Bridging the Gap between XPDL and Situation Calculus: A Hybrid Approach for Business Process Verification

Bing Li and Junichi Iijima

Graduate School of Decision Science and Technology
Tokyo Institute of Technology, Japan

Abstract. Business Process Verification (BPV) is increasingly important in emerging BPM Systems. Although many approaches related to BPV exist, the gap between formal models defined in previous research and informal models used in the industry prevents these approaches from large-scale industrial applications. This paper proposes a hybrid approach to bridge the gap between XPDL and Situation Calculus. XPDL defines a business process informally while Situation Calculus provides a formal specification to enable the function of BPV. The gap between them will be bridged by a devised language – XSSL, based on which the process model will be logically verified.

1 Introduction

Evolving from traditional workflow management and embracing some new computing concepts, emerging business process management systems (BPMS) are trying to make strength on system integration [1, 2]. As part of the BPMS, the business process verification (BPV) emphasizes the correctness properties of business processes [3]. Despite the increasing importance of BPV in the BPMS, the verification function is still not sufficient [4]. Presently many BPM tools have no explicit or strong verification about process definitions.

Research related to BPV exists, in some of which formal methods are applied to verify business processes and thus a process model can be ensured to some degree of robustness. Due to an emphasis on the underlying mathematical foundation, these approaches are not yet well applied to the BPM industry. The reason resides in the existing gap between the formal models defined in these researches and the informal models usually used in the industry. Explicit connections with industrial standards or transformation rules from them to formal models are necessary in order to bridge or decrease the gap between them.

Flexibility is also an important concern in BPMS. In order to improve the integratability of the BPMS, the industry is making great efforts in standardization. The BPM related research would also benefit from sharing some industrial standards such as business process specification languages. This will enable collaboration among different verification methods and thus reach a higher degree of robustness.

This paper tries to solve the problem in the face of BPMS by concentrating on BPV. A hybrid approach is proposed to verify the XPDL-based process definition.

Li B. and Iijima J. (2007).
Bridging the Gap between XPDL and Situation Calculus: A Hybrid Approach for Business Process Verification.
In *Proceedings of the 5th International Workshop on Modelling, Simulation, Verification and Validation of Enterprise Information Systems*, pages 151-156
DOI: 10.5220/0002416501510156

The industrial standard – XML Process Definition Language (XPDL) – will be used to describe the source process model while Situation Calculus works as the underlying logical language to make verification. For convenient transformation from XPDL to Situation Calculus and improvement in reusability of process or activity specification, XML Situation-calculus Specification Language (XSSL) is originally proposed. The XSSL tries to bridge the gap between the informal and formal process models.

This paper is organized into four sections. Section 2 briefly reviews the related research. Then the BPV approach is explained in detail in Section 3. Finally the conclusion and future work are given in Section 4.

2 Related Work

Business Process Verification (BPV) is to verify the correctness of the business process definition. It usually applies a formal method and specific correctness properties are defined and verified according to the employed formalism. A formal process model does not only make mathematical verification possible, but also enables potential extension for intelligent functions [4, 5]. There exist many formal approaches related to BPV, in which Petri-Net based and logic based approaches are two popular categories.

Petri-Net has been widely used and extended ever since it was invented. The Petri-Net based approaches for business process modelling have some obvious advantages: one is the consistence with the task-oriented view of business processes; a second merit is the graphical notation. These make it popular in the field of workflow modelling. The critical problems in these approaches include the expressive power such as capability of data modelling and in some degree they have been solved by some extension of the classic Petri-Net, such as including hierarchical or object-oriented concepts [6, 7].

Logic-based approaches form another category. Usually mathematical logics are the theoretical foundation of these approaches. By using a specific logic language, business processes are precisely specified. Based on the formal specification, some proof tasks including verification can be performed. The reasoning function distinguishes these logic-based approaches from other ones, which can make the business system more intelligent and dynamic. But since these approaches emphasize the logic notation and calculi, they are not easily understood by the business process analysts. Moreover it takes efforts to formally define business processes [8, 9, 10].

Generally speaking, the present BPV approaches are in need of improvement in order to be applied to the BPM industry. Connections with the emerging industrial standards are necessary and the gap between the industrial and academic process models should be bridged. This paper tries to provide a hybrid approach to integrate such informal and formal process models. This way the separate strengths such as practicability and robustness can both be obtained.

3 BPV Approach

In order to bridge the gap between the informal and formal process models, a hybrid approach for BPV is proposed in this paper. Firstly we will explain this approach from the perspective of model transformation in the desired system.

3.1 Model Transformation

In this approach there are three forms of process models: XPDL, XSSL and Prolog.

XPDL is an industrial standard and supported by many BPMS vendors. In this approach XPDL is selected as the source process model for its analyzability in XML syntax. XPDL focuses on the business logics and can specify the transition relations in business processes. Constituents in a process are separately represented as *WorkflowProcess*, *Activity*, *Transition* and so on [11]. This paper concentrates on the control flow perspective, that is, transitional relations in business processes. But this approach is possibly applied to verify other aspects of business processes such as global constraints [8].

XSSL is designed based on some concepts in Situation Calculus. The direct purpose to design this XML-based language is to bridge the gap between the XPDL defined process model and the logic-based process model. Thus transformation between them is more convenient. Moreover, XSSL will enable separation of activity or primitive process specification from complex processes and improve their reusability. Furthermore, the XML-based syntax of XSSL will enable potential application of Situation Calculus in web service computing such as service choreography.

Prolog works as the implementation language for logical reasoning. It will represent a process model logically by using Situation Calculus. Based on this inferable model, the business process will be formally verified. Benefiting from this formal model, other intelligent analysis will be possibly implemented such as planning or monitoring in BPMS.

Between these process models there are two types of transformation. The first transformation is from XPDL to XSSL. Information about processes, activities, transitions and conditions will be extracted from an XPDL file, and automatically transformed to the compact format in XSSL. The second transformation is from XSSL to Prolog. This will automatically transform an XSSL specified process model to a Prolog program.

3.2 Theories about Transformation

Formal definition about the approach is being written in [13] which concentrates on the underlying formalization of XPDL, XSSL and the mapping between them by employing mathematical structures. In this paper we focus on the overall introduction about the approach.

3.2.1 Situation Calculus

Situation Calculus has strength of reasoning about actions. This formalism can be applied to BPV [9]. The semantic transformation from XPDL to Situation Calculus seems intuitive and uncomplicated. Furthermore Situation Calculus is extensible to include some dynamic features such as concurrency and reactiveness [9, 12].

3.2.2 XSSL

XSSL is initially designed to contain the compact format of action specification in XML syntax. It includes some basic elements such as actions, arguments, preconditions and postconditions. The following is part of an XSSL file that is generated automatically by parsing the XPDL file.

```
<Action Id="check_credit">
    <args>
        <arg>CardNo</arg>
        <arg>Rate</arg>
    </args>
    <preconditions>
        <precondition>
             OrderStatus=="received"
        </precondition>
        <precondition>
             PayWay=="credit"
        </precondition>
    </preconditions>
    <postconditions>
        <postcondition>
             CreditStatus=="valid"
        </postcondition>
        <postcondition>
             OrderStatus=="checked"
        </postcondition>
    </postconditions>
</Action>
```

The defined elements in XSSL correspond intuitively to the concepts in Situation Calculus except for that postconditions refer to successor state axioms. Compared with XPDL, the *Action* element corresponds to an activity in XPDL. The *args* element represents the parameters of an activity. The preconditions and postconditions are transformed from the conditions in XPDL by using some defined rules for different transition types [13].

From XSSL, the Prolog formatted specification about actions involved in a process can be automatically generated. Some additional conditions will be added by restricting the relation between arguments.

3.2.3 Business Process Verification

Based on the formalism of Situation Calculus, the BPV is defined by the following three properties for actions:

- *Executability.* For an action, it is possible to be executed.

- *Satisfiability.* For an action, its postcondition can be satisfied.
- *Effectiveness.* For an action, a satisfiable situation is effective.

Executability ensures that the precondition of an action is satisfied, i.e., the action can be executed; Satisfiability states that the postcondition of an action can be satisfied; Effectiveness ensures that the satisfiable situation is effective, that is, this situation is the successor state resulting from executing the action at the executable situation. Stronger effectiveness is interpreted as that the satisfiable situation is also in the set of possible action sequences that can be extracted from the XPDL routing information.

From the above properties, it can be seen that a business process is obviously verified by its actions (activities). A process consists of a collection of coordinated activities. The pre-and postcondition of each activity in the context of a process also include some constraints from the process structure. After the three properties are verified for each action, the whole process is accordingly ensured to be executed and reach the goal situation.

3.2.4 The Prototype System

A prototype system is designed and developed to demonstrate the feasibility of this BPV approach. The XPDL defined process definition is the input of this BPV system. The XPDL file is automatically transformed to XSSL; and XSSL can be automatically transformed into a Prolog file. Then the user can verify each activity in a process by interacting with the system such as setting initial situation or modifying transition conditions.

3.2.5 Related Issues

Some related issues are necessary to explain and clarify the approach. This approach is concentrating on the control flow perspective of a business process. Transition relations in XPDL are verified with Situation Calculus. But this approach can be extended to other aspects of a business process.

The current verification focuses on the level of a workflow process. But when a complex business process consisting of several workflow processes is considered as a procedure or complex action, the verification can be extended to check complex business processes. Golog is just such a programming language based on Situation Calculus. During the verification, we also use a Golog interpreter [12] and thus this work can be directly extended to complex processes.

Furthermore, the possible transition routes can be extracted from XPDL. By parsing such information, the possible action sequences or situations in Situation Calculus can be obtained. This can be used to aid verification and improve Prolog performance by avoiding some backtracking on situations. This aspect will be explicitly defined in our next work.

4 Conclusions

This paper proposes a hybrid approach for business process verification. The industrial standard XPDL and the formalism of Situation Calculus are selected as the languages for process specification. Although they are usually used in different fields, they share some functions in specifying processes or activities. By linking them to cooperate in business process verification, some meaningful results will be obtained. The practicability and robustness are direct merits. Besides, the target model is inferable and can enable more intelligent functions.

XSSL is devised to bridge the gap between XPDL and Situation Calculus. Some concepts of Situation Calculus are represented in XSSL with the syntax of XML while encapsulating the underlying logic calculus. This attempts to make part of this formalism accessible to general business analysts and furthermore enable potential application in service computing.

There is still much work to be done in our future research and some extension work was explained in Section 3.2.5. Other meaningful work will include the extension of the underlying formalization of XPDL and the mapping from XPDL to XSSL. Furthermore, representation of conditions in XPDL will be also necessary to improve the degree of semantic verification.

References

1. Havey, M.: Essential Business Process Modeling. O'Reilly, 1st edition (2005)
2. Marinescu, D.C.: Internet-Based Workflow Management: Towards a Semantic Web. Wiley-Interscience, New York (2002)
3. Van Der Aalst, W.M.P., Ter Hofsede, A.H.M.: Verification Of Workflow Task Structures: A Petri-Net-Based Approach. Information System, Vol. 25, No. 1, 43–69 (2000)
4. Van Der Aalst, W.M.P., Ter Hofstede, A.H.M., Weske, M.: Business Process Management: A Survey. BPM 2003, LNCS 2678. Springer (2003) 1–12
5. Schumann, J. M.: Automated theorem proving in software engineering. Springer (2001)
6. Reijers, H.A.: Design and Control of Workflow Processes: Business Process Management for the Service Industry. LNCS 2617, Springer (2003)
7. Janssens, G.K., Verelst, J. and Weyn, B.: Techniques for Modelling Workflows and Their Support of Reuse. Business Process Management, LNCS1806, Springer (2000) 1–15
8. Mukherjee, S., Davulcu, H., Kifer, M., Senkul, P. and Yang, G.: Logic Based Approaches to Workflow Modelling and Verification. Logics for Emerging Applications of Databases, Springer (2003)
9. Koubarakisa, M., Plexousakis, D.: A formal framework for business process modelling and design. Information Systems 27 (2002) 299–319
10. Iijima, J.: Formal Approach for Business Process Modeling. Proceedings of National Conference of Japan Society for Management Information (1999) 151–154
11. WfMC: Process Definition Interface – XML Process Definition Language. WFMC-TC-1025, Version 2.0 (2005)
12. Reiter, R.: Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems. MIT Press (2001)
13. Li, B., Iijima, J.: Formal Verification of XPDL-based Business Process Definition. Submitted to the International Journal of Business Process Integration and Management (2007)