Agenda for today

• Announcements for the Final Exam
• Student Feedback
• Brief lecture on Change Management in Workflows
• Revision Questions (if any)

Final Exam

• Sunday 25/1
• 1700 – 1900
• Room A11

• 40/100 Marks
• Closed book/notes
• Similar format to Quiz & Mid Semester

Final Exam

• Scope
  – Module 1 (DB Integration, and Dist. DB Design only)
  – Module 2 (Complete)
  – Module 3 (Complete, and including workflow labs and SAP Seminar)
• See also Mark Distribution in Scope Document

Student Feedback

Purpose of this questionnaire is to provide feedback to the lecturer. You response will not affect your grade in any way!

Workflow Change Management

An Overview

• Scope of the problem
• Dimensions of change
  – Dynamism
  – Adaptability
  – Flexibility

GOOD LUCK
Scenario

Organization with functionally complex processes consisting of several distributed and heterogeneous information systems
  - Good candidate for deployment of workflows?
  - Impact of change on a successful setup?

Process Change

- Business process change can arise due to three main reasons:
  - Process Improvement, which involves performing the same business process with increased efficiency.
  - Process Innovation, which involves performing the business process in a radically different way [Davenport 93].
  - Process Adaptation, which involves adapting the process to unforeseen change.

Reasons for Process Change

<table>
<thead>
<tr>
<th>Change characteristics of workflows:</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Dynamism</td>
</tr>
<tr>
<td>• Adaptability</td>
</tr>
<tr>
<td>• Flexibility</td>
</tr>
</tbody>
</table>

Dimensions of Change

<table>
<thead>
<tr>
<th>Dimensions of Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Dynamism</td>
</tr>
<tr>
<td>- ability of the workflow process to change when the business process evolves</td>
</tr>
<tr>
<td>• Adaptability</td>
</tr>
<tr>
<td>• Flexibility</td>
</tr>
</tbody>
</table>

Dimensions of Change

<table>
<thead>
<tr>
<th>Dimensions of Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Process Reengineering</td>
</tr>
<tr>
<td>- Evolving Workflows</td>
</tr>
<tr>
<td>Dynamic Schema</td>
</tr>
<tr>
<td>Change</td>
</tr>
<tr>
<td>Dealing with Active Instances</td>
</tr>
</tbody>
</table>
**Dimensions of Change**

- **Dynamism**
- **Adaptability**
  - ability of the workflow processes to react to exceptional circumstances
- **Flexibility**

**Dynamic Workflows**

ability of the workflow process to change when the business process evolves

- How to define the model?
- How to verify that the changes do not introduce any structural or semantic errors?
- How to propagate the change to active workflow instances?

**Methodology**

- **Defining**
  - Changes to the Process Model
  - Changes to the Workflow Model
  - Verification of the new model
- **Conforming**
  - Instance Grouping
  - Compliance Issue
- **Enacting**
  - Architectural Implications
  - Temporal Implications
  - Organizational Implications

**Immigration Process**

**Immigration Process Instance**
Defining the Change

- Changes to the Process Model
  - Must be known
  - Must be verified (for semantic correctness)

- Changes to the Workflow Model
  - Add new tasks
  - Delete existing tasks
  - Modify order of execution
  - Modify task properties

- Verification of new Workflow Model

Conforming to the Change

- Instance Grouping
  - Type
  - Compliance
  - Stage
- Compliance graph generation
  - Compensation Tasks
  - Plug Point

Affected Instance

Compliance Graph

Enacting the change

- Handling workflow execution during the transition period
  - Instances may follow old model
  - New instances will follow new model
  - Affected instances will follow compliance graphs
Challenges

- Compliance criteria
- Suspension of affected instances
- Dealing with semantic objects
- Suitability of the business process

Compliance Criteria

- Containment Method
  - Structural containment of an instance sub graph in the new model
- Equivalence Method
  - Compliance can be found, even when strict structural containment cannot be established due to certain classes of equivalent structures

Equivalence Classes

- Decomposition
- Redundancy
- Ordering
- Deletion
- Transformation
- Property

Decomposition Class

Redundancy Class

Ordering Class
Deletion Class

Transformation Class

Property Class

Adaptive Workflows

ability of the workflow processes to react to exceptional circumstances

• Can exceptions be defined?
• How to model exceptions?
• How to detect, diagnose and resolve an exception?

Exception Classes

• Embedded
  – Exceptions which are embedded within the process model become part of the core process and generally are not even considered as exceptions.
• Separated
  – Exceptions which are implemented through exception rules and/or exception workflows constitute the set of useful exceptions for the process.
• Unanticipated
  – The business process can not be completely captured through its core and exception processes because of the existence of unanticipated (or true) exceptions.

Example

Conference Management

– Several participants: conference organizers, the program committee, authors, publishers and sponsors
– Block object represents complex activity
### Embedded Exceptions

**Possible Exceptions in Publication Block**
- Authors are unable to view the author kit attachments
- The author sends the paper but does not fully comply with the formatting guidelines
- The author does not send the paper at all, or after the deadline

### Separated Exceptions

*Exception Workflow*
- The instantiation of the exception workflow will (temporarily) suspend the corresponding workflow of the exception raising instance, the execution of which may, or may not resume later.
  - Creating an instance of this workflow would cause the conference workflow instance for that paper to be terminated.

### Unanticipated Exceptions

- Unanticipated exceptions are *True* exceptions
- Similar to dynamic modification, except that all active instances will not be affected
- Change will affect one or a few instances
- Nature of exception will dictate if exception raising instance is to be modified or handled outside the system

### Exception Handling Phases

- **Detection (Event)**
  - Failure Conditions
    - Task level
    - Process level
- **Diagnosis (Condition)**
  - Exception Rules
  - Control Flow
  - External
- **Resolution (Action)**
  - Exception Workflow

Mechanism is required to manage the signalling between the triggering objects and the WFMS. The workflow management system then has to search for the given failure condition and find the corresponding exception handling process. The invocation of exception handling process takes place, which may require transfer of control to the exception workflow, where required, changing the node states of the triggering tasks in the exception raising instance and finally resuming the execution of the exception raising instance.

### Resuming Execution

<table>
<thead>
<tr>
<th>Case</th>
<th>nodestate (n, i) = Active</th>
<th>nodestate (n, i) = Scheduled</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>n is aborted and control never returns to i</td>
<td>Control never returns to i</td>
</tr>
<tr>
<td>B</td>
<td>Control returns to i and n is (re)activated</td>
<td>Control returns to i and n is (re)scheduled</td>
</tr>
<tr>
<td>C</td>
<td>Control returns to i, but execution is resumed at a different point</td>
<td>Control returns to i, but execution is resumed at a different point</td>
</tr>
<tr>
<td>D</td>
<td>Control is not diverted from n, and n executes in parallel with e</td>
<td></td>
</tr>
</tbody>
</table>

The exception workflow of paper withdrawal in the conference workflow.
Flexible Workflows

ability of the workflow process to execute on the basis of a loosely specified model
• How to define a flexible model?
• What degree will keep a balance between flexibility and control?

Flexible Workflows

• Motivation
  – It may not be desirable or even possible to define the entire process before execution
  – There are flexible business processes where a prescriptive model will compromise the process goals
• Applications
  – Healthcare
  – Tertiary Education
  – Customer Relationship Management (CRM)

Achieving Flexibility

• Traditional workflow technology applicable only on ‘well structured’ and ‘well defined’ processes
• Achieving flexibility using traditional WF technology
  – Flexibility by definition
  – Flexibility by granularity
• Approaches for extending traditional workflow functionality
  – Extended modelling structures
  – Flexible workflows

Pockets of Flexibility

• Simple process modelling language as a foundation
• Loosely specified models
• Complex functional requirements through ‘pockets of flexibility’ structures
• Build and execute “flexible pockets” of workflows at runtime
• Making the process of change part of the workflow process

Modelling the Pocket

• A defined core process containing
  – Identifiable (pre-defined) workflow activities and control dependencies
  – Pockets of flexibility within the process, represented as a special workflow activity called the build activity, and consisting of:
    • Set of workflow fragments, where a workflow fragment may consist of a single activity, or a sub-process
    • Set of rules or build constraints for concretizing the pocket with a valid composition of workflow fragments.

Example in Healthcare
**Example in Healthcare**

- **Flexible Workflow Model**

  - **Instance Template**
    - An instance template is an instance specific copy of the process model.
    - Typically in systems that have little change support, the instance template would be an exact replica of the original process model.
    - The pocket’s framework attempts to make the instance template a dynamic object which may be changed several times during the life of an instance.
    - The definition of the instance template must be controlled.

---

**Challenges**

- How to model “pockets of flexibility”?
- Defining constraints for allowable runtime building?
- Engine capabilities for handling dynamic modifications of instances (templates) at runtime?
- End user change modelling: graphical or interactive?

---

**Constraint Specification**

- Building may be constrained by several factors including:
  - the data relating to that instance,
  - the stage of execution of the instance,
  - temporal constraints,
  - fragments available for building,
  - and the business rules of the particular application for which the template is being defined.
- These constraints are distinct from strong constraints.
  - Where as strong constraints will map to one and only one valid construct in the workflow model,
  - these weak or flexible constraints may map to several constructs.
- The manifestation of flexibility through build constraints is the key to providing configurable process models.

---

**Constraint Types**

- **Containment Constraints**
  - The constraints belonging to the containment class identify conditions under which fragments can and cannot be contained in the resulting templates.
  - For example, a fragment A cannot be included when a fragment B is included.
- **Structural Constraints**
  - The constraints belonging to the structural class impose restriction on how fragments can be composed in the templates.
  - Fragment A and B must always be done in sequence.

---

**Unified Framework**

Can we provide a framework that can support the combined methods for dynamism, adaptation and flexibility?

- Concept of templates may hold the answer.
- Many open and interesting questions still.
Change Policies

Three dimensions of change (Dynamism, Adaptability and Flexibility) differentiated through Change Policies

- **Flush**: Instances are allowed to complete
- **Abort**: Instances are terminated before completion
- **Migrate**: Instances are ‘switched’ to new model
- **Adapt**: Instances have to divert from the model
- **Build**: Instance execution defines the model

Scope of the Policies

<table>
<thead>
<tr>
<th>Policy</th>
<th>Model Changed?</th>
<th>Affected Instances</th>
<th>Compliant Instances</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flush</td>
<td>Y</td>
<td>None</td>
<td>-</td>
</tr>
<tr>
<td>Abort</td>
<td>Y/N</td>
<td>Some</td>
<td>None</td>
</tr>
<tr>
<td>Migrate</td>
<td>Y</td>
<td>All</td>
<td>Some</td>
</tr>
<tr>
<td>Adapt</td>
<td>N</td>
<td>Some</td>
<td>Some</td>
</tr>
<tr>
<td>Build</td>
<td>Y/N</td>
<td>Some</td>
<td>All</td>
</tr>
</tbody>
</table>

Unified Framework

<table>
<thead>
<tr>
<th>Dimension</th>
<th>Policy</th>
<th>Description</th>
<th>Mechanism</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dynamism</td>
<td>Migrate (Flush, Abort)</td>
<td>Represents a permanent change in the process, affecting many or all active instances. May affect executed part and hence cause loss of work</td>
<td>Compliance graphs</td>
</tr>
<tr>
<td>Adaptability</td>
<td>Adapt</td>
<td>Represents a deviation of one or more instances from the process model. May affect executed part and hence cause loss of work</td>
<td>Exception processes</td>
</tr>
<tr>
<td>Flexibility</td>
<td>Build</td>
<td>Represents a customization of an instance to specific requirements. Does not affect executed part</td>
<td>Build Constraints</td>
</tr>
</tbody>
</table>

Suggested Reading
