A BUSINESS PROCESS MODELLING ENVIRONMENT FOR DESCRIBING DYNAMIC PROCESSES

M. Nikolaidou
Department of Informatics, University of Athens
Panepistimiopolis 15784, Athens, Greece
E-mail: mara@di.uoa.gr

D. Anagnostopoulos
Harokopion University of Athens
El. Venizelou Str, 17671 Athens, Greece
E-mail: dimosthe@hua.gr

A.Tsalgatidou
Department of Informatics, University of Athens
Panepistimiopolis 15784, Athens, Greece
E-mail: afrodite@di.uoa.gr

M. Kitrinakis
Department of Informatics, University of Athens
Panepistimiopolis 15784, Athens, Greece
E-mail: mkitrina@byte.gr

KEYWORDS
Dynamic Process Modelling, Petri-nets, Case Study.

ABSTRACT
In order to survive in the competitive business environment introduced by the technological burst of the last decade, most organisations automated their business activities. The efficient automation of business activities leads to improving their performance and enabling enterprise-wide monitoring and coordination. Business process (BP) modelling is an effective tool towards analysing the behaviour of business activities and facilitating their accurate and complete representation in an information system. Four types of business processes are usually studied in the literature: production, administrative, ad-hoc and collaborative. Administrative and production BPs refer to bureaucratic procedures that include well-defined steps and can be easily described by conventional modelling tools. Ad-hoc and collaborative processes deal with unique or loosely defined conditions and are dynamic in nature. Thus, they are not easily modelled.

In this paper, we present a BP modelling environment for efficiently describing dynamic processes. BP modelling is based on extensions of the Modified Petri-Net (MPN) model and enables the description of activities in multiple levels of abstraction. The environment build to support BP modelling is based upon Lotus Domino/Notes groupware product. In order to depict MPN functionality, the experience obtained from its deployment for modelling dynamic banking activities is also presented.

1. INTRODUCTION
Business process modelling provides means for describing process-oriented systems and decomposing them into manageable parts. Business processes (BPs) are collections of activities with a common objective, such as fulfilling a business contract or satisfying a customer need. Business process definition (i.e. a description of a BP at a high conceptual level necessary for process understanding, evaluation and redesign) requires a well-defined model, that provides a set of concepts appropriate to describe BPs [Zelm et al, 1995]. The model should be rich enough and enable process validation (e.g. by simulation or static analysis) to decide whether the process definition accurately represents the system under study. BP modelling is a significant tool for re-engineering organisational procedures and it is usually followed by BP automation, aiming at improving business process performance and enabling organisation-wide monitoring and coordination. An automated BP is referred to as a workflow, while a Workflow Management System (WMS) is the software used for its coordination and control [Mohan et al, 2000]. WMSs also provide a set of interfaces to users and applications involved in the workflow progress. For efficient workflow development, one should start with defining and understanding business processes (BP modelling), before specifying and implementing the corresponding workflow applications (BP automation). Provision of generic and flexible modelling methods is thus required both at BP modelling and BP automation levels.

Several methods have been suggested for BP modelling, most of which are based on textual programmable languages or graphical notations, such as dataflow diagrams, state transition diagrams, petri-nets and related notations. Combination of different BP modelling methods has also been examined to give new, enhanced approaches [Abeyesinghe and Phalp, 1996].

Four types of business processes are usually studied in the literature [Alonso and Mohan, 1997]. These are: production, administrative, ad-hoc and collaborative. Administrative and production BPs refer to bureaucratic procedures that include well-defined steps and are controlled by a set of well-known rules. Such processes can be easily described by conventional modelling tools and are usually automated using a WMS [Hollingsworth, 1995].

Ad-hoc (dynamic) processes are similar to administrative processes, except for the fact that they deal with unique or loosely defined conditions, which are not easily modelled. Collaborative processes are characterised by the number of
participants involved and the synchronisation needed, and are handled more effectively using groupware technology.

In this paper, we present a BP modelling environment for efficiently describing dynamic processes. BP modelling is based on extensions of the Modified Petri-Net (MPN) model [Tsalgatidou et al, 1996] and enables the description of activities in multiple levels of abstraction. The software environment build to support the BP modelling is based upon Lotus Domino/Notes groupware product [Lotus Co, 2000]. Notes database is used to implement MPN repositories. In order to depict MPN functionality, the experience obtained from its deployment for modelling dynamic banking activities is also presented.

The rest of the paper is organised as follows: In section 2 an overview of BP modelling approaches is presented. Section 3 provides an analytical description of the extended MPN model and the graphical environment built to support it. In section 4, we discuss our experience using it to represented dynamic banking activities. Conclusions reside in Section 5.

2. BUSINESS PROCESS MODELLING

The BP model must fulfil the following requirements:

- Enable the accurate description of dynamic processes
- Facilitate the evaluation of BPs through simulation
- Support the direct mapping of entities into the workflow environment to minimise implementation cost (optional).

Numerous modelling methodologies, such as IDEF0 [Marca and McGowan, 1993] and RADs [Ould, 1992], provide the means to understand the behaviour of static systems [Starke, 1994]. Production and administrative activities usually fall in this category. BP modelling is also used to tackle the problem of changing or evolving systems [Phalp, 1998]. Ad-hoc processes can be viewed as an evolving system. The business model used to depict such systems should be flexible enough to facilitate accurate description of business activities. BP modelling approaches based on extending Petri-net functionality [Murata, 1989] can provide better solutions for this kind of problem [Oberweis, 1996], [Tsalgatidou et al, 1996], since they focus on depicting the relationship between activities and resources rather than the relationship between activities. A Petri-net consists of places and transitions between them. Arcs are used to denote relations between places and transitions. A transition is performed whenever all its input places are filled with tokens. When a transition is completed, output places are filled with tokens. Transitions depict processes and their components as activities and tasks, respectively, while places represent resources [Jensen, 1992]. An overview of modelling approaches based on Petri-nets is included in [Tsalgatidou, 1996]. BP models based on Petri-Nets have also the advantage that they can be easily simulated using discrete event simulation [Rajala and Savolainen, 1996]. The modelling formalism adopted is the Multi-level Modified Petri-net (MPN) [Tsalgatidou et al, 1996], which is an extension of Coloured Petri-nets [Jensen, 1992]. The formal and executable nature of MPN models enables the employment of simulation techniques for validation purposes.

3. DYNAMIC BP MODELLING APPROACH

MPN is used for modelling BPs at various levels of abstraction. Transition decomposition depicts the decomposition of a BP to its activities, sub-activities and tasks and demonstrates the control and data flows between the different organisational units involved in the BP.

MPN facilitates the representation of dynamic BPs, as each activity is not connected with others, i.e. it does not follow nor is followed by another activity, as in IDEF0, and there is no activity ordering. Activities can be initiated whenever all input places are occupied by the appropriate token, i.e. whenever the necessary resources and participants are available. Upon completion, each activity provides tokens to its output places, i.e. releases the resources needed for the activation of another activity. Thus, both static and dynamic processes are described uniformly. It also provides a clear, visual representation of activity steps executed with the collaboration of many actors, facilitating the description of cooperative processes.

MPN also facilitates the description of an organisational model. Places in the Petri-net can be inscribed with organisational entities, such as actors and roles, making the integration among organisation models and process models smooth and tightly coupled. Since places can also be inscribed with resource and control objects, a desirable integration between control flow, data flow and the organisation model is attained. All entities inscribed in Petri-net places are stored as objects within the MPN Repository. The repository needed to support the MPN model is implemented using Lotus Notes (Lotus Co, 2000). Notes serves as a groupware platform and support object-based database services. The MPN repository is implemented as a Notes database. This approach offers the advantage that the same dictionary can be used during both BP modelling and workflow implementation. In this way, workflow implementation cost is minimised, since the developer only adds code segments in the preconstructed Notes structures. The MPN model and the repository architecture are presented in the following paragraphs.

3.1 Extended MPN Model

A BP model should encapsulate information related to: (a) activities, (b) resources assigned to activities, i.e. objects necessary for the execution of activities, such as actors, documents, data, etc, (c) control of a BP which describes 'when' and 'which' activity is executed, (d) the flow of data in the process and (e) the organisational structure which consists of organisational units, people, roles, competence, etc [Tsalgatidou and Junginger, 1995]. These entities must be therefore mapped within MPN. The formal definition of MPN model is given in [Tsalgatidou et al, 1996]. The MPN modelling approach is based on the following principles:
The overall MPN model represents a specific BP and consists of different SubMPNs depicting the decomposition of the BP into a more detailed level.

Activities, that may either be simple tasks or further decomposed, are modelled as transitions. If the activity is further decomposed, a lower level MPN is used for its description. A script is related with each activity and represents the set of steps to be carried out during its execution. Scripts can be described using a high level language and are particularly useful in the case of simple tasks that are not further decomposed.

Control information, resources and performers required for the execution of activities are modelled as objects inscribing the MPN places. Control objects enable the representation of control flows within the process. Resource objects are data objects used by the process and enable the representation of data flows and data modification within the process limits. Resource objects are maintained in MPN Repository. Each resource is identified by properties and can be either simple or complex. Actors represent a position profile, e.g. manager or programmer, assigned to a specific employee within the organisational. More than one employee may be associated with one Actor. Roles group sets of duties and responsibilities assigned to a specific actor. Roles can be described in terms of other roles. Roles, Actors or Employees are required at the input places of a given transition, where the presence of specific operators (performers) is essential for its enactment.

In order to construct MPN models, a graphical interface was developed using Java. The GUI module, named MPN editor, communicates with Repository database stored within Domino Server. The same database is used to maintain activity description and activity decomposition scheme.

3.2. MPN Repository Implementation

According to Domino/Notes architecture, all entities defined within Notes platform are stored as Notes Objects. Different kinds of object classes can be defined using Notes Forms. Notes Templates are used to implement basic functionality. Based on these templates, it is possible to construct Notes Databases supporting the characteristics of specific applications. Both templates and databases are constructed using Notes programming tools. The authors suggest the following mapping of MPN entities within Lotus Notes, as depicted in table 1.

<table>
<thead>
<tr>
<th>MPN Entity</th>
<th>Notes Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Employee</td>
<td>Notes User and Employee form</td>
</tr>
<tr>
<td>Actor</td>
<td>Actor form</td>
</tr>
<tr>
<td>Role</td>
<td>Role form</td>
</tr>
<tr>
<td>Organisational entity</td>
<td>Organisational entity form</td>
</tr>
<tr>
<td>Resource</td>
<td>Resource form</td>
</tr>
<tr>
<td>BP definition</td>
<td>BP form</td>
</tr>
<tr>
<td>(new MPN model)</td>
<td></td>
</tr>
<tr>
<td>Activity definition</td>
<td>Activity form</td>
</tr>
<tr>
<td>Task definition</td>
<td>Supported Tasks form</td>
</tr>
</tbody>
</table>

Table 1: Mapping of MPN within Notes Platform

Resources defined using the Resource form are maintained in the Resource Notes Database. The corresponding form and view are automatically created for each resource defined. Forms are left blank so that the developer can customise them during workflow development.

Definition of BP, Activity and task (simple activities) is respectively supported using BP, Activity and Supported Tasks forms. Only the Supported Tasks form is used during workflow implementation. As an example of the structures created within CT, the formal description of the object class corresponding to “Activity” form is presented in Table 2. Activity_script is the code executed upon activation.

```
class Activity {
    Name: String;
    Parent_Activity: String;
    SubActivities: Activity[];
    Activity_Script: Script[];
    Guards: Script[];
    Input: String[];
    Output: String[];
    Performer: String[];
}
```

Table 2: Class Definition within Notes

Guards contain the conditions for triggering the activity and are realised as an array of script sentences that calculates to a Boolean value. Input/Output contain input and output places respectively.

4. USING MPN TO MODEL DYNAMIC ACTIVITIES

The banking sector is a competitive environment, where business process re-engineering is constantly needed. Business process modelling and automation is an effective tool towards this direction, improving the performance of business activities and enabling enterprise-wide monitoring and coordination. In this section, we present a case study of modelling business processes in the Loan Monitoring Department of a medium-sized Bank. Loan monitoring is a typical banking activity, which includes business processes concerning loan approval, collection of delinquent loans and initiation of appropriate legal claims. These processes are often performed in cooperation with external business partners, such as legal firms and brokers, have collaborative properties and are considered of dynamic nature. Their efficiency strongly depends on human operator experience and subjective criteria. The loan monitoring policy employed is a significant factor for determining profits. Thus, relevant business processes should always be monitored, evaluated and refined.

In the following, we discuss the Delinquent Loan Collection process, since it combines both ad-hoc and collaborative properties. Delinquent loan collection may be performed by any employee working in the corresponding Department. As Actors, we have defined the positions included in it: Department Manager, Section Manager, Group Leader and Collector. Collector role describes the task of collecting delinquent loans. Other roles are Administrator, Case Assignor, Supervisor etc. Collector is a
A simple role obtained by all employees, while Case Assignor or Supervisor roles are only obtained by specific managers.

New delinquent loan cases and loan payments are downloaded from the account management system on a daily basis. When a loan case is assigned to a specific collector, it appears in his/her daily diary. Performing loan collection actions results in the modification of the loan status. Reminder data for the loan case are also maintained. There is a predefined set of actions initiated by collectors, some of which are:

- Client contact without success
- Client contact with success, not payment scheduling
- Client contact with success and payment scheduling
- Sending a letter to the client
- Defining an auction
- Defining a new payment settlement

The loan case is handled by LMD until delinquent loan instalments are paid. The generic MPN model depicting delinquent loan collection is depicted in figure 1.

At the more detailed level, collection actions are distinguished into three categories: simple actions completed by the collector, actions requiring the approval of his/her supervisor and actions for which sending a letter is necessary. Determining the proper action is based partially on LMD policy and partially on the collector’s strategy and experience, the sequence of actions, thus, is not predetermined. This is depicted in the MPN model using a control place, as indicated in figure 2. Green places indicate control. Control places represent control flow. A control place has more than one outputs, while the decision concerning which output will be activated may depend on deterministic or random rules (e.g. employee decision). Control places are particularly useful when describing semi-constructed or dynamic BPs.

Loan collection process consists of two main activities: **Collection Action** and **Loan Case Removal**. The first one is further analysed by another MNP, while the second one is a simple task. **Loan Case Removal** has no performer place as input and it is performed automatically. For **Collection Action** activity, which has such an input, the symbol “R” within the place indicates that a role has been defined as the performer, more specifically the **Collector Role**. Guards specify transition constraints. The **Collection Action** activity may be initiated when the collector responsible for the specific loan case is available (A.Responsible=B) and a reminder is set in his/her daily diary (A.Reminding_Date=@Today). Places are numbered using capital letters or numbers to simplify guard definition.

The in-depth description of the **Collection Action** activity is beyond the purpose of this paper. However, one should note some points of interest. Input and output places of the MPN must be the same as the input and output places of **Collection Action** activity presented in figure 1.

Although other resources, such as **Letter**, are produced during this activity, they are not indicated as output. In the MPN model presented in figure 2, the **Supervisor role** is
also depicted as a performer. This is allowed, as the Supervisor role also includes Collector role. As indicated in Action Approval activity, guards are not static and enable the representation of BPs with dynamic evolution. Although it is not mandatory, scripts can be defined to describe activity functionality (scripts are depicted under activities). Scripts are useful especially when describing tasks, as they are automatically transformed into code within the workflow environment.

Using the MPN model presented in figure 2, one can simulate the execution of the business processes and reach conclusions concerning the behaviour exhibited by specific collectors, as well as the effectiveness of various strategies and the indication of potential inefficiencies and bottlenecks.

The MPN enabled the formal description of all business processes related with loan monitoring. Gathering and evaluating information concerning the department operation and policies in a systematic way was one of the major contributions. Business process analysis proved to be time consuming due to the complexity encountered in gathering collectors experience. The description of all business processes within MPN and the direct mapping of BP models stored in MPN repository within Lotus Notes facilitated the accurate workflow implementation and reduced development time.

5. CONCLUSIONS

In order to efficiently automate business activities, one should first fully model their functionality. There are many methods facilitating BP modelling, such as IDEF0 and RADs, which provide standard output that can be imported within most popular workflow implementation environments. Unfortunately, they can not be used dynamic activities, since they only depict activities consisting of well-defined steps executed in a predefined sequence. Petri-net models can provide better solutions for this kind of processes.

Business process automation provides significant results only if all activities, independent of their nature, are fully described in a formal way. The extended MPN model, presented in this paper, facilitates the description of dynamic activities using multiple levels of abstraction in a concise manner. It allows the description of process by defining all activities and subactivities, their input and output and rules for their activation. The control place concept allows for the description of ad-hoc activities. Guards are used to depict the conditional invocation of activities, facilitating the provision of a general model properly initiated.

REFERENCES


