Temporal Workflow Management Systems

Carlo Combi\textsuperscript{1}  Giuseppe Pozzi\textsuperscript{2}

\textsuperscript{1}Department of Computer Science, University of Verona
\textsuperscript{2}Department of Electronics and Information, Politecnico of Milan

Padova 2006
1. Introduction
   - Business Processes and WfMSs
   - Goal of the work

2. Managing temporal aspects in workflow systems
   - The organizational model
   - The process model
   - The information model

3. Architectures for a temporal WfMS
   - Implementation issues

4. Discussions and conclusions
Any *business process* requires the coordinated execution of single activities to achieve a common goal: a *workflow* formally describes these activities, including criteria to assign single activities to executing units.

Workflow management systems, *WfMS*, are software systems supporting the execution of workflow instances. Most WfMSs use a database management system (DBMS) based on the relational model.
Workflow systems

- Workflows are activities involving the coordinated execution of
  - single tasks;
  - performed by different agents.

- The workflow specification describes the component tasks, their controlled and coordinated execution, the processing entities.

- Workflow management systems (WfMS) support the automatic execution of workflows. A WfMS provides:
  - the process model;
  - the information model;
  - the organizational model.
Workflow systems

- Workflows are activities involving the \textit{coordinated execution} of
  - single \textit{tasks};
  - performed by different \textit{agents}.

- The \textit{workflow specification} describes the component tasks, their controlled and coordinated execution, the processing entities.

- Workflow management systems (\textit{WfMS}) support the automatic execution of workflows. A WfMS provides:
  - the process model;
  - the information model;
  - the organizational model.
Workflow systems

- Workflows are activities involving the *coordinated execution* of
  - single *tasks*;
  - performed by different *agents*.

- The *workflow specification* describes the component tasks, their controlled and coordinated execution, the processing entities.

- Workflow management systems (*WfMS*) support the automatic execution of workflows. A WfMS provides:
  - the process model;
  - the information model;
  - the organizational model.
WfMS data models and temporal information

- **Organizational model.** Temporal aspects are mainly related to working days and hire time of the agent.
- **Process model.** Temporal aspects are related to changes of a schema of a business process.
- **Information model.** The information model considers both process specific and historical data. The first ones are data collected by the case (i.e., a process instance). The latter ones describe the history of the cases managed by the WfMS.
Goal of the work

- Discuss *temporal aspects for WfMS data models*
  - *Valid Time;*
  - *Temporal Constraints.*
- Discuss *different architectures for a temporal WfMS.*
- Deal with some *implementation issues.*
Outline

1. Introduction
   - Business Processes and WfMSs
   - Goal of the work

2. Managing temporal aspects in workflow systems
   - The organizational model
   - The process model
   - The information model

3. Architectures for a temporal WfMS
   - Implementation issues

4. Discussions and conclusions
The temporal organizational model

- **Agent**
  - Unavailability
  - Availability
  - Role
  - Actor
  - Function
  - Team
  - Group
  - Supervisor
  - Accessibility
  - Unaccessibility

- **Combi and Pozzi**
  - Temporal WfMS
### The temporal organizational model

#### Example

**Agent**

<table>
<thead>
<tr>
<th>Id_A</th>
<th>Name</th>
<th>Email</th>
<th>VT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Jane Austen</td>
<td><a href="mailto:jane.austen@literature.com">jane.austen@literature.com</a></td>
<td>[10/01/2000 ÷ +∞]</td>
</tr>
<tr>
<td>2</td>
<td>James Joyce</td>
<td><a href="mailto:james.joyce@literature.com">james.joyce@literature.com</a></td>
<td>[10/01/2000 ÷ +∞]</td>
</tr>
<tr>
<td>3</td>
<td>Emily Bronte</td>
<td><a href="mailto:emily.bronte@literature.com">emily.bronte@literature.com</a></td>
<td>[01/03/2001 ÷ +∞]</td>
</tr>
<tr>
<td>4</td>
<td>Charlotte Bronte</td>
<td><a href="mailto:charlotte.bronte@literature.com">charlotte.bronte@literature.com</a></td>
<td>[10/01/2001 ÷ +∞]</td>
</tr>
<tr>
<td>5</td>
<td>Charles Dickens</td>
<td><a href="mailto:charles.dickens@literature.com">charles.dickens@literature.com</a></td>
<td>[10/04/2000 ÷ +∞]</td>
</tr>
<tr>
<td>6</td>
<td>Samuel Beckett</td>
<td><a href="mailto:samuel.beckett@literature.com">samuel.beckett@literature.com</a></td>
<td>[10/01/1999 ÷ +∞]</td>
</tr>
</tbody>
</table>

**Role**

<table>
<thead>
<tr>
<th>Id_R</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Secretary</td>
</tr>
<tr>
<td>2</td>
<td>Committee Member</td>
</tr>
<tr>
<td>3</td>
<td>Committee President</td>
</tr>
</tbody>
</table>

**Performance**

<table>
<thead>
<tr>
<th>Actor</th>
<th>Role</th>
<th>VT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>[10/01/2000 ÷ +∞]</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>[10/01/2000 ÷ +∞]</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>[01/03/2001 ÷ +∞]</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>[10/01/2001 ÷ 31/08/2002]</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>[10/04/2000 ÷ +∞]</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>[10/01/1999 ÷ +∞]</td>
</tr>
</tbody>
</table>
Availability/unavailability of an agent may present some periodicity.

**Example**
- "every Monday 8:00a.m. ÷ 4:30p.m.”;
- "the first Wednesday of every month 1:30p.m. ÷ 5:30 p.m.”;
- "every day 8:30a.m. ÷ 4:30p.m.”.
The supported periodic expressions: notations and examples

<table>
<thead>
<tr>
<th>Expression</th>
<th>( P )</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>([P]/\text{Days in Weeks})</td>
<td>( x ) such that ( x \in [1,\ldots,7] )</td>
<td>([1]/\text{Days in Weeks} = ) every Monday</td>
</tr>
<tr>
<td></td>
<td>( 1..x ) such that ( x \in [2,\ldots,7] )</td>
<td>([1..3]/\text{Days in Weeks} = ) the first 3 days of the week</td>
</tr>
<tr>
<td></td>
<td>( x,y ) such that ( x,y \in [1,\ldots,7] )</td>
<td>([1,3]/\text{Days in Weeks} = ) every Monday and every Wednesday</td>
</tr>
<tr>
<td>([P]/\text{Days in Months})</td>
<td>( 1..p ) such that ( p \in [2,\ldots,M], M \in [28,\ldots,31] )</td>
<td>([1..5]/\text{Days in Months} = ) the first 5 days of the month</td>
</tr>
<tr>
<td></td>
<td>( p,t ) such that ( p,t \in [1,\ldots,M] ) ( M \in [28,\ldots,31] )</td>
<td>([2, 6]/\text{Days in Months} = ) day 2 and day 6 of the month</td>
</tr>
<tr>
<td></td>
<td>( w ) such that ( w \in [1,\ldots,7] )</td>
<td>([1]/\text{Days in Months} = ) the first Monday of the month</td>
</tr>
<tr>
<td></td>
<td>( p ) such that ( p \in [1,\ldots,M], M \in [28,\ldots,31] )</td>
<td>([20]/\text{Days in Months} = ) day 20 of the month</td>
</tr>
<tr>
<td>([P]/\text{Weeks in Months})</td>
<td>( 1..q ) such that ( q \in [2,\ldots,5] )</td>
<td>([1..2]/\text{Weeks in Months} = ) the first 2 weeks of the month</td>
</tr>
<tr>
<td></td>
<td>( q ) such that ( q \in [1,\ldots,5] )</td>
<td>([1]/\text{Weeks in Months} = ) the first week of the month</td>
</tr>
<tr>
<td></td>
<td>( q,r ) such that ( q,r \in [1,\ldots,5] )</td>
<td>([2,4]/\text{Weeks in Months} = ) the second and the fourth week of the month</td>
</tr>
</tbody>
</table>
### Example

#### Availability

<table>
<thead>
<tr>
<th>Id_Av</th>
<th>Start_Time</th>
<th>End_Time</th>
<th>Time_Av</th>
<th>VT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>09:00</td>
<td>12:00</td>
<td>DAILY_TIME</td>
<td>[10/01/2000 ÷ +∞]</td>
</tr>
<tr>
<td>2</td>
<td>10:00</td>
<td>13:30</td>
<td>DAILY_TIME</td>
<td>[10/01/2000 ÷ +∞]</td>
</tr>
<tr>
<td>3</td>
<td>09:00</td>
<td>14:00</td>
<td>DAILY_TIME</td>
<td>[01/03/2001 ÷ +∞]</td>
</tr>
<tr>
<td>4</td>
<td>10:00</td>
<td>15:30</td>
<td>PERIODIC_TIME</td>
<td>[10/04/2000 ÷ +∞]</td>
</tr>
<tr>
<td>5</td>
<td>12:00</td>
<td>17:30</td>
<td>DAILY_TIME</td>
<td>[10/01/2000 ÷ +∞]</td>
</tr>
<tr>
<td>6</td>
<td>09:00</td>
<td>17:30</td>
<td>DAILY_TIME</td>
<td>[10/01/1999 ÷ +∞]</td>
</tr>
</tbody>
</table>

#### Accessibility

<table>
<thead>
<tr>
<th>Agent</th>
<th>Availability</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
</tr>
</tbody>
</table>
Availabilities and unavailabilities

Example

| Unav | | | | | |
|---|---|---|---|---|
| Id.Unav | Description | Start.Time | End.Time | Time.Unav | VT |
| 1 | Holiday | 09:00 | 17:30 | DAILY_TIME | [03/08/2001 ÷ 03/08/2001] |
| 2 | Business Trip | 09:00 | 17:30 | PERIODIC_TIME | [01/09/2002 ÷ 30/10/2002] |
| 3 | Holiday | 09:00 | 17:30 | DAILY_TIME | [10/08/2001 ÷ 20/08/2005] |
| 4 | Business Trip | 09:00 | 17:30 | PERIODIC_TIME | [01/09/2003 ÷ 30/11/2003] |

Inaccessibility

<table>
<thead>
<tr>
<th>Agent</th>
<th>Unavailability</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
</tr>
</tbody>
</table>

Periodic.Time

<table>
<thead>
<tr>
<th>Id</th>
<th>Type</th>
<th>Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Av</td>
<td>[5,6]/Days.in.Weeks</td>
</tr>
<tr>
<td>2</td>
<td>Unav</td>
<td>[2]/Weeks.in.Months</td>
</tr>
<tr>
<td>4</td>
<td>Unav</td>
<td>[1..5]/Days.in.Months</td>
</tr>
</tbody>
</table>
The process model: an example

The enrollment process of graduated students applying for PhD candidate position: starting from September 1\textsuperscript{st}, 2003, any received application leads to an interview of the applicant. After a few days, the university realizes that interviewing any student is extremely expensive. After September 30\textsuperscript{th}, 2003, the new adopted process model states that applicants’ CVs are analyzed first: applicants whose CV is passed will be interviewed, only.
Changes in the process model

The enrollment process: first version

StudentEnrollment

Workflow variables:
StudentName
InterviewResult

ReceiveApplication
Application is received

Interview
Applicant is interviewed

R1

InterviewResult = "no"
RejectApplication
Applicant is rejected

InterviewResult = "yes"
AcceptApplication
Applicant is accepted

Combi and Pozzi
Temporal WfMS
Changes in the process model

The enrollment process: second version

- **StudentEnrollment**
  - Workflow variables:
    - StudentName
    - CVResult
    - InterviewResult

- **ReceiveApplication**
  - Application is received

- **AnalyzeCV**
  - CV is analyzed

  - CVResult = "OK"
  - CVResult = "Reject"

  **R2**

- **Interview**
  - Applicant is interviewed

  - InterviewResult = "no"
  - InterviewResult = "yes"

  **R1**

- **RejectApplication**
  - Applicant is rejected

- **AcceptApplication**
  - Applicant is accepted

- **RejectAndThank**
  - Applicant is rejected

Combi and Pozzi

Temporal WfMS
Temporal constraints in the process model

The enrollment process: temporal constraints

**StudentEnrollment**
- **Workflow variables:** StudentName, CVResult, InterviewResult
- **S[3, 12]E days**

**ReceiveApplication**
- **[5, 10]min**
- Application is received
- **[2, 24]hours**

**AnalyzeCV**
- **[2, 3]hours**
- CV is analyzed

**Interview**
- **[20, 40]min**
- Applicant is interviewed
- **[0, 10]min**
- InterviewResult = "no"

**RejectApplication**
- **[2, 5]min**
- Applicant is rejected

**AcceptApplication**
- **[10, 15]min**
- Applicant is accepted

**RejectAndThank**
- **[2, 10]min**
- Applicant is rejected
### Example

#### Workflow

<table>
<thead>
<tr>
<th>SchemaName</th>
<th>StartTask</th>
<th>VT</th>
</tr>
</thead>
<tbody>
<tr>
<td>StudentEnrollment</td>
<td>ReceiveApplication</td>
<td>[01/09/2003 ÷ +∞]</td>
</tr>
</tbody>
</table>

#### WorkTask

<table>
<thead>
<tr>
<th>SchemaName</th>
<th>TaskName</th>
<th>Role</th>
<th>Tcons</th>
<th>VT</th>
</tr>
</thead>
<tbody>
<tr>
<td>StudentEnrollment</td>
<td>ReceiveApplication</td>
<td>1</td>
<td>...</td>
<td>[01/09/2003 ÷ +∞]</td>
</tr>
<tr>
<td>StudentEnrollment</td>
<td>Interview</td>
<td>2</td>
<td>...</td>
<td>[01/09/2003 ÷ +∞]</td>
</tr>
<tr>
<td>StudentEnrollment</td>
<td>RejectApplication</td>
<td>3</td>
<td>...</td>
<td>[01/09/2003 ÷ +∞]</td>
</tr>
<tr>
<td>StudentEnrollment</td>
<td>AcceptApplication</td>
<td>3</td>
<td>...</td>
<td>[01/09/2003 ÷ +∞]</td>
</tr>
<tr>
<td>StudentEnrollment</td>
<td>AnalyzeCV</td>
<td>2</td>
<td>...</td>
<td>[01/10/2003 ÷ +∞]</td>
</tr>
<tr>
<td>StudentEnrollment</td>
<td>RejectAndThank</td>
<td>3</td>
<td>...</td>
<td>[01/10/2003 ÷ +∞]</td>
</tr>
</tbody>
</table>
### Temporal tables for the process model (2)

**Example**

<table>
<thead>
<tr>
<th>SchemaName</th>
<th>TaskName</th>
<th>NextTask</th>
<th>Tcons</th>
<th>VT</th>
</tr>
</thead>
<tbody>
<tr>
<td>StudentEnrollment</td>
<td>ReceiveApplication</td>
<td>Interview</td>
<td>...</td>
<td>[01/09/2003 ÷ 30/09/2003]</td>
</tr>
<tr>
<td>StudentEnrollment</td>
<td>Interview</td>
<td>R1</td>
<td>...</td>
<td>[01/09/2003 ÷ +∞]</td>
</tr>
<tr>
<td>StudentEnrollment</td>
<td>RejectApplication</td>
<td>end_flow</td>
<td>...</td>
<td>[01/09/2003 ÷ +∞]</td>
</tr>
<tr>
<td>StudentEnrollment</td>
<td>AcceptApplication</td>
<td>end_flow</td>
<td>...</td>
<td>[01/09/2003 ÷ +∞]</td>
</tr>
<tr>
<td>StudentEnrollment</td>
<td>RejectAndThank</td>
<td>end_flow</td>
<td>...</td>
<td>[01/10/2003 ÷ +∞]</td>
</tr>
<tr>
<td>StudentEnrollment</td>
<td>ReceiveApplication</td>
<td>AnalyzeCV</td>
<td>...</td>
<td>[01/10/2003 ÷ +∞]</td>
</tr>
<tr>
<td>StudentEnrollment</td>
<td>AnalyzeCV</td>
<td>R2</td>
<td>...</td>
<td>[01/10/2003 ÷ +∞]</td>
</tr>
</tbody>
</table>
### Example

**RoutingTask**

<table>
<thead>
<tr>
<th>SchemaName</th>
<th>RTName</th>
<th>Type</th>
<th>VT</th>
</tr>
</thead>
<tbody>
<tr>
<td>StudentEnrollment</td>
<td>R1</td>
<td>mutualex_fork</td>
<td>[01/09/2003 ÷ +∞]</td>
</tr>
<tr>
<td>StudentEnrollment</td>
<td>R2</td>
<td>mutualex_fork</td>
<td>[01/10/2003 ÷ +∞]</td>
</tr>
</tbody>
</table>

**AfterFork**

<table>
<thead>
<tr>
<th>SchemaName</th>
<th>ForkTask</th>
<th>NextTask</th>
<th>Cond</th>
<th>VT</th>
</tr>
</thead>
<tbody>
<tr>
<td>StudentEnrollment</td>
<td>R1</td>
<td>RejectApplication</td>
<td>InterviewResult = “no”</td>
<td>[01/09/2003 ÷ +∞]</td>
</tr>
<tr>
<td>StudentEnrollment</td>
<td>R1</td>
<td>AcceptApplication</td>
<td>InterviewResult = “yes”</td>
<td>[01/09/2003 ÷ +∞]</td>
</tr>
<tr>
<td>StudentEnrollment</td>
<td>R2</td>
<td>Interview</td>
<td>CVResult = “OK”</td>
<td>[01/10/2003 ÷ +∞]</td>
</tr>
<tr>
<td>StudentEnrollment</td>
<td>R2</td>
<td>RejectAndThank</td>
<td>CVResult = “Reject”</td>
<td>[01/10/2003 ÷ +∞]</td>
</tr>
</tbody>
</table>
### Temporal tables for the information model

#### Example

**StudentEnrollmentData**

<table>
<thead>
<tr>
<th>CaseId</th>
<th>StudentName</th>
<th>CV Result</th>
<th>InterviewResult</th>
<th>VT</th>
</tr>
</thead>
<tbody>
<tr>
<td>27</td>
<td>Marple T.R.</td>
<td>n/a</td>
<td>“yes”</td>
<td>[09/09/2003 ÷ +∞]</td>
</tr>
<tr>
<td>89</td>
<td>Wallace E.S.</td>
<td>“yes”</td>
<td>“no”</td>
<td>[03/10/2003 ÷ +∞]</td>
</tr>
</tbody>
</table>

**CaseHistory**

<table>
<thead>
<tr>
<th>CaseId</th>
<th>SchemaName</th>
<th>Resp</th>
<th>Start</th>
<th>End</th>
<th>VT</th>
</tr>
</thead>
<tbody>
<tr>
<td>89</td>
<td>StudentEnrollment</td>
<td>3</td>
<td>09:00</td>
<td>10:19</td>
<td>[03/10/2003 ÷ 31/10/2003]</td>
</tr>
</tbody>
</table>

**TaskHistory**

<table>
<thead>
<tr>
<th>CaseId</th>
<th>TaskName</th>
<th>FinalState</th>
<th>Ag</th>
<th>Start</th>
<th>End</th>
<th>VT</th>
</tr>
</thead>
<tbody>
<tr>
<td>89</td>
<td>ReceiveApplication</td>
<td>Completed</td>
<td>3</td>
<td>09:00</td>
<td>09:07</td>
<td>[03/10/2003 ÷ 03/10/2003]</td>
</tr>
<tr>
<td>89</td>
<td>AnalyzeCV</td>
<td>Completed</td>
<td>4</td>
<td>16:01</td>
<td>16:12</td>
<td>[10/10/2003 ÷ 10/10/2003]</td>
</tr>
<tr>
<td>89</td>
<td>Interview</td>
<td>Completed</td>
<td>6</td>
<td>09:32</td>
<td>10:03</td>
<td>[31/10/2003 ÷ 31/10/2003]</td>
</tr>
<tr>
<td>89</td>
<td>RejectApplication</td>
<td>Completed</td>
<td>5</td>
<td>10:03</td>
<td>10:19</td>
<td>[31/10/2003 ÷ 31/10/2003]</td>
</tr>
</tbody>
</table>
Outline

1. Introduction
   - Business Processes and WfMSs
   - Goal of the work

2. Managing temporal aspects in workflow systems
   - The organizational model
   - The process model
   - The information model

3. Architectures for a temporal WfMS
   - Implementation issues

4. Discussions and conclusions
Workflow components

Workflow Management Systems (WfMSs) are complex software systems and include many different components, such as:

- the *workflow engine*, made of the interpreter of the process definition language (PDL) and of the workflow scheduler;
- the process model designer unit, which helps the workflow designer to suitably define a process model according to the supported PDL;
- the resource management unit (also known as *resource executive*), to assign tasks to executing agents;
- the database connectivity unit, to access data stored into a DBMS;
- the transaction manager;
- the e-mail feeder, to send agents messages and attached documents;
- the web server and the worklist server.
Workflow components and temporalities

- The *workflow engine* reads the process model and schedules the activities, looking for the successor(s) of a task as soon as it completes.

- In the real practice, the process model of a workflow goes through different *refinements over time*, e.g. due to corrective and to perfective maintenance, leading to different versions of the same process model.

- The *wash-out policy* (wait for completion of all running cases of the old schema before any new case can be started according to the latest schema) is not always acceptable.
Workflow components and temporalities

- A reasonable policy is that of completing all running cases according to the schema that was valid at their respective creation time, and to start new cases according to the latest available schema.

- Additionally the *workflow engine* manages the history of all the cases run by the WfMS, by suitably updating the CaseHistory and TaskHistory tables.

- The *resource executive* assigns tasks to agents: criteria for agent selection are statical (e.g. the role the agent must own, the agent’s availability in terms of working time during weekdays), and dynamical (e.g. workload balancing among agents over the last two weeks).
Some options:
- flat DBMSs vs fully-fledged temporal DBMSs (T-DBMSs);
- standard DBMSs vs active DBMSs.
1. A temporal architecture

Diagram:

- Web Server
- Worklist server
- Engine Interface
- Agent's Client
- WfEngine
- Temporal Layer
- Resource Executive
- DBMS

Combi and Pozzi
Temporal WfMS
1. Temporal queries

Example

Work load balancing (over time)

```
SELECT SEQUENCED A.Id_A, T.TaskName
FROM Agent A, Accessibility C, Availability B, TaskHistory T, Performance P1, Performance P2, Role R
  FROM TaskHistory T1
  WHERE T1.Agent = A.Id_A)
```
2. A temporal active architecture
2. Temporal active features for workflows

Assuming that there is a straight connection between the completed task and its successor (i.e., there is no fork or join connector in between), the trigger FindSuccessor can be expressed in the extended Chimera-Exception language as:

```
define trigger FindSuccessor
  event insert into TaskHistory
  condition TaskHistory TH, ToAssign TA,
             WorkTask WT, CaseHistory CH, Next N,
             occurred(insert(TH)),
             TH.FinalState='Completed', CH.CaseID=TH.CaseId,
             N.SchemaName=WT.SchemaName, N.TaskName=TH.TaskName,
             VALID(WT) CONTAINS BEGIN(VALID(CH)),
             VALID(N) CONTAINS BEGIN(VALID(CH))
  action insert into ToAssign(CaseId,TaskName,Role,AgentId)
          values TA.CaseId=TH.CaseId,
                 TA.TaskName=N.NextTask,
                 TA.Role=WT.Role,
                 TA.AgentId=NULL,
                 VALID PERIOD(current_time(), +∞)
end trigger
```
3. An active architecture with a temporal layer

![Diagram of an active architecture with a temporal layer]

- Agent's Client
- Web Server
- Worklist server
- DB Interface
- Choose Successor
- Resource Executive
- Temporal Layer
- A-DBMS
The trigger `ActivateChooseSuccessor` is fired when an `insert` is performed over the `TaskHistory` table, and that simply activates a stored procedure, namely `ChooseSuccessor`.

```sql
define trigger ActivateChooseSuccessor
    event insert into TaskHistory
    condition NULL
    action activate(ChooseSuccessor)
end trigger
```
Implementation issues

1. Agent's Client
2. Worklist server
3. DB Interface
4. TimeDB
5. Choose Successor
6. Resource Executive
7. Oracle Server

- Combi and Pozzi
- Temporal WfMS
Implementation issues

Let us consider the functionality of choosing a successor as performed by the workflow engine.

- The agent sends back to the worklist server the completed task: the DB interface module automatically updates the workflow history tables via the TimeDB layer.

- The Oracle trigger `ActivateChooseSuccessor` has to invoke the Java stored procedure `LaunchChooseSuccessor`, which in turn invokes the `ChooseSuccessor` procedure, also written in Java, via a remote method invocation (RMI).

- The stored procedure is executed by the internal Java virtual machine of Oracle, while the `ChooseSuccessor` procedure needs for being executed outside Oracle, so that the procedure can access the database via the TimeDB layer.
In this way, the `ChooseSuccessor` procedure can perform temporal queries that could not be performed if inside the Oracle environment.

We define a public Java class, we register it inside the database of the procedures, and define the Oracle trigger, obtaining thus the following:

```java
public class LaunchChooseSuccessor {
    ... 
};
create procedure LaunchChooseSuccessor
    as language Java ...
create trigger ActivateChooseSuccessor
    after insert TaskHistory
    for each row LaunchChooseSuccessor
```
This work around, however, deepens the portability problems that feature the definition of a trigger over a DBMS:

- (a) no SQL-like standard is available yet for the trigger definition language, and the adoption of a new DBMS requires a new set of triggers;
- (b) very often the language used for the stored procedures is strictly DBMS-dependent, and again a change in the adopted DBMS requires rewriting the stored procedures.
Introduction

1. Business Processes and WfMSs
2. Goal of the work

Managing temporal aspects in workflow systems

2. The organizational model
3. The process model
4. The information model

Architectures for a temporal WfMS

3. Implementation issues

4. Discussions and conclusions
In this talk we introduced and discussed temporalities for the process, organization, and information models of workflow management systems.

We analyzed some architectures for a temporal workflow management system, featuring the management of the introduced temporalities.
Future work

- Verifying *run-time consistency* of cases wrt temporal constraints.
- Extending the engine to adopt many different schema migration policies: *concurrent to completion* which is the only one currently implemented, *conditional or unconditional migration to final workflow*, and *migration to ad-hoc workflow*.
- Defining a temporal interpreter for active rules.
- Extending obtained results to an open source WfMS.
References

C. Combi and G. Pozzi.
Towards temporal information in workflow systems.

C. Combi and G. Pozzi.
Temporal conceptual modeling of workflows.

C. Combi and G. Pozzi.
Architectures for a temporal workflow management system.

C. Combi and G. Pozzi.
Task Scheduling for a Temporal Workflow Management System.
Managing temporal aspects in WfMSs

Architectures for a temporal WfMS

Discussions and conclusions

References

Temporal reasoning in workflow systems.

J. Eder, W. Gruber, and E. Panagos.
Temporal modeling of workflows with conditional execution paths.

Temporal reasoning in workflow systems.
References

