

GoMoKIT: Towards an applicable goal-oriented Business Process Modelling approach for Knowledge-Intensive Tasks -

Daniela Feldkamp, Simon Nikles

University of Applied Sciences Northwestern Switzerland, School of Business,
Riggenbachstr. 16, 4600 Olten, Switzerland
{daniela.feldkamp, simon.nikles}@fhnw.ch

Abstract. The Business Process Execution Language (BPEL) connects the benefit of service-oriented architecture and business process modelling by using standard control constructs to define the interactions between Web Services. The usage of predefined control constructs is efficient for production oriented business processes, but it leads to inflexible process models, which do not reach the challenges of today's enterprises. To cope with increasing changes, uncertainty and unpredictability they have to be flexible and adaptable. We present an approach which combines well defined process models with goal-driven parts to reach less complex process models and more flexibility and adaptivity during runtime.

Keywords: BPEL, Knowledge-intensive task, SOA

1 Introduction

Adaptivity and flexibility are the most important challenges of today's business [1]. Enterprises which meet these challenges are able to cope with increasing changes, uncertainty and unpredictability. Since the early 90s the IT wants to support businesses to reach more flexibility and adaptivity by separating the process logic from business application up to the implementation of workflow management systems, which separates the process flow from the different tasks. But, it was realized quite soon, that workflow management systems are mainly useful for well-structured processes. They lack flexibility in process execution. An alternative approach towards agility takes into account that every business application is based on rules to conduct the business logic. When compliance requirements increased, along with the demand for business flexibility the business rules approach emerged.

Service-oriented Architecture inherently enables flexibility and adaptivity through choreography of services where each service can select and invoke any other web service. The Business Process Execution Language for Web Services (BPEL) supports the specification for coordination and composition of services [2]. Workflows are composed in BPEL using standard control constructs, like switch or

sequence. The use of predefined control constructs can lead to complex process models, which are hard to maintain if all possible variants of knowledge intensive tasks and especially all rare and exceptional cases are respected. When the tasks are mutually depending on each other, this modelling sometimes is even impossible. Like mentioned by Hepp and Roman modelling the process flow has several disadvantages hence a declarative process description is preferred instead [13]. But the use of standard control constructs has the advantage of efficiency in automated workflows.

To achieve more flexibility we provide an approach, which combines the benefit of BPEL and a goal-driven approach used for unstructured process parts. These unstructured process parts are modelled more abstractly at design time whilst providing the necessary conditions to ascertain the flow of these parts during runtime regarding the case. Modelling business processes more abstractly has the benefits of having a good starting point for discussions or supports discovering similar business processes [12]. The abstract parts are modelled using business rules and semantic technologies.

2 Related Works

A number of approaches to compose services exist, which emerged from the workflow and AI planning research community [3]. The workflow-based composition methods are mainly manual. Abstract process models are captured, whose tasks are related to real web services, which can be selected dynamically. There exist several techniques for web service composition based on AI planning, like rule-based planning and the situation calculus. Our approach is to combine approaches of both communities. For the static part, we use a model, which is bound to real web services, additionally we use the approach of OWL-S which provides a web service language, directly related to AI planning[4]. Their preconditions and postconditions specify the state change during execution of services. These preconditions and postconditions are widely used in planning.[5]

In [6] an approach is presented, whose goals are decomposed in several sub goals, which are related to plans of processes. Unfortunately the explicit modelling of goals and their relation to sub goals in a hierarchical way leads to inflexibility. In [7] an approach is represented, where goals are also decomposed in sub goals. A planner helps to organize the sequence of sub goals and assigns services to realize them under given policies or constraints.

In our approach, we combine these goal-driven approaches with the workflow based composition and OWL-S to create static process models and dynamic process models as well. The problem is, that business users are familiar with modelling of processes using common notations as BPMN, but expressing goals and processes using OWL-S is much more difficult and unusual. Our modelling approach proposes the application of SBVR, which is provided by the OMG[8]. The vocabulary of rules is represented as ontology, which can base on existing ontologies, like ontologies for the modelling of commercial and public enterprises provided by the TOVE¹ project.

¹TOVE-Project: <http://www.eil.utoronto.ca/enterprise-modelling/tove/>

The advantage of using ontologies for representing facts and terms lies in higher expressiveness and organization wide common vocabulary.

3 Example

Figure 1 illustrates a simplified example for approving a building application. The process model describes the flow from storing the application data to a simple check whether there are building restrictions conflicting with the application ending with a complex verification including inspection on location, revision by an expert panel, checking of natural, environmental and historical agencies etc.

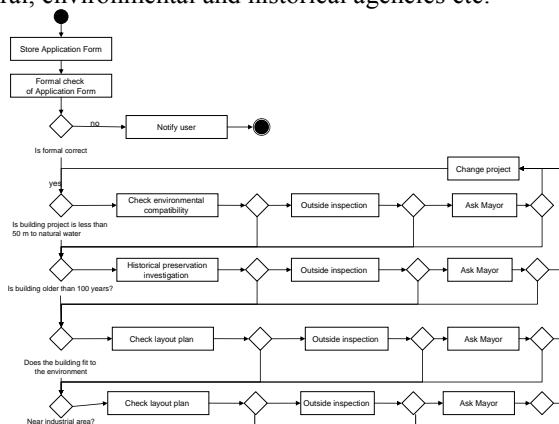


Fig. 1 Example building process

4 GoMoKIT – Approach

For the structured part we use BPEL. As BPEL only provides standard control constructs, which can lead to complex process models, we use a goal driven approach to achieve flexibility during runtime. To avoid the necessity of extending the BPEL standard, we want to implement a web service which provides the additional functionalities for process orchestration and execution during runtime.

Regarding our example shown in figure 2, we replace the complex part of the process using an additional object type, the so-called “variable process” [9]. This object type is related to a pool of tasks. The execution of these tasks is determined at runtime and hence, we avoid modelling them strictly. Additionally the object type is related to a rule set, which combines several rules, which are used to inference knowledge regarding the case.

For instance, a building, which is older than 100 years, is a historical building. Because it is easier for business people to express their business rules using “structured English” we use the Semantics of Business Vocabulary and Rules (SBVR).

The task pool contains several sub tasks. The sub tasks are executed to fulfil a specific goal. Therefore, we explicitly express the goals utilise the SBVR formalism. This goal is decomposed into sub goals which can be fulfilled by the tasks of the task pool. We relate therefore each task of the task pool to a sub goal. We add preconditions which specify what conditions have to be fulfilled before the task can be invoked and postconditions which define what the execution of the task effects. All conditions are expressed also using SBVR.

Figure 2 shows the introduced process modelled with the GoMoKIT approach. The complex part of Figure 1 is replaced with the activity “Check Application” which is expected to achieve the goal “Application is approved“. It is related to a rule-set containing two rules. The first rule expresses, that all buildings must be farther than 50 m from natural water and the second rule expresses, that a historical building is older than 100 years. This rule-set has to be executed first to inference knowledge regarding the current case. After the execution of the rules the tasks of the task pool can be invoked. In this example seven tasks are combined in the task pool. For each task the sub goal, its preconditions and postconditions are defined, such as the task “Historical Preservation Investigation“. After execution of the task the goal “Historical preservation investigation is approved” is fulfilled. The postcondition “Historical preservation investigation is accepted or not” specifies, that after the execution of the task the historical preservation investigation is accepted or denied. The precondition “building is a historical building“ defines, that the task should be invoked only when a building is older than 100 years.

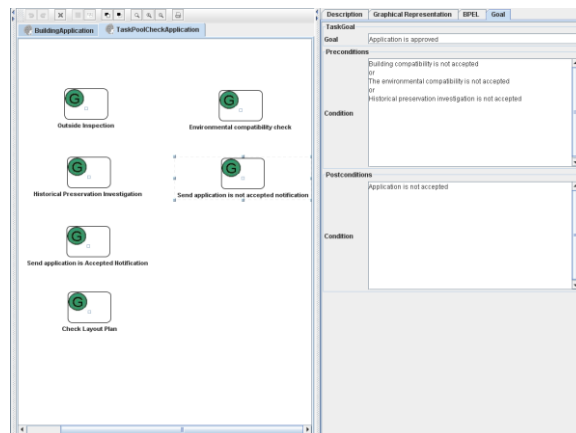


Fig. 2 Task pool belonging to the “Building application” process and defining Subtasks with their goals, preconditions and postconditions.

5 Formalization

For the static part of the process model we use BPEL. The task pool is described using OWL-S. Every task of the task pool is described as an atomic process.

Preconditions, postconditions and goals are expressed using the precondition, effect and result-concepts of OWL-S.

Therefore, the rules, preconditions, goals and postconditions expressed in SBVR have to be transformed to a formal modelling language which can be executed by a rule engine.

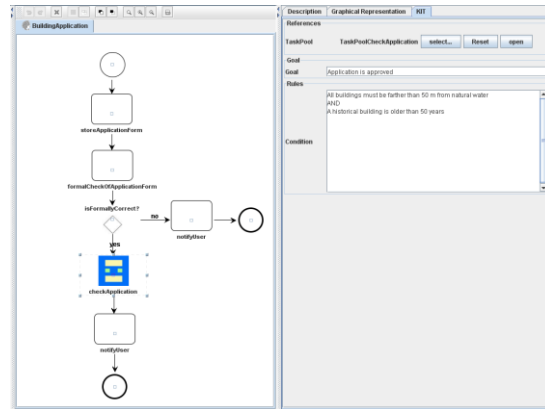


Fig. 3: Example building process with variable process containing a main goal and rules pointing to a task pool

5.2 Runtime approach

To demonstrate our approach we assume that we have a building which is older than 100 years, more than 50 m away from natural water and fits to the surroundings. The first activities (“Storing the application data”, “Formal Check” and the statement) of the process are executed by the BPEL engine. The variable process part relates to a web service, which provides the functionalities for executing rules, planning and executing the sub tasks. This web service gets the rule-set and the OWL-S-file as input. First, the rules are executed. In this use case two rules must be executed and both rules will be fired, regarding our case. The knowledge is stored in a database which is used for the whole process and therefore can be accessed by the variable process.

After the execution of the rules, the service ontology is parsed to get the main goal, which is specified as effect of the OWL-S profile.

All atomic processes are parsed to get these tasks which fulfil the main goal. If no main goal is found, an exception must be thrown to the BPEL-engine. In our use case the main goal is to approve the application, what is fulfilled by two tasks (“Send application is Accepted Notification” and “Send application is not accepted notification”).

The preconditions of the found tasks are used to get the previous tasks, by parsing the postconditions and goals of the other tasks of the task pool. If tasks are found which have no preconditions or whose preconditions can be fulfilled without executing a previous task, these tasks will be executed first. In our case, this applies to

“Check Layout Plan“, because it has no precondition and to “Historical Preservation Investigation”, because the precondition can be met by means of the knowledge stored in the database.

These two tasks are the starting points of the sub process, from which the process is planned.

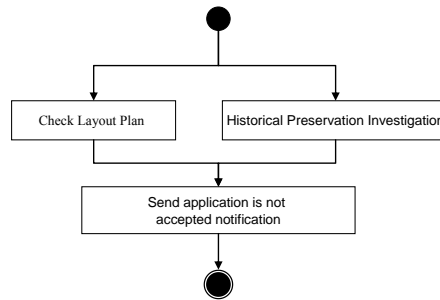


Fig. 3: Plan regarding the case

Because we have modelled the sub process using OWL-S, we have to export them to executable models. The transformation from OWL-S to BPEL is presented in [10].

6 Conclusion

In this paper we have shown how structured and unstructured parts can be combined to one process, to avoid a complex process models by abstracting the complex part using an additional object type. We have represented an approach of combining the static and dynamic parts by using BPEL, semantic technologies and business rules. We have tried to keep the modelling simple, so business users can model the processes without an IT expert.

Moreover, we have shown how this approach can be executed. This approach leads to more flexible and adaptable processes. Further works have to be done to transform the rules, preconditions, postconditions and goals from SBVR to an executable rule language. Additionally, we have to find a strategy for the execution part. If two tasks are executed parallel, we have to make sure, that the two tasks may not overwrite their result. Concerning the modelling user interface, facilities as input checking and support (e.g. selection of attributes) have to be complemented.

7 References

1. Hammer, M. and J. Champy, *Reengineering the Corporation*. 1993, New York. Harper Collins Publishers.
2. Alonso, G., et al., *Web Services, Concepts, Architectures and Applications*. 2004: Springer Verlag.

3. Rao, J. and X. Su. *A Survey of Automated Web Service Composition Methods*. in *First International Workshop, SWSWPC 2004*. 2004. Springer Berlin / Heidelberg.
4. Grosz, B.I.N., et al. *Description Logic Programs: Combining Logic Programs with Description Logics*. in *WWW 2003*. 2003. ACM.
5. Sutton, S.M.J., *Preconditions, Postconditions, and Provisional Execution in Software Processes*. 1995.
6. Greenwood, D. and G. Rimassa. *Autonomic Goal-Oriented Business Process Management*. in *ICAS '07: Proceedings of the Third International Conference on Autonomic and Autonomous Systems*. 2007. IEEE Computer Society.
7. Kaiser, M. and J. Lemcke. *Towards a Framework for Policy-Oriented Enterprise Management*. in *AAAI Spring Symposium, AI Meets Business Rules and Process Management*. 2008. AAAI Press.
8. OMG, *Semantics of Business Vocabulary and Business Rules Specification*. 2006.
9. Feldkamp, D., K. Hinkelmann, and B. Thönssen. *KISS- Knowledge-Intensive Service Support: An Approach for Agile Process Management*. in *Advances In Rule Interchange and Applications, International Symposium RuleML 2007*. 2007. Springer-Verlag.
10. Feldkamp, D. and N. Singh. *Making BPEL Flexible*. in *AAAI Spring Symposium, AI Meets Business Rules and Process Management*. 2008. AAAI Press.
11. Liu, R. and A. Kumar. *An analysis and taxonomy of unstructured workflows*. in *BPM 2005 : business process management 2005*. Springer, Berlin.
12. Dietz, J.L.G., *The deep structure of business processes*, Communications of the ACM, Volume 49 , Issue 5 (May 2006), pp. 58 - 64 <http://portal.acm.org/citation.cfm?id=1125976>.
13. Hepp, M. and Roman, D.: *An Ontology Framework for Semantic Business Process Management*, proceedings of the 8th international conference Wirtschaftsinformatik 2007, February 28 - March 2, 2007, Karlsruhe. In: Oberweis, A; Weinhardt, C.; Gimpel, H.; Koschmider, A.; Pankratius, V.; Schmitzler, B.: *eOrganisation: Service-, Prozess, Market-Engineering*, Vol. 1, Universitätsverlag Karlsruhe, pp. 423-440.