LAYERED PROCESS MODELS

Analysis and Implementation (using MDA Principles)

Samia Oussena
Thames Valley University, London, U.K.

Balbir S. Barn
Middlesex University, London, U.K.

Keywords: Model-Driven Architecture, Service-Oriented architecture SOA, Business Process modelling notation BPMN.

Abstract: One of the key challenges of Service-oriented architecture (SOA) is to build applications, services and processes that truly meet business requirements. Model-Driven Architecture (MDA) promotes the creation of models and code through model transformation. We argue in this paper that the same principle can be used to drive the development of SOA applications, using a Business Process Modelling (BPM) approach, supported by Business Process Modelling Notation (BPMN). We present an approach that allows the SOA application to be aligned with the business requirements, by offering guidelines for a systematic transformation of a business process model from requirements analysis into a working implementation.

1 INTRODUCTION

The challenges of developing distributed enterprise systems such as large development teams, complex technology and changing business requirements have the potential of being addressed by the convergence of three key areas of technical developments: Service oriented architecture; Model driven architecture and Business process management. Soley and Watson (2008) raise the same point at a key note address at the 19th conference on Australian Conference on Software Engineering where they articulated the role of MDA as a bridge between SOA and BPM.

Service oriented architecture (SOA) has gained much attention as a unifying technical architecture and has been concretely realised with web service technologies (Erl, 2005). SOA promotes the building, deploying and integrating of services independently of any specific application and the computing platform on which they run. Such services generally observe characteristics such as being individually useful or can composed to provide higher level services thus promoting re-usability. Services also communicate with each other or clients via the exchange of messages and may be part of a workflow application within the organization. As well as a technological imperative, SOA demands a wider organizational re-think to understand what impact such an architectural approach requires on solutions and their design, what it means to assemble them from composable services and how applications are deployed and managed.

Modelling in general is viewed as a capstone of many software engineering approaches where it is used to as an approach to user requirements definition and as a basis for developing information systems to meet those requirements (Moody 2005). Models provide a vehicle for explaining and sharing understanding of complex problems and provide capabilities for different views of the underlying problem at different levels of abstraction. Model driven architecture takes this premise further by providing an overarching conceptual structure for using and applying transformations to models in a structured and controlled manner in all stages of the software engineering development process. The Object Management Group (OMG) provides a set of standards to express models and model-model transformation and has been leading industry

Oussena S. and S. Barn B. (2009).
LAYERED PROCESS MODELS - Analysis and Implementation (using MDA Principles).
In *Proceedings of the 11th International Conference on Enterprise Information Systems - Information Systems Analysis and Specification*, pages 168-175
DOI: 10.5220/0001995601680175

initiatives in the promotion of technologies, methods and standards under the banner of model driven architecture (MDA). There are three key guiding principles for MDA which are relevant to this paper: Models expressed in a well defined language (notation and semantics) are critical for understanding system requirements and solutions. Systems can be organised and built around a coordinated set of models and transformations between models. Models and their modelling languages can be formally described using metamodels to enable model interchange, integration and automation by tools.

Separately but connected to the developments in SOA and MDA, business process modelling is experiencing a resurgence of activity. A new wave of business process modelling initiatives (Smith and Fingar, 2003) involves the return to business process modelling where organizations will rapidly adapt a business process to address a new need, measure the performance of the new business process and then make further changes to the business process to optimize the performance of the process – Business Process Management (BPM). In order to make this happen, important technologies needed to be in place. These technologies enable processes to be designed, implemented executed and evaluated (from a performance perspective) and then changed in real time on business management servers. A key aspect is thus the notion of a model (discussed above), that is, the process definition (and thereby its documentation) and its execution specification. BPM technical infrastructure includes: standards for notations for business process modelling, standards for translation of models into executable languages and the building of systems from multiple process definitions (Delphi 2003).

The last two years have seen the OMG – authors of the Unified Modelling Language (UML) – publish the Business Process Modelling Notation BPMN standard (BPMN, 2006) which includes mappings to the execution language Business Process Execution Language for Web Services. BPEL4WS. At the same time, there has been the emergence of a new form of toolset – BPM tools that are model based and allow a process to be taken from analysis through to execution in a single environment. Therefore, BPMN with a supporting toolset provides a bridge between business-oriented process modelling notation and an IT-oriented execution language. It allows the modelling to encompass both system and human activities. It provides the notation for the modelling of the business flow and web service interaction.

2 RELATED WORK

A comprehensive literature review is outside the scope of this paper instead our primary focus on the work that links the three areas of SOA, MDA and BPM in some way.

Zhang and Jiang have taken an MDA approach to workflow based applications (Zhang and Jiang, 2008). Like the approach described in this paper they distinguish between models for analysis requirements and those models for implementation. Their target implementation models are based on the use of BPEL while their analysis models utilise a specially developed UML 2.0 profile for business modelling. They provide transformation mappings from their business models to BPEL. A key difference between our two approaches is that while we share a conceptual framework based on MDA principles we place greater importance on the role of a standard such as BPMN and transformations pertaining to that standard. Murzek at al (2006) present a categorisation of different model transformation scenarios which include model integration, model translation and model synchronization. In contrast to the algorithmic approach adopted in this paper, workflow patterns at the domain level are used to support the transformation types. Brown et al describe a software development approach to application design using service oriented architecture principles coupled with MDA tool support (Brown et al 2006). They emphasise the importance of utilising such technologies in the context of business architecture (process driven) as response to more recent criticism that MDA has failed to deliver. Huang and Fan propose an approach to enterprise integration where MDA is the philosophy of system development and SOA is the infrastructure of system implementation (Huang and Fan, 2007). This approach shares some similarity with the approach described here but a missing component is the role of BPM as a bridging mechanism between SOA and MDA. Thomas and Leyking present a rationalisation of the importance of business process management and how process models can be used to design and implement service oriented architectures (Thomas and Leyking 2008). Their approach provides a discussion of event process chains, BPMN and BPEL. Their approach however is not contextualised in a MDA framework. Other relevant work is Zdun et al's discussion on modelling process driven service oriented architectures (Zadun et al, 2007). As part of their problem statement they comment:

"Process-driven SOA models are hard to understand and be

kept consistent because many different kinds of models are relevant for a SOA and only loosely interconnected.

Model-driven development of process-driven SOAs is not yet well supported because there are no precise modelling approaches and model-driven development tools that are focusing on the domain of process-driven SOA in general." (page 4: Zdun et al 2007)

This echoes the challenges identified and partially addressed in this paper. The BPMN area raises issues specific to that domain. Research has largely focused on the notation and its efficacy. In particular the expressiveness of BPMN was subject to an examination by Recker et al (2006).The notation set was reviewed from perspectives of construct deficits (lack of notation support for modelling requirement, construct excesses (notations which may not be needed) and construct overloads (notation for which there are multiple meanings). While the research indicated that not all theoretical problems were seen as critical, some issues identified by the research were a lack of business rules, a lack of a structuring capability, and ambiguity around the definitions of Lane and Pool constructs.

In summary, the convergence of SOA, MDA and BPM presents an opportunity for organizations to develop an IT strategy that will allow them to respond to changing business requirements in a structured and coherent manner. However, the literature would indicate that there a number of key challenges that must be addressed.

2.1 Motivation, Research Question and Contribution of this Paper

Historically, business process models have been developed by less IT-oriented, more business-focussed people, and have been technically separated from the processes representations required by the implementation and the execution of these processes. There has therefore been the need to manually translate the original business process model to the execution model. Such translations are subject to errors due to a lack of continuity in the development process and the lack of a framework for managing model transformation in an organised and systematic fashion. The lack of an end-to-end approach also makes it difficult to understand the evolution of the original process model and traceability of model transformations.

The question that we are trying to address in this paper is thus: **Is it possible to articulate a transformation of the business process model that follows an MDA approach?** Deconstructing further: can we provide a systematic transformation of an analysis process model to an implementation process model?

This paper discusses an approach to providing transformations from the initial analysis process (business-oriented model) to the execution process (implementation-oriented model). We present a method that provides the transition between the tasks of the business processes definition and the operational systems that implement the capability, in turn aligning the development of web services with the business needs. The remainder of this paper is structured as follows: The next section (3) introduces key concepts underpinning the paper. Section 4 presents the research approach and the details of the case study supporting the development work. Section 5 describes the model transformation algorithm that supports the transformation of an analysis process model to an implementation process model. The algorithm is illustrated by examples of transformations from the process models development for case study. Section 6 concludes the paper and outlines further intended work.

3 BACKGROUND CONCEPTS

Here we briefly introduce the main modelling concepts to enable sufficient understanding the remainder of this paper. BPMN is a standard for modelling business process flows and web service interaction. It comprises one diagram a corresponding notation set. The diagram and its modelling language include four core modelling elements: activities, events, gateways and connectors:

- An **activity** is an action that is performed within a business process. Activities can be either an atomic or non-atomic part of a process model. These are depicted in a BPMN diagram as either sub-processes or tasks. Activities may be performed once only, or can have internally defined loops. A task is an atomic activity that is included within a process, whereas a sub-process enables hierarchical process development. At the implementation level, it is used more in the context of having different scope – access rights – to the process variables. Sub-processes represent compound activities within the process.
- An **Event** is something that "happens" during the course of a business process. These Events affect the flow of the Process and usually have a trigger or a result. They can start, interrupt, or

end the flow. There are multiple types of events for starting, interrupting and end events.

- The flow of control between each activity is depicted by sequence flow. Business decisions and branching of flows are modelled using gateways. **Gateways** are the modelling elements that are used to control how sequence flows interact as they converge and diverge within a process.
- **Connectors** are of three kinds:
  - A **Sequence Flow** is used to show the order that activities will be performed in a Process
  - A **Message Flow** is used to show the flow of messages between two entities that are prepared to send and receive them
  - An **Association** is used to associate data, information and artefacts with flow objects

This notation is simple enough to be used by business analysts in dialogue with users and rich enough to support programming level semantics. The challenge is to provide a mechanism for a controlled transformation from one use to another.

3.1 Model Views

Before looking at the transformation, let's look at each of the views. The objective of the business view (analysis process model) is to understand the "as is" or "to be" process at a high level. Thus we are concerned with: the ordering of tasks (flow), the roles responsible for the tasks and the objects affected by the tasks. At this stage, we are concerned with producing models that are manageable and readable by primarily business analysts. How a process may be translated into an executable model is not of concern. Such a model is tagged in MDA as a Computation Independent Model (CIM) and a Platform Independent Model. (PIM)

The implementation business process model extends the business analysis model with additional focus on the detail of the use of services and user interactions for the process. The model may represent a process that is composed of a set of external services (web services, user tasks, or other processes) and specify the exact order of activities and input and output messages and their data formats. Issues of parallelism, call-backs are also addressed. The main interests at this level are: Describing the logic of business processes through composition of services; the services could be: web service, user task or other processes; Invoking web service operations; Waiting for the client (the service) to invoke the process through sending messages (receiving a request); Manipulating data variables; Combining basic activities by using structures such as (sequence, branches and loops); Structuring processes into several scopes or sub processes; Handling message-related and time related event and creating compensation activities in case of failure.

4 APPROACH TAKEN

The work described in this paper refers to and follows from earlier work on process modelling, service oriented architecture and associated methodologies. Thus the research approach follows a conjunction of a number of research methods. Firstly the approach draws upon principles of action research, in that we are attempting to integrate theory and practice by a process of experimentation, reflection and iteration (Lau, 1997). The second research method adopted is that of case study research as there are several examples in IS research providing evidence that case study based methodologies are well suited for exploring business processes in an organizational setting (Huang et al, 2005). We also felt it was important to provide experimental data that was derived from the implementation of the case study business process using a BPMN toolset. We chose to implement our case study process using the Intalio Designer Community Edition (Intalio, 2008). This version provides full expressive model based capability to produce BPMN diagrams which can be executed provided enough model based information is entered. The toolset includes a server for executing workflows of a specified process. A full description of the toolset is outside the scope of this paper.

The work described in this paper is derived from the Cova project [http://cova.tvu.ac.uk/] which aims at evaluating the use of BPMN in a Higher Education (HE) context. This project built on research from a previous project: COVARM [http://covarm.tvu.ac.uk/].

The objective of COVARM was to develop a reference model for course validation in HE (Barn et al, 2006) – that is the design of a new course where the primary output of the process is a course specification – a description of a new course.

The COVARM project took a business process based approach to analysing the problem space and developed four case studies of the course validation processes in different institutions. These processes were synthesized into a single canonical reference process which was then used in subsequent development. This process required both computer and human interaction. COVARM went on to develop two technical scenarios for the canonical process:

1. The preparation of a proposal for a new programme and;
2. The organising of a validation event.

The scenarios were implemented as WSDL services and choreographed using the Business Process Execution Language (BPEL). In the COVA project, we modelled and implemented these two scenarios in BPMN using a BPM toolset. We used existing services developed for COVARM, and developed new services when required.

Process design in higher education has relevance to the wider community. The course design process is a core business process and as such it has parallels to product design processes in industry. Further because of the complexity and longevity of the process multiple functional areas (roles) are involved in activities within the process. Issues of transactional completeness, appropriate management of delays are thus similar to that found in industry.

5 RESULTS AND EVALUATION

This section describes the transformation algorithm that maps the analysis process model into a platform specific implementation model. Two examples are used to illustrate aspects of the algorithm.

5.1 The Transformation Approach

The transformation algorithm is applied to the analysis process model. Currently the algorithm is implemented manually. As shown in Fig 1. Each task (BPMN activity) is treated to the "Map_task" algorithm.

Taking the tasks identified in the business view, we apply an iterative process to each task. In overview, the map task function uses the input/output data (based on the modelled data flows) to determine the data type schemas. An attempt to stereotype (classified) each task is carried on. If a task can be stereotyped it can then be mapped into an implementation model task element. Otherwise, it is decomposed into sub tasks or sub processes, depending on the data.

```
FOR EACH task in the analysis business process
{
 /* Apply the mapping algorithm */
 map_task( aTask);
}

Map_task (task)
{
Identify the data input(s) and output(s) for the task;

/* compare the level of granularity of the input and the output data with the data that
may be presented to a user or consumed by a service */

IF (the task can be stereotyped)
 THEN

        /* This is the level of granularity required and it is now possible to move to the
        implementation phase for this task. */
        Stereotype the task
        /* the input data was mapped input schema of either a user task or input
        service and the output is mapped to the output schema */
        Exit;
              ELSE
        {
        /* Decompose the task into sub-tasks */
         List { sub  task } = Decompose (task);
        FOR EACH  sub-task in the list
        {
        /* apply the process */
        Map_task(sub_task);
        }
        }
}
```

Figure 1: Transformation Algorithm.

5.1.1 Identifying Task Data

The identification of the data input(s) and output(s) for a task can be determined by its relation to other tasks within the process. Data flow should already be modelled at an abstract level within the business view. Further analysis of the input(s) and output(s) for the task will inform data type schemas, and the next stage of the process: stereotyping the task. In this activity the analyst will define each input and output either in terms of primitive or complex types. The complex type is then compared to a service message schema or checked if it is at the level that can be presented to a user. If this is not the case, it will then require further decomposition.

5.1.2 Stereotyping Tasks

In UML, a stereotype is used to refine the meaning of a model element or used to describe a model element that can differ in meaning or usage from another element (UML, 2007). Similarly, we use stereotype to categorise a task in a business process model of being of a certain type. We stereotype tasks in a business process as being either:

- User Tasks (tasks requiring human interaction, most usually implemented with forms in a workflow application)
- Service Tasks (tasks that require the invocation of a method on an external service, such as a web service)
- Data Manipulation Tasks (tasks where data is manipulated internally within the process)

Data Manipulation tasks can be treated in the same way as Service Tasks, the difference being that calls to methods internal to the process are used instead of calls to external service methods.

If a task can take the defined input(s) and return the defined output(s) while employing only one instance of one of these stereotypes, the task is at the correct level of granularity for implementation.

It may be the case that a task requires a number of instances of one stereotyped task, or a mixture of stereotypes, in which case the task must be decomposed.

5.1.3 Decomposing Tasks

A task can be decomposed into stereotyped tasks by inspection of data flow requirements. It should be the case that a sub-task (task-within-the-task) is required when data is manipulated, altered or introduced, and when a stereotype is identified.

Once these sub-tasks have been identified, the recursive process starts again with each sub-task. Decomposing a task can results in either: sequence/parallel set of sub tasks; sub tasks included in a sub process or sub tasks included in a loop sub process.

It is desirable that tasks are decomposed into a set of parallel tasks unless there are constraints such as the input from one subtask need to be used by another subtask; in which case the decomposition results in a sequence of subtasks. Sub processes are used when data is required to be isolated or a subtask requires compensation. In case the data is an array of values, the decomposition will be a loop sub process.

5.2 Examples of Implementing a Layered Process Model

In this section we describe two examples of refining the business view of a process model into an implementation view.

The first example is a fragment describing the scheduling of an event once the course documentation and panel members for the validation event have been selected. These additional constraints make this more complex than a usual event scheduling process.

Once a course specification document has been prepared, a request for setting up a validation event is issued. Upon receiving the request, the academic office, will check the completeness of the documentation and decide on whether to proceed with the organisation of the validation event. The fragment of the analysis model and its corresponding implementation model are shown in Fig 2.

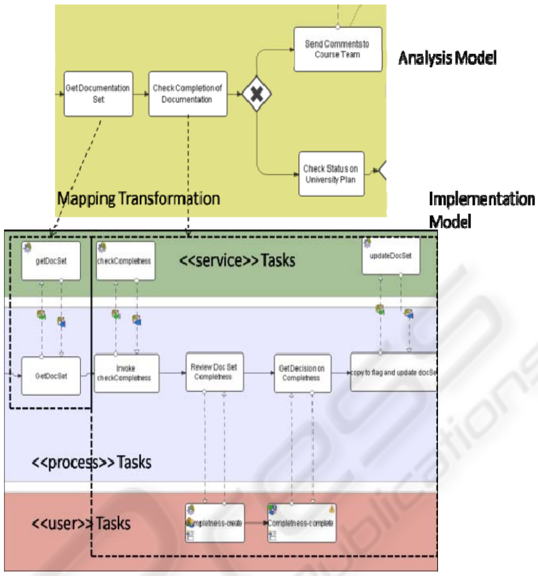

Figure 2: Example 1.

This example illustrates the decomposition of tasks in a sequence of subtasks. The 'GetDocumentSet' task takes a request of retrieving a document set and returns the actual document set. This can directly be mapped into a web service. The "decide on completeness of documentation" task takes a document set as an input and the output is Boolean. Here to arrive to the decision we need to present the document set to the user with the intention of making a decision on the completeness of the document. It is therefore mapped into getting an automatic review of the completeness (service), a viewing of the document and a completeness report and capturing the user decision on completeness. It was also mapped into a task of saving any changes made to the document set, which lead to the invocation of a web service for storing the updated document.

The second example illustrates two other types of decomposition. We first look at the decomposition into parallel subtasks and then into loop sub processes. Here, the academic office tries to coordinate a panel for the review of the course specifications.

Table 1: Examples of transformation.

Analysis Model	Transformation	Implementation Model
Example 1		
Get Documentation Set	Direct mapping	<<**Service Tasks**>> GetDocSet
		<<**Process Tasks**>> GetDocSet
Decide on the Completeness of the Documentation	Mapping into tasks sequence	<<**Service Tasks**>> CheckCompletness UpdateDocSet
		<<**Process Tasks**>> InvokeCheckCompleteness ReviewDocSetCompleteness GetDecisionOnCompleteness CopyFlag&UpdateDocSet
		<<**User Tasks**>> Completeness-Create Completeness-Complete
Example 2		
Get Potential Panel Member List	Mapping into parallel tasks	<<**Service Tasks**>> CreateEvent GetExternal GetInternal
		<<**Process Tasks**>> CreateEvent GetExternal GetInternal
Validate Suitability of Panel	Mapping into sequence tasks	<<**Process Tasks**>> SelectPanelMember RetrieveSelection
		<<**User Tasks**>> ViewPotentialMembers SelectPotentialMembers
Choose Panel Members	Decomposed into Sequence of tasks & 2 parallel subprocesses. Each subprocess decomposed into parallel loop task and a subprocess (illustrated in Figure 4)	<<**Process Tasks**>> GetpossibleDates SuprocessInviteExternal (loop task: inviteExternal & subprocessProcessResp: (GetInvitationResponse, ProcessResponse)) SubprocessInviteInternal (same as the external)
		<<**User Tasks**>> PanelInvite-create PanelInvite-complete
		<<**Service Tasks**>> ProcessMemberResponse

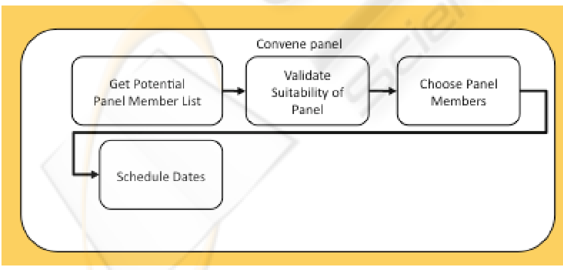

Figure 3: Example 2.

A panel is usually composed of external subject specialists, internal (institutional) staff and a chairperson. The academic office reviews a list of potential panel members, chooses an appropriate number for each role (chair, external, internal), and arranges a date for the meeting.

A portion of the analysis model is shown in fig 3. An examination of the data for the "Get Potential Panel Member List" task revealed that the identification of two types of panel members (external and internal) was required. This lead to the task being mapped into two tasks: "getExternals" and "getInternals". Both tasks could be stereotyped into web service tasks. We noticed that the order of completion of the tasks are not important –getting internal panel members and getting external panel members were not dependent on each other. A new task was also created requesting the event web service to create a new event that will hold the data related to the event.

The "Choose Panel Members" task needs to be mapped into two sub-processes: an "invite external" sub process and an "invite internal" sub process. As illustrated in Fig 4, within each sub process, panel members are to be notified of their invitation to the event. Each individual member is required to receive the invitation at the same time; hence, the task that invites each panel member needs to be in a parallel loop. This process was completed by putting a timer on the response sub process to allow for eventualities such as a non-responding panel member.

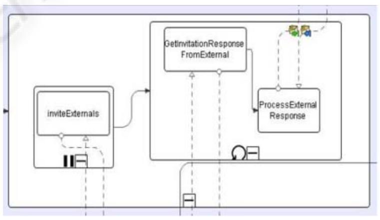

Figure 4: Sub Process Invite External.

6 CONCLUSIONS

In this paper we outlined the case for the application of MDA principles in business process analysis and implementation. We have provided examples that illustrated an algorithm for model transformation from analysis to implementation. These examples have been based on the use of Intalio Designer. We recognise that other toolsets may be used to implement such transformations, but we feel that our approach is toolset-independent. The transformation stages, while straightforward, are sufficient to achieve the consistent level of granularity required to successfully bridge the gap between process analysis and process implementation.

The transformation from a high-level analysis to a lower-level implementation model can involve some degree of complexity, even for seemingly straightforward processes. Key to achieving a successful transformation is the decomposition of these high-level tasks into tasks at an appropriate level of granularity for the implementation of the process. The transformation approach proposed has the potential to be a useful technique for methodologies that need to be developed for BPM. The algorithm presented is implementation neutral and with further experimentation may be sufficiently generalisable and transferable between toolsets such that we feel a conversion from manual transformations to automated wizard-type transformations is a real and achievable possibility using this approach.

REFERENCES

Barn, B.S., Dexter, H., Oussena, S., Petch, J. 2006, An Approach to Creating Reference Models for SOA from Multiple Processes. In IADIS Conference on Applied Computing.

BPMN. 2006, Business Process Modeling Notation (BPMN) Specification [Online] BPMN. Available at: http://www.bpmn.org/Documents/OMG%20Final%20Adopted%20BPMN%201-0%20Spec%2006-02-01.pdf [Accessed Nov 2008].

Brown A, Delbaere M, Eeles P, Johnston S, Weaver R. Realizing service-oriented solutions with the IBM Rational Software Development Platform. IBM Systems Journal [serial online]. September 2005;44(4):727-752.

Delphi Group. 2003, BPM 2003 – Market Milestone Report. White Paper.'

Erl, T. 2005, Service Oriented Architecture – Concepts, Technology and Design, Prentice-Hall, USA

Huang J.C., Newell S., Poulson B., Galliers R.D. 2005, Deriving Value from a Commodity Process: a Case Study of the Strategic Planning and Management of a Call Center. In Proceedings of the Thirteenth European Conference on Information Systems Regensburg, Germany, May 26-28, 2005.

Huang, S. and Fan, Y. 2007. Model Driven and Service Oriented Enterprise Integration---The Method, Framework and Platform. In Proceedings of the Sixth international Conference on Advanced Language Processing and Web information Technology (ALPIT 2007) - Volume 00 (August 01 - 01, 2007). ALPIT. IEEE Computer Society, Washington, DC, 504-509.

Intalio. n.d, Product List, [Online] Intalio. Available at: http://www.intalio.com/products, [Accessed Nov 2008].

Lau, F. 1997. A review on the use of action research in information systems studies. In Proceedings of the IFIP Tc8 WG 8.2 international Conference on information Systems and Qualitative Research (Philadelphia, Pennsylvania, United States). A. S. Lee, J. Liebenau, and J. I. DeGross, Eds. Chapman & Hall Ltd., London, UK, 31-68.

Moody, D. L. (2005) Theoretical and practical issues in evaluating the quality of conceptual models: current state and future directions. *Data & Knowledge Engineering* 55, 243-276.

Murzek, M., Kramler, G., and Michlmayr, E. 2006. Structural Patterns for the Transformation of Business Process Models. In Proceedings of the 10th IEEE on international Enterprise Distributed Object Computing Conference Workshops (October 16 - 20, 2006). EDOCW. IEEE Computer Society, Washington, DC, 18.

Recker, J., Indulska, M., Rosemann, M., Green, P.: (2006) How Good is BPMN Really? Insights from Theory and Practice. In: Ljungberg, J., Andersson, M. (eds.): Proceedings of the 14th European Conference on Information Systems. Association for Information Systems, Goeteborg, Sweden (2006) 1582-1593

Smith H., Finger P., Business Process Management (BPM) the third wave, Meghan-kiffer press

Soley, R. and Watson, A. 2008. Service Oriented Architecture: Making the Leap, Leveraging Model Driven Architecture and Achieving Software Agility with BPM, SOA and MDA®. In Proceedings of the 19th Australian Conference on Software Engineering (Aswec 2008) - Volume 00 (March 26 - 28, 2008). ASWEC. IEEE Computer Society, Washington, DC, 32-34.

Thomas, O., Leyking, K., and Dreifus, F. 2008. Using Process Models for the Design of Service-Oriented Architectures: Methodology and E-Commerce Case Study. In Proceedings of the Proceedings of the 41st Annual Hawaii international Conference on System Sciences (January 07 - 10, 2008). HICSS. IEEE Computer Society, Washington, DC, 109.

UML, 2007, Unified Modeling language, http://www.omg.org/docs/formal/07-11-04.pdf

Zdun, U., Hentrich, C., and Dustdar, S. 2007. Modeling process-driven and service-oriented architectures using patterns and pattern primitives. ACM Trans. Web 1, 3 (Sep. 2007), 14.

Zhang, L. and Jiang, W. 2008. Transforming Business Requirements into BPEL: A MDA-Based Approach to Web Application Development. In Proceedings of the IEEE international Workshop on Semantic Computing and Systems - Volume 00 (July 14 - 15, 2008). WSCS. IEEE Computer Society, Washington, DC, 61-66.