ER12-Tutorial: Enabling Flexibility in Process-aware Information Systems
Challenges, Methods, Technologies

BARBARA WEBER
UNIVERSITY OF INNSBRUCK

MANFRED REICHERT
ULM UNIVERSITY

ER 2012, FLORENCE
WWW.PROCESS-FLEXIBILITY.COM

© B. Weber and M. Reichert, 2012
Content

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

Welcome customer

Offer Clothes

Bill Clothes

Hand over clothes

Mendling 2006
Business Process Lifecycle

- **Design & Analysis**
  - Design: Business Process Identification and Modeling
  - Analysis: Validation, Simulation, Verification

- **Enactment**
  - Enactment: Operation, Monitoring, Maintenance

- **Configuration**
  - Configuration: System Selection, Implementation, Test and Deployment

- **Evaluation**
  - Evaluation: Process Mining, Business Activity Monitoring


© B. Weber and M. Reichert, 2012
BPM Value Proposition

- Value to shareholders and competitiveness
- Process modeling
- Business insight
- Compliance & consistency
- IT agility
- Efficiency
- Knowledge
- Transformation

Stakeholders:
- Workers, supervisors, and managers
- Customers and partners
- CIO
- CFO
- CXO
- CEO

BPM adoption maturity

Forester 2007 BPM Market Overview

© B. Weber and M. Reichert, 2012
Process-aware Information System

Process Composer
Create Process Schema
Modify Process Schema
Check Process Schema

Process Repository
Application Components
Process Models

Process Engineer

Users

Process Execution Engine

Process-aware Information System (PAIS)

<table>
<thead>
<tr>
<th>Feature</th>
<th>Processes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Late Modeling</td>
<td>Instance 1, 2</td>
</tr>
<tr>
<td>Admin. API</td>
<td>Instance 3, 4</td>
</tr>
<tr>
<td>Validation</td>
<td>Instance 5, 6</td>
</tr>
<tr>
<td>Authorization</td>
<td>Instance 7, 8</td>
</tr>
<tr>
<td>Exceptions</td>
<td>Instance 9, 10</td>
</tr>
<tr>
<td>Audit Trail</td>
<td>Instance 11, 12</td>
</tr>
<tr>
<td>Time Mgmt</td>
<td>Instance 13, 14</td>
</tr>
<tr>
<td>Dyn. Change API</td>
<td></td>
</tr>
</tbody>
</table>

© B. Weber and M. Reichert, 2012
Business Process – System Perspective

Process Schema S

Process Instance I1

Execution Trace:
\( \sigma_1 = < \text{"Patient Admission"}, \text{"Anamnesis & Clinical Examination"}, \text{"X-ray"}> \)

Activity States: ▲ Enabled □ Completed △ Skipped

Process Instance I2

Execution Trace:
\( \sigma_2 = < \text{"Patient Admission"}, \text{"Anamnesis & Clinical Examination"}, \text{"Non Operative Therapy"}> \)
User Perspective

Process Instance I5

- Patient Admission
- Anamnesis & Clinical Examination
- X-ray
- MRT
- Sonography

Non Operative Therapy
- Non Operative Therapy 1
- Initial Treatment & Operation Planning
- Operative Treatment

Operative Treatment
- Discharge & Documentation

Withdrawn

© B. Weber and M. Reichert, 2012
Let’s do the MRT

User Perspective

Process Instance I5

Joe

- Patient Admission ✓
- Anamnesis & Clinical Examination ✓
- MRT
- Sonography ✓
- Initial Treatment & Operation Planning ✓
- Operative Treatment
- Discharge & Documentation

Peter

- Non Operative Therapy
- Non Operative Therapy 1

Offered ✓
Allocated ✓
Started ✓
Completed ✓
Withdrawn

© B. Weber and M. Reichert, 2012
User Perspective

Process Instance I5

Joe

Patient Admission ✓ Anamnesis & Clinical Examination ✓

X-ray ✓ MRT ✓ Sonography

Non Operative Therapy

Non Operative Therapy 1

Initial Treatment & Operation Planning

Operative Treatment

Discharge & Documentation

Withdrawn

Offered ✓ Allocated ✓ Started ✓ Completed

© B. Weber and M. Reichert, 2012
User Perspective

Process Instance I5

Joe

Patient Admission ✓
Anamnesis & Clinical Examination ✓
X-ray ✓
MRT ✓
Sonography ✓
Non Operative Therapy

Peter

Non Operative Therapy 1
Initial Treatment & Operation Planning
Operative Treatment

Discharge & Documentation

Offered ✓
Allocated ✓
Started ✓
Completed ✓
Withdrawn

© B. Weber and M. Reichert, 2012
Business Processes and Workflows

Flexibility Issues

BARBARA WEBER
UNIVERSITY OF INNSBRUCK

MANFRED REICHERT
ULM UNIVERSITY

ER 2012, FLORENCE

© B. Weber and M. Reichert, 2012
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 is typical for many domains and requires that processes are handled differently depending on the particular context.

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

Example: Vehicle Repair
Knowledge-intensive processes cannot be fully pre-specified, but require loose specifications.

Drivers
- Unpredictability
- Non-Repeatability
- Emergence

Example: Treatment Processes in a Hospital

© B. Weber and M. Reichert, 2012
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
• Ability of the implemented process to change when the business process evolves

• Drivers

- Changing Business Context
- Changing Technological Context
- Changing Legal Context
- Organizational Learning

Real-world Process

PAIS

represented in

Internal

External

provide feedback to

Design Errors
Technical Problems
Poor Internal Quality

© B. Weber and M. Reichert, 2012
Evolution

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

- **Duration**
  - Temporary
  - Permanent
Evolution

- **Swiftness**
  - Deferred
    - Ongoing instances are not affected
  - Immediate
    - Ongoing instances are affected

- **Visibility**
  - Observable Behavior
  - Internal Structure

Example: Tender Preparation
Example: Inconsistent Naming of Process Models

© B. Weber and M. Reichert, 2012
Flexibility Issues along the Process Lifecycle

Traditional Process Lifecycle Support

1. Create Instances
2. Process Monitoring
3. Process Execution
4. Execution Log
5. Create Instances

Need for Process Evolution

Need for Process Adaptation (Support for Planned and Unplanned Exceptions / Special Cases)

Need for Variability Support

Need for Looseness of Process Specifications

© B. Weber and M. Reichert, 2012
### Flexibility Needs and Technological Requirements

<table>
<thead>
<tr>
<th>Flexibility Need</th>
<th>Dimension</th>
<th>Technological Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variability</td>
<td></td>
<td>Configuration</td>
</tr>
<tr>
<td>Looseness</td>
<td></td>
<td>Loosely-specified processes</td>
</tr>
<tr>
<td>Adaptation</td>
<td>Planned</td>
<td>Exception Handling</td>
</tr>
<tr>
<td></td>
<td>Unplanned</td>
<td>Ad-hoc Changes</td>
</tr>
<tr>
<td>Evolution</td>
<td>Deferred</td>
<td>Versioning</td>
</tr>
<tr>
<td></td>
<td>Evolution</td>
<td>Process Instance Migration</td>
</tr>
<tr>
<td></td>
<td>Immediate</td>
<td>Refactoring</td>
</tr>
<tr>
<td></td>
<td>Evolution</td>
<td>Monitoring, Analysis and Mining</td>
</tr>
<tr>
<td></td>
<td>Poor Internal</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Quality</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Organizational</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Learning</td>
<td></td>
</tr>
</tbody>
</table>
Business Processes and Workflows
Pre-specified Process Models and Flexibility-by-Design

BARBARA WEBER
UNIVERSITY OF INNSBRUCK

MANFRED REICHERT
ULM UNIVERSITY

ER 2012, FLORENCE

© B. Weber and M. Reichert, 2012
Basic Control Flow Concepts & Patterns

24

Transition Conditions

Atomic Activity  Sequence Flow  XOR Gateway  Default Path  AND Gateway

© B. Weber and M. Reichert, 2012
Basic Data Flow Concepts & Patterns

Data Edge – Write Access

Data Object

Data Edge – Read Access

Transition Condition references SparePartsList

End Message with Data Object Invoice
Examples of Control Flow Patterns (1)

A) Sequence Pattern

B) Parallel Split (AND-Split)

C) Synchronization (AND-Join)

D) Exclusive Choice (XOR-Split)

© B. Weber and M. Reichert, 2012
Expressiveness and Flexibility-by-Design

A) Interleaved routing pattern (extended variant)

Preheat Oven → Heat Oil → Cook → Transfer to Bowl along with Spinach, Toss and Cool → Arrange Mushroom Caps → Add Feta to Spinach Mixture

Add Mushroom Stems
Add Onions
Add Salt
Add Pepper

Add Salt to Spinach Mixture
Add Pepper to Spinach Mixture

Divide Filling Between Mushrooms, Bake
(Missing) Expressiveness and Flexibility by Design
Business Processes and Workflows
Configurable Process Models

BARBARA WEBER
UNIVERSITY OF INNSBRUCK

MANFRED REICHERT
ULM UNIVERSITY

ER 2012, FLORENCE
Motivation – Change Management Process

<table>
<thead>
<tr>
<th>Standard</th>
<th>Process variant I: Quality issues affected</th>
<th>Process variant II: Low risk/costs; long to realize</th>
<th>Process variant III: Low risk/ costs; fast to realize; affects quality issues</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) Change request</td>
<td>1) Change request</td>
<td>1) Change request</td>
<td>1) Change request</td>
</tr>
<tr>
<td>Applier</td>
<td>Applier</td>
<td>Applier</td>
<td>Applier</td>
</tr>
<tr>
<td>2) Request for Statements (Stm.)</td>
<td>2) Request for Statements (Stm.)</td>
<td>2) Request for Statements (Stm.)</td>
<td>2) Request for Statements (Stm.)</td>
</tr>
<tr>
<td>Responsible</td>
<td>Responsible</td>
<td>Responsible</td>
<td>Responsible</td>
</tr>
<tr>
<td>3a) Stm.</td>
<td>3b) Stm.</td>
<td>3c) Stm.</td>
<td>3d) Stm.</td>
</tr>
<tr>
<td>Development (Dev.)</td>
<td>Pilot</td>
<td>Production-Planning (PP)</td>
<td>Quality Department (QDept.)</td>
</tr>
<tr>
<td>4) Integration of Stm.</td>
<td>4) Integration of Stm.</td>
<td>4) Integration of Stm.</td>
<td>4) Integration of Stm.</td>
</tr>
<tr>
<td>Project Leader</td>
<td>Project Leader</td>
<td>Project Leader</td>
<td>Project Leader</td>
</tr>
<tr>
<td>5) Permission</td>
<td>5) Permission</td>
<td>5) Permission</td>
<td>5) Permission</td>
</tr>
<tr>
<td>Decision Board</td>
<td>Decision Board</td>
<td>Decision Board</td>
<td>Decision Board</td>
</tr>
<tr>
<td>6) Realization</td>
<td>6) Realization</td>
<td>6) Realization</td>
<td>6) Realization</td>
</tr>
<tr>
<td>Dev.</td>
<td>Dev.</td>
<td>Dev.</td>
<td>Dev.</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>7) Completion</td>
<td>7) Completion</td>
<td>7) Completion</td>
<td>7) Completion</td>
</tr>
<tr>
<td>Responsible</td>
<td>Responsible</td>
<td>Responsible</td>
<td>Responsible</td>
</tr>
</tbody>
</table>

Quality Department (QDept.)
Motivation – Vehicle Repair Process

Conclusion: Many processes with different variants, depending on the process context.
Conclusion: Both approaches can be supported by commercial BPM tools, but do not enable transparent and explicit management of process variants.
Motivation – Handling Medical Examinations

Variety of related variants

- Same business objective
- Commonalities
- Differences due to varying application context
Two Main Approaches for Capturing Process Variability

- **Behaviour-based Approaches**

- **Structural Approaches**
Behavior-based Approaches

- Main idea: Merging all possible behavior in on reference model with **configurable nodes**
  - Extension of an existing process modeling language by adding configurable elements (e.g., activities, control connectors)
  - Examples: C-EPC, C-YAWL

- Configurable nodes represent **variation points** associated with configuration alternatives

- Possible combinations of configuration alternatives can be restricted through constraints

© B. Weber and M. Reichert, 2012
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
Configurable Model: Handling Medical Examinations
Two Main Approaches for Capturing Process Variability

- Behaviour-based Approaches
- Structural Approaches
Deriving Variants through Structural Changes of a Base Process Model
Representing a Process Family

• Through a configurable base process model
  ○ Policy 1: Standard Process
  ○ Policy 2: Most frequently used process
  ○ Policy 3: Superset of all process variants
  ○ Policy 4: Intersection of all process variants

• and a related set of pre-specified changes
  ○ Adjustment points
  ○ Change options (i.e., a grouping of change operations)
### Examples of Change Operations

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

© B. Weber and M. Reichert, 2012
Constraining Allowed Combinations of Change Options

- Implications
- Mutual exclusion
- Hierarchy
Context Model

- Context-specific selection of change options
  - Context variables

<table>
<thead>
<tr>
<th>Context Variable</th>
<th>Value Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examination-Type</td>
<td>Standard medical exam., Emergency medical exam</td>
</tr>
<tr>
<td>Scheduling-Type</td>
<td>Examination with appointment,</td>
</tr>
<tr>
<td></td>
<td>Examination with simple registration,</td>
</tr>
<tr>
<td></td>
<td>Emergency registration</td>
</tr>
<tr>
<td>Preparing-Patient</td>
<td>Yes, No</td>
</tr>
<tr>
<td>Informing-Patient</td>
<td>Yes, No</td>
</tr>
<tr>
<td>Transporting-Patient</td>
<td>Yes, No</td>
</tr>
</tbody>
</table>
Bringing all together ...

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

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

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

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

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

(a) Linking Domain Facts and Configurable Activities

(b) Linking Domain Facts and Configurable Connectors
Questionnaire-driven Process Configuration

Such questionnaire models are used for configuring a reference process model, e.g., by linking domain facts to configurable activities or connectors!
Business Processes and Workflows
Exception and Compensation Handling

BARBARA WEBER
UNIVERSITY OF INNSBRUCK

MANFRED REICHERT
ULM UNIVERSITY

ER 2012, FLORENCE

© B. Weber and M. Reichert, 2012
Process Adaptations

- Planned
- Exception Handling
- Unplanned
- Ad-hoc Changes
Exception Handling in PAIS

We will skip respective techniques for handling planned exceptions in this tutorial and refer to our textbook instead!
Business Processes and Workflows
Handling Unforeseen Exceptions

BARBARA WEBER
UNIVERSITY OF INNSBRUCK

MANFRED REICHERT
ULM UNIVERSITY

ER 2012, FLORENCE

© B. Weber and M. Reichert, 2012
Process Adaptations

- Planned
- Exception Handling

- Unplanned
- Ad-hoc Changes

© B. Weber and M. Reichert, 2012
User View on an Ad-hoc Process Change

Exception – We need an additional lab test!
Behavioral Changes Require Structural Process Model Adaptations
Dynamic Change Bug

Diagram of Dynamic Change Bug:

- Diagram a) shows a simple process flow with states A, B, C, and D.
- Diagram b) includes dynamic changes with a highlighted state C and a note on dynamic change bug.
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

Data flow error caused by missing data

No Proper Completion ensured. End node can be reached while B is still enabled
Ad-hoc Changes of a Process Instance Must Not Affect any Other Process Instances

**Process Type Level**

Process Schema S

- Patient Admission
- Anamnesis & Clinical Examination
- X-ray
- Non Operative Therapy
- MRT
- Sonography
- Initial Treatment & Operation Planning
- Operative Treatment
- Discharge & Documentation

**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:
\[ \sigma_2 = \langle \text{Patient Admission} \rangle \]

© B. Weber and M. Reichert, 2012
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

<table>
<thead>
<tr>
<th>Adding / Deleting</th>
<th>AP1: Insert Process Fragment</th>
</tr>
</thead>
<tbody>
<tr>
<td>AP2: Delete Process Fragment</td>
<td></td>
</tr>
<tr>
<td>Moving / Replacing Process Fragments</td>
<td>AP3: Move Process Fragment</td>
</tr>
<tr>
<td>AP4: Replace Process Fragment</td>
<td></td>
</tr>
<tr>
<td>AP5: Swap Process Fragment</td>
<td></td>
</tr>
<tr>
<td>AP14: Copy Process Fragment</td>
<td></td>
</tr>
<tr>
<td>Adding / Removing Process Levels</td>
<td>AP6: Extract Sub Process</td>
</tr>
<tr>
<td>AP7: Inline Sub Process</td>
<td></td>
</tr>
<tr>
<td>Adapting Control Dependencies</td>
<td>AP8: Embed Process Fragment in Loop</td>
</tr>
<tr>
<td>AP9: Parallelize Process Fragments</td>
<td></td>
</tr>
<tr>
<td>AP10: Embed Process Fragment in Conditional Branch</td>
<td></td>
</tr>
<tr>
<td>AP11: Add Control Dependency</td>
<td></td>
</tr>
<tr>
<td>AP12: Remove Control Dependency</td>
<td></td>
</tr>
<tr>
<td>Change Transition Conditions</td>
<td>AP13: Update Condition</td>
</tr>
</tbody>
</table>
Adaptation Patterns versus Change Primitives

Process Model S

MOVE ACTIVITY C TO
POSITION BETWEEN A AND B

Process Model S'

Snapshot Difference (Change Primitives)

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

© B. Weber and M. Reichert, 2012
Bild tauschen
Barbara Weber; 02.04.2011
Correctness of Process Instance Changes

Ensuring Dynamic Correctness

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

Need for general correctness criterion

⇒ State Compliance

[A C D E F]

[ReDa98, RRW08a, RRD04a, RRD04b]

© B. Weber and M. Reichert, 2012
Correctness of Process Instance Changes

Ensuring Dynamic Correctness

Schema S:

\[\left\langle A \right\rangle, \left\langle B \right\rangle, \left\langle D \right\rangle \Rightarrow \text{Trace reproducible on new schema?}\]

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

- **Patient Admission**
- **Anamnesis & Clinical Examination**
- **X-ray**
- **Sonography**
- **MRT**
- **Non Operative Therapy**
- **Non Operative Therapy 1**
- **Operative Treatment**
- **Discharge & Documentation**

**Process Instance Level**

Process Instance I3

- **Initial Treatment & Operation Planning**

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)

[ReDa98]
User Assistance & Change Reuse (1)

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

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

[RWR+05, WRW+09, WRR+05, WRW06, WWB04]
The treatment of cruciate ruptures routinely includes a magnetic resonance tomography (MRT), an X-ray and a sonography. However, for a particular patient the MRT may have to be skipped as the respective patient has a cardiac pacemaker.

\[ \text{sol}_c = \text{<Delete}(S,\text{MRT}) \]

\[ \text{qaSet}_c = \{(\text{Does the patient have a cardiac pacemaker?}, \text{patient.problemList.hasPacemaker} = \text{"Yes"})\} \]

\[ \text{freq}_c = 1 \]
Business Processes and Workflows
Process Evolution

BARBARA WEBER
UNIVERSITY OF INNSBRUCK

MANFRED REICHERT
ULM UNIVERSITY

ER 2012, FLORENCE

© B. Weber and M. Reichert, 2012
Drivers

External

Changing Business Context
Changing Technological Context
Changing Legal Context
Organizational Learning

represent in

Real-world Process

provide feedback to

PAIS

Internal

Design Errors
Technical Problems
Poor Internal Quality

© B. Weber and M. Reichert, 2012
Schema Evolution

Schema S

Enter Customer Request → Check Feasibility → Create Offer → Submit Tender

Standard customer:
- Create Offer

Gold customer:
- Create Special Offer
  - Request Approval
  - special offer not approved

Schema S’

Enter Customer Request → Check Feasibility → Create Offer → Submit Tender

by applying change $\Delta S$ with:

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

**Process Instance I1**

**Process Instance I2**

**Process Schema S’**

**Process Instance I1**

**Process Instance I2**

Change is propagated to all running process instances

**Schema Evolution**

Inconsistent state

© B. Weber and M. Reichert, 2012
Scenario 2 - Version Control

- Co-existence of instances of different schema versions

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

Process Schema S

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

Old instances remain with schema S

- Instances created from S (before schema evolution)
  
  Process Instance I1
  
  Process Instance I2

Instances created from S' (after schema evolution)

- Process Instance I4

- Process Instance I5

© B. Weber and M. Reichert, 2012
Scenario 3 – Instance Migration

Compliant instances are migrated to the new schema

Type change results into a new version of schema S

Migration of compliant process instances to S'

Process Instance I1

Process Instance I2

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

Process Instance I2 not compliant with S'

[RRD04a]

© B. Weber and M. Reichert, 2012
Process Model Refactoring

(1) Identify refactoring opportunities

(2) Determine which refactoring should be applied

(3) Ensure that the applied refactoring preserves model behavior

(4) Apply the refactoring

(5) Assess the effect of the refactoring on the quality characteristics of the process model repository
Catalogue of Process Model Smells

PMS1: Non Intension Revealing Naming of Activity / Process Model
PMS2: Contrived Flexibility
PMS4: Large Process Models
PMS3: Redundant Process Fragment
PMS5: Lazy Process Model
PMS6: Unused Branches
PMS7: Frequently Occurring Instance Change
PMS8: Frequently Occurring Variant Change

[WeRe08, WRR+xx]
Process Model Smells: Example

PMS3: Redundant Process Fragment
Process Model Smells: Example

PMS1: Non Intension Revealing Naming of Activity
Process Model Smells: Example

PMS4: Large Process Model
Catalogue of Process Model Refactorings

RF1: Rename Activity
RF2: Rename Process Schema
RF3: Substitute Process Fragment
RF4: Extract Process Fragment
RF5: Replace Process Fragment by Reference
RF6: Inline Process Fragment
RF7: Re-label Collection
RF8: Remove Redundancies
RF9: Generalize Variant Change
RF10: Remove Unused Branches
RF11: Pull up Instance Change

© B. Weber and M. Reichert, 2012
Process Model Smells: Example

PMS3: Redundant Process Fragment

RF4: Extract Process Fragment

RF5: Replace Process Fragment by Reference
Process Model Smells: Example

PMS1: Non Intension Revealing Naming of Activity

RF1: Rename Activity
Process Model Smells: Example

PMS4: Large Process Model

RF4: Extract Process Fragment
Process Model After Refactoring
Integrated Lifecycle Support for Adaptive and Dynamic Processes (1)

Traditional Process Lifecycle Support

1. Evolve Process Schema
2. Create Instances
3. Process Execution
4. Execution Log
5. Process Monitoring

Schema S:

A → B → C → D → E

Instance I:

A → C → E

© B. Weber and M. Reichert, 2012
Lifecycle Support in adaptive PAISs

1. Create Process Schema
2. Create Instances
3. Process Execution
4. Change Log
5. Execution Log
6. Change Propagation
7. Process Monitoring

Exception: Delete \((I_1, E)\)

Process engineer / Process administrator

Process participant

© B. Weber and M. Reichert, 2012
Revised lifecycle for dynamic processes – The ProCycle Approach

 instances I1

\[ A \rightarrow D \rightarrow B \rightarrow x \rightarrow x \rightarrow E \]

Schema S:

\[ A \rightarrow x \rightarrow C \rightarrow x \rightarrow E \]

Schema S':

\[ B \rightarrow \]

Create Instance

Process engineer / Process administrator

Evolve Process Schema

Create Process Schema

Process

Execution

Log

Change Log

Instance-specific Change

Process participant

Process Execution

Migrate Case Base

Memorization and Change Reuse

Exception: Delete (I1, E)

Derive Process Type Change

Create Instances

Change Propagation

Integrated Lifecycle Support for Adaptive and Dynamic Processes (3)

© B. Weber and M. Reichert, 2012
Business Processes and Workflows
Loosely Specified Processes

BARBARA WEBER
UNIVERSITY OF INNSBRUCK

MANFRED REICHERT
ULM UNIVERSITY

ER 2012, FLORENCE

© B. Weber and M. Reichert, 2012
Loosely specified Processes

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

<table>
<thead>
<tr>
<th>Pattern</th>
<th>Degree of Freedom</th>
<th>Planning Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traditional Workflow</td>
<td>None</td>
<td>Plan-driven</td>
</tr>
<tr>
<td>Late Selection</td>
<td>Selection</td>
<td>Plan-driven</td>
</tr>
<tr>
<td>Late Modeling and Composition</td>
<td>Modeling / Composition</td>
<td>Plan-driven</td>
</tr>
<tr>
<td>Iterative Refinement</td>
<td>Modeling / Composition</td>
<td>Iterative</td>
</tr>
<tr>
<td>Ad-hoc Composition</td>
<td>Modeling / Composition</td>
<td>Ad-hoc</td>
</tr>
</tbody>
</table>
Late Selection Pattern
Late Selection – The Worklets Approach

Casuality Treatment Process Model

- Condition
- Conclusion
- Fever = True
- Treat Fever
- Wound = True
- Treat Wound
- Abdominal Pain = True
- Treat Abdominal Pain
- Fracture = True
- Treat Fracture
- Pregnant = True
- Treat Labor
- Rash = True
- Treat Rash
- HeartRate > 190
- Treat High Heart Rate

Repository of Process Fragments

- Admit Patient
- Perform Triage
- Treat Patient
- Discharge Patient

Treat Feaver
- Test Fever
- Treat Fever

Treat Wound
- ...

Treat Abdominal Pain
- ...

Treat Fracture
- ...

Treat Labor
- ...

Treat Rash
- ...

Treat High Heart Rate
- ...

© B. Weber and M. Reichert, 2012
Late Modeling

Build-time

Process Model S

Run-time

Composed Fragment

Repository of Process Fragments

© B. Weber and M. Reichert, 2012
Late Modeling – Pockets of Flexibility

Customer Relationship Management Model S

Log Customer Request → Solve Problem → Pocket of Flexibility → Inform Customer

Constraints:
- Any of the tests may be done, but only one at a time
- Approve Request is required to execute activity Conduct Site Visit

Examples of Sub-Process Fragments Producible for Placeholder Activity

Order Test 912 → Order Test 67 → Approve Request

Order Test 166 → Conduct Site Visit

Order Test 912 → Order Test 67 → Approve Request → Conduct Site Visit

[SSO01, SSO05]
Ad-hoc Composition - Declare

A) Build-time

Activity Templates

Constraints

If T no X

B) Run-time

start(P) complete(P)

start(X) complete(X)

start(Y) complete(Y)
Iterative Refinement - Alaska

The diagram illustrates the build-time and run-time activities with activity templates and constraints. The build-time activities are labeled P, Q, R, S, T, U, X, and Y. The run-time activities include scheduling and completing tasks.

Constraints:
- If T no X
- After completing P a minimum time lag of 2 hrs is needed before starting U

The schedule and completion times are shown in tables for different time slots (8:00, 9:00, 10:00, 11:00, 12:00, 13:00, 14:00). The checkmarks indicate the scheduled and completed tasks.

[WZP+09]
Business Processes and Workflows

Declarative Processes

BARBARA WEBER
UNIVERSITY OF INNSBRUCK

MANFRED REICHERT
ULM UNIVERSITY

ER 2012, FLORENCE

© B. Weber and M. Reichert, 2012
Declarative Processes

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

© B. Weber and M. Reichert, 2012
Modeling Declarative Processes

Declarative Process Model S

<table>
<thead>
<tr>
<th>Activities A</th>
<th>Constraints C</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>B</td>
<td>B</td>
</tr>
<tr>
<td>C</td>
<td>C1</td>
</tr>
<tr>
<td>D</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>C2</td>
</tr>
</tbody>
</table>

Legend

- NOT CO-EXISTENCE
  A and B are mutually exclusive

- RESPONSE
  If A is executed, B needs to be executed afterwards

Execution trace producible on S:

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

Execution trace not producible on S:

- $\sigma_4 = <A, C, E, A>$
- $\sigma_5 = <B, D, C>$
- $\sigma_6 = <A, D, B, F, E>$

© B. Weber and M. Reichert, 2012
Modeling Declarative Processes

- **5 Major Categories**
  - Selection Constraints
  - Relation Constraints
  - Branching Constraints
  - Negation Constraints
  - Choice Constraints
### Example: Selection Constraints

<table>
<thead>
<tr>
<th>Condition</th>
<th>Expression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Existence</td>
<td>$\text{existence}(a, n)$</td>
<td>Activity $a$ must occur at least $n$ times in every trace. Example: $\text{existence}(A, 1)$ Supported traces, e.g.: $&lt;A&gt;, &lt;A, A, A&gt;$ Unsupported trace, e.g.: $&lt;&gt;$.</td>
</tr>
<tr>
<td>At Most</td>
<td>$\text{at_most}(a, n)$</td>
<td>Activity $a$ must occur at most $n$ times in every trace. Example: $\text{at_most}(A, 3)$ Supported traces, e.g.: $&lt;&gt;, &lt;A&gt;, &lt;A, A&gt;, &lt;A, A, A&gt;$ Unsupported trace, e.g.: $&lt;A, A, A, A&gt;$.</td>
</tr>
<tr>
<td>Exactly</td>
<td>$\text{exactly}(a, n)$</td>
<td>Activity $a$ must occur exactly $n$ times in every trace. Example: $\text{exactly}(A, 2)$ Supported trace, e.g.: $&lt;A, A&gt;$ Unsupported traces, e.g.: $&lt;A&gt;, &lt;A, A, A&gt;$.</td>
</tr>
<tr>
<td>Init</td>
<td>$\text{init}(a)$</td>
<td>Activity $a$ must be the first executed activity in every trace. Example: $\text{init}(A)$ Supported trace, e.g.: $&lt;A, C, D, B&gt;$ Unsupported trace, e.g.: $&lt;D, C, B, A&gt;$.</td>
</tr>
</tbody>
</table>
## Example: Relation Constraints

<table>
<thead>
<tr>
<th>Relation</th>
<th>Diagram</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>response(a, b)</strong></td>
<td><img src="response.png" alt="Diagram" /></td>
<td>If a is executed, b needs to be executed afterwards (but not necessarily directly after)</td>
</tr>
<tr>
<td>Example: response(A, B)</td>
<td></td>
<td>Supported traces, e.g.: &lt;A, B&gt;, &lt;A, A, A, B&gt;, &lt;B&gt;</td>
</tr>
<tr>
<td>Unsupported trace, e.g.:</td>
<td></td>
<td>&lt;A&gt;</td>
</tr>
<tr>
<td><strong>precedence(a, b)</strong></td>
<td><img src="precedence.png" alt="Diagram" /></td>
<td>Activity b needs to be preceded by activity a</td>
</tr>
<tr>
<td>Example: precedence(A, B)</td>
<td></td>
<td>Supported traces, e.g.: &lt;A, B&gt;, &lt;A, B, B, B&gt;, &lt;A&gt;</td>
</tr>
<tr>
<td>Unsupported trace, e.g.:</td>
<td></td>
<td>&lt;B&gt;</td>
</tr>
<tr>
<td><strong>succession(a, b)</strong></td>
<td><img src="succession.png" alt="Diagram" /></td>
<td>If a is executed, b needs to be executed afterwards (but not necessarily directly after); activity b needs to be preceded by activity a</td>
</tr>
<tr>
<td>Example: succession(A, B)</td>
<td></td>
<td>Supported traces, e.g.: &lt;A, B&gt;, &lt;A, A, A, B&gt;, &lt;A, B, B, B&gt;</td>
</tr>
<tr>
<td>Unsupported traces, e.g.:</td>
<td></td>
<td>&lt;A&gt;, &lt;B&gt;</td>
</tr>
<tr>
<td><strong>respondedExistence(a, b)</strong></td>
<td><img src="respondedExistence.png" alt="Diagram" /></td>
<td>If activity a is executed, activity b needs to be executed either before or after a</td>
</tr>
<tr>
<td>Example: respondedExistence(A, B)</td>
<td></td>
<td>Supported traces, e.g.: &lt;A, B&gt;, &lt;B, A&gt;, &lt;A, B, A&gt;, &lt;B&gt;</td>
</tr>
<tr>
<td>Unsupported trace, e.g.:</td>
<td></td>
<td>&lt;A&gt;</td>
</tr>
</tbody>
</table>
Executing Declarative Processes

Declarative Process Model S

Activities A

A
B
C
D
E
F

Constraints C

A
B
C1

C
F
C2

NOT CO-EXISTENCE
A and B are mutually exclusive

RESPONSE
If A is executed, B needs to be 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

© B. Weber and M. Reichert, 2012
Executing Declarative Processes

### Declarative Process Model S

<table>
<thead>
<tr>
<th>Activities A</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
</tr>
<tr>
<td>B</td>
</tr>
<tr>
<td>C</td>
</tr>
<tr>
<td>D</td>
</tr>
<tr>
<td>E</td>
</tr>
<tr>
<td>F</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Constraints C</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
</tr>
<tr>
<td>B</td>
</tr>
<tr>
<td>C</td>
</tr>
<tr>
<td>C1</td>
</tr>
<tr>
<td>F</td>
</tr>
<tr>
<td>C2</td>
</tr>
</tbody>
</table>

#### Execution

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Termination

- Activities A, B, C, D, E, F and G are enabled
- Instance I can terminate, i.e., no termination constraints violated

© B. Weber and M. Reichert, 2012
Executing Declarative Processes

### Declarative Process Model S

#### Activities A
- A
- B
- C
- D
- E
- F

#### Constraints C
- A
- B
- C
- F

### Timeline

- **Process Instantiation**
- **A started**
- **A completed**

### Execution

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Termination

- **Process Termination**

- **As A is executed B cannot be executed any longer**

- **No termination constraint violations, i.e., I can terminate**

© B. Weber and M. Reichert, 2012
Executing Declarative Processes

Declarative Process Model S

<table>
<thead>
<tr>
<th>Activities A</th>
<th>Constraints C</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>C1</td>
</tr>
<tr>
<td>B</td>
<td>C2</td>
</tr>
<tr>
<td>C</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td></td>
</tr>
</tbody>
</table>

Process Instantiation
- A started
- A completed

Process Termination
- C started
- C completed

Execution
- A
- B
- C
- D
- E
- F

Constraint violations, i.e., I cannot terminate

© B. Weber and M. Reichert, 2012
Executing Declarative Processes

Declarative Process Model S

<table>
<thead>
<tr>
<th>Activities A</th>
<th>Constraints C</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>C1</td>
</tr>
<tr>
<td>B</td>
<td>B</td>
</tr>
<tr>
<td>C</td>
<td>F</td>
</tr>
<tr>
<td>D</td>
<td>C</td>
</tr>
<tr>
<td>E</td>
<td>F</td>
</tr>
</tbody>
</table>

Timeline

Execution

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Termination

Constraint violations, i.e., I cannot terminate

© B. Weber and M. Reichert, 2012
Executing Declarative Processes

Declarative Process Model S

Activities A
- A
- B
- C
- D
- E
- F

Constraints C
- A → B
- C → F

Timeline

Execution

<table>
<thead>
<tr>
<th>Process Instantiation</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>A started</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A completed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C started</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C completed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E started</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E completed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F started</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F completed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Termination

- No constraint violations, i.e., I can terminate

© B. Weber and M. Reichert, 2012
Overriding Constraints

Declarative Process Model S

Activities A
- A
- B
- C
- D
- E
- F

Constraints C
- A
- B
- C
- C1
- C2
- F

Execution
- A
- B
- C
- D
- E
- F

Termination

Soft constraints can be ignored during process execution

Users terminating process instance I are informed about constraint violation

Warning “F not executed after C”
The Declare System

Composing Declarative Processes with Declare

Executing Declarative Processes with Declare

van der Aalst, Pesi and Schonenberg 2009  [APS09]

©  B. Weber and M. Reichert, 2012
The Alaska Simulator

- Is an interactive planning tool providing support for business process composition
- Uses journey as a metaphor for business processes
- Effectively handling uncertainty is fundamental in both domains
- Both the planning of a journey and the execution of a business process is oriented towards a goal
- Selection, ordering and resource constraints are relevant in both settings
- Information on benefits (i.e., business value), cost and duration are essential for decision making

http://alaskasimulator.org

Weber, Zugal, Pinggera and Wild 2009 [WZP+09]
Enabling Flexibility in Process-Aware Information Systems
Challenges, Methods, Technologies


References (Configurable Process Models)


© B. Weber and M. Reichert, 2012


References (Handling Unforeseen Exceptions)


References (Process and Variant Mining)


References (Process Evolution)


[Kli00] Justus Klingemann: Controlled Flexibility in Workflow Management. CAiSE 2000: 126-141
References (Loosely-specified Processes)


References (Declarative Workflows)


© B. Weber and M. Reichert, 2012