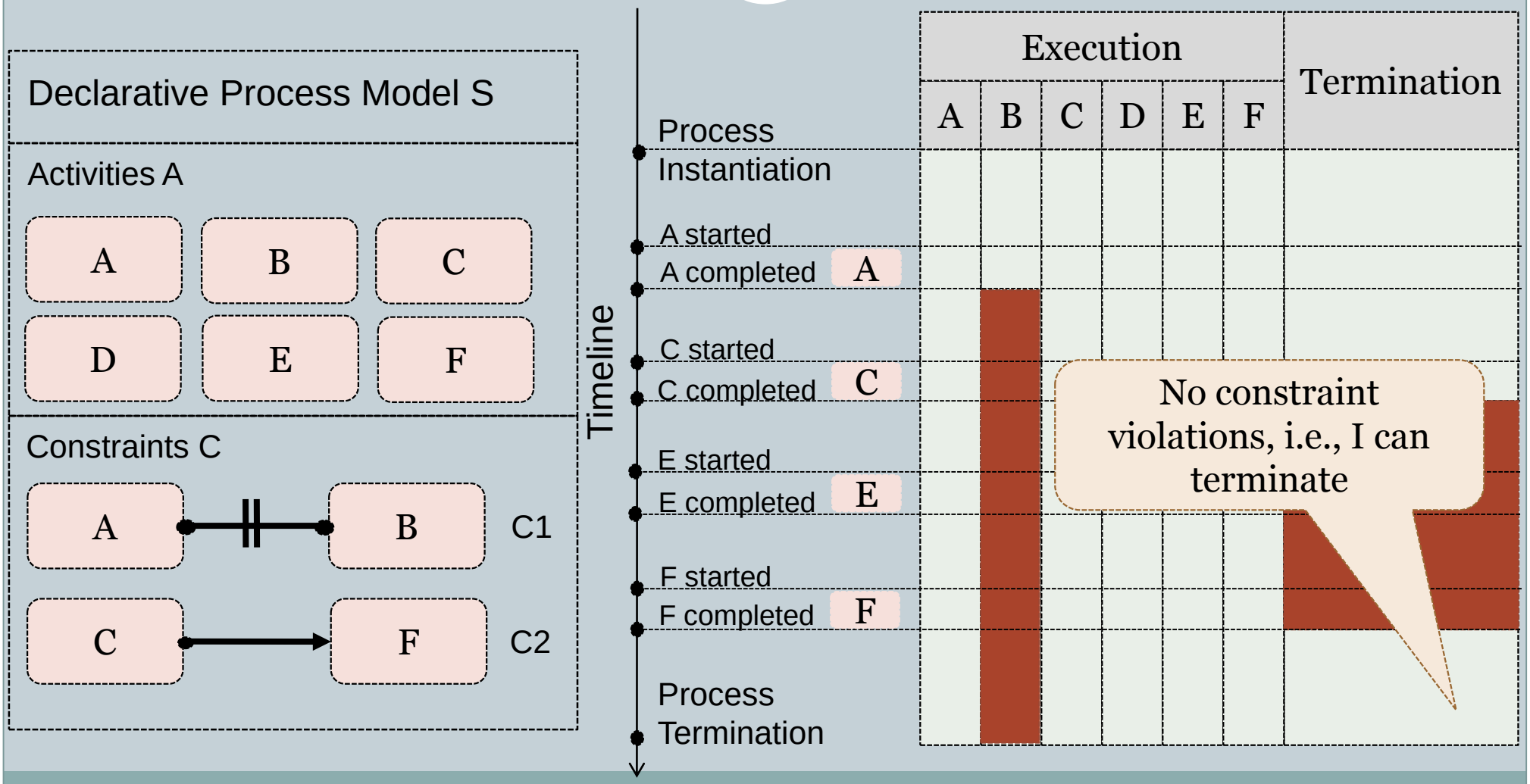
Executing Declarative Processes



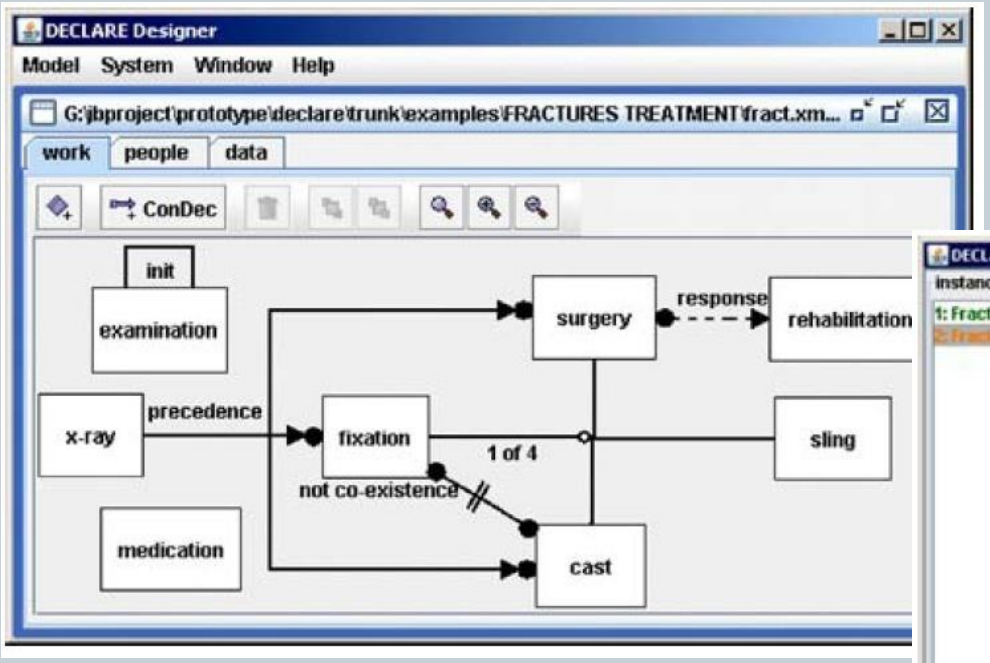
Executing Declarative Processes



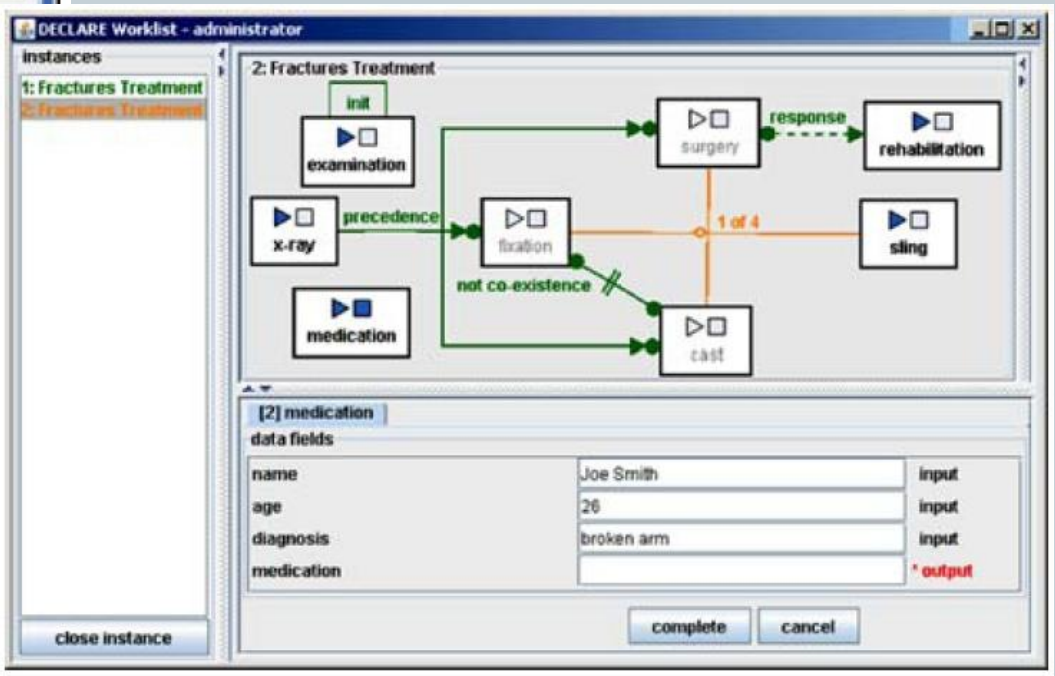
Executing Declarative Processes



The Declare System

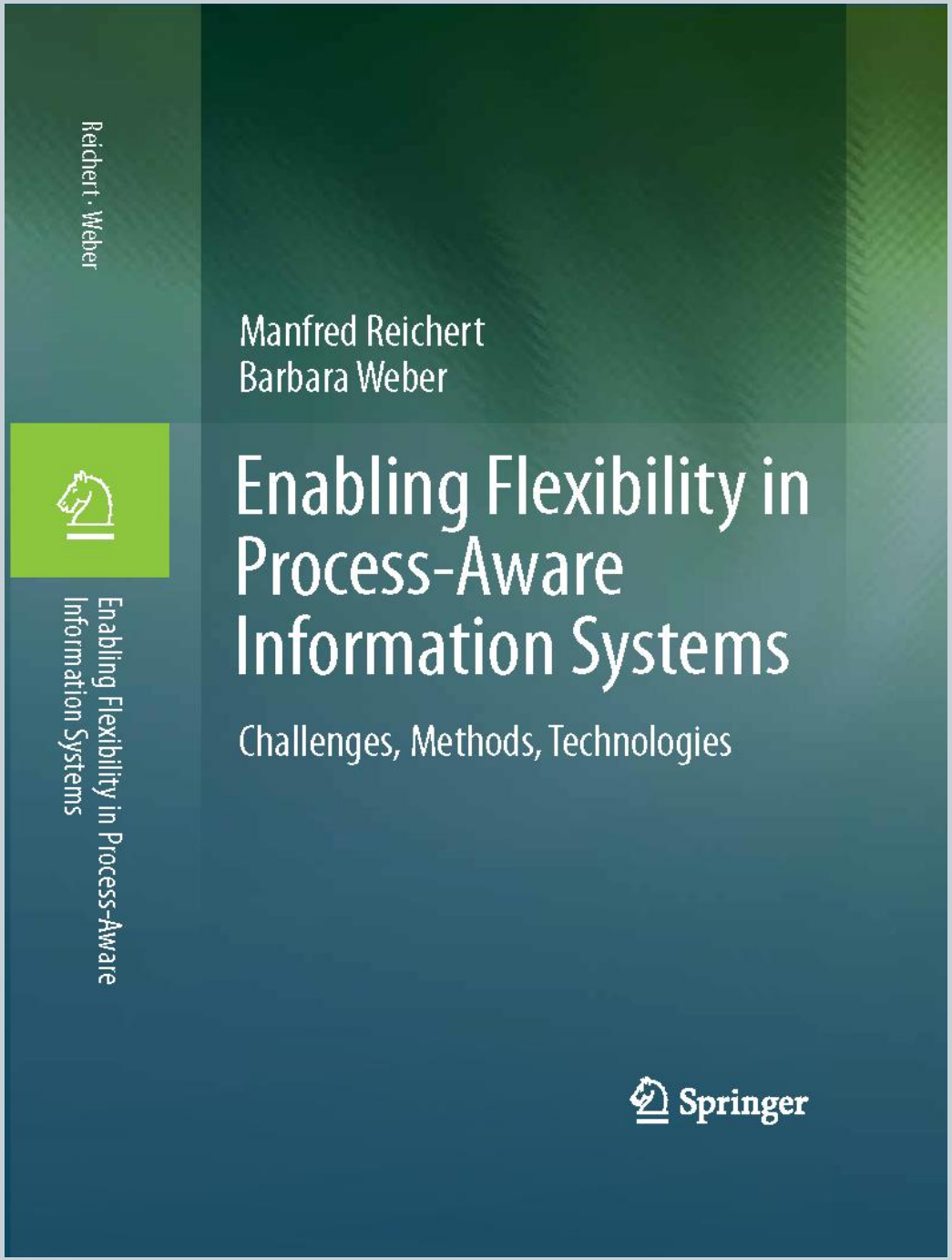


Composing Declarative Processes with Declare

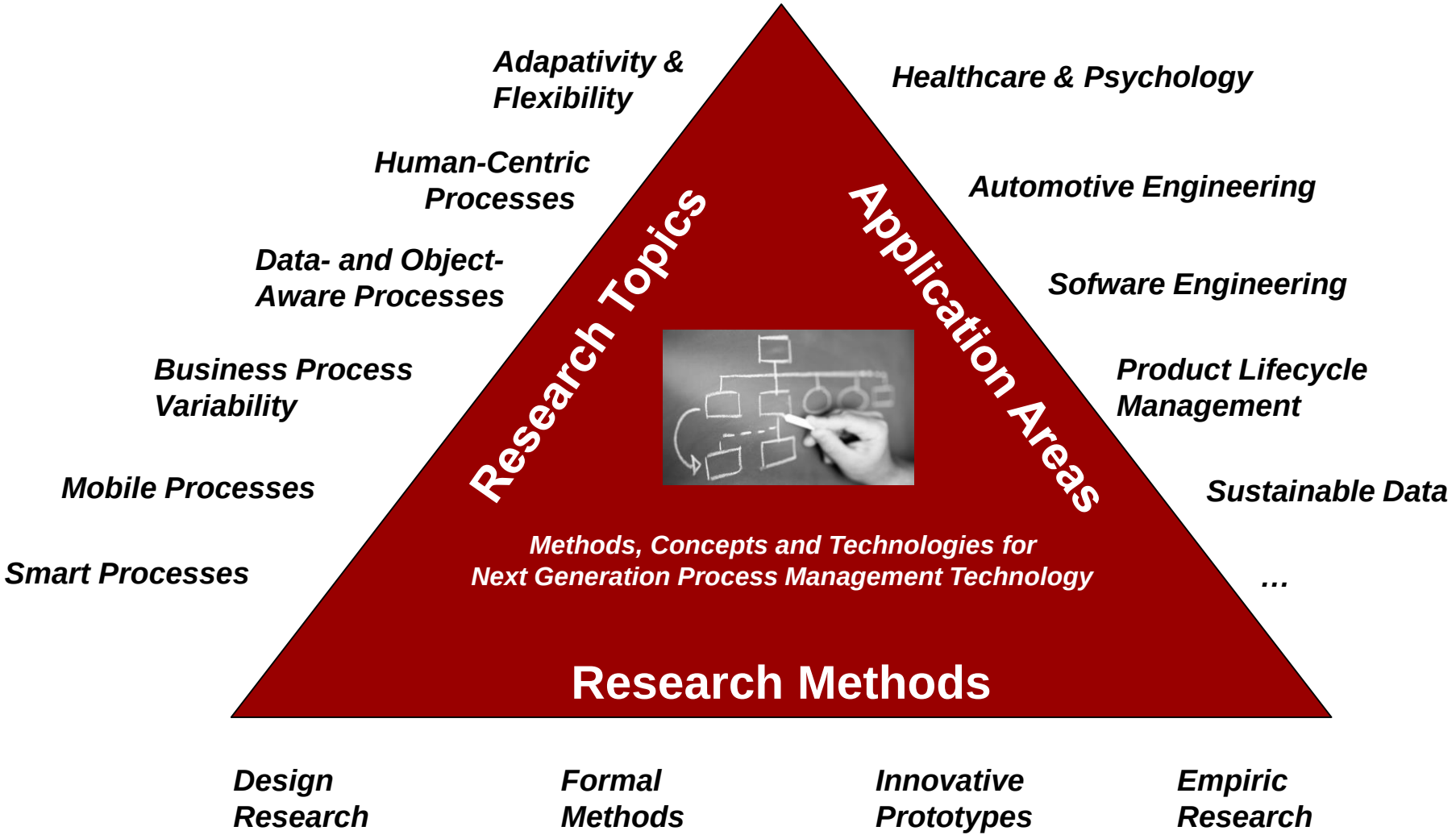


Executing Declarative Processes with Declare

van der Aalst, Pesic and Schonenberg 2009 [APS09]


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

Research in my Group



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.

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)

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.

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

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.

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)

References (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.

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

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)

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