Towards Business Level Verification of Cross-Organizational Business Processes

Kioumars Namiri[1], Nenad Stojanovic[2],

[1] SAP Research Center CEC Karlsruhe, SAP AG, Vincenz-Prießnitz-Str.1
76131 Karlsruhe, Germany
Kioumars.Namiri@sap.com

[2]FZI Karlsruhe, Haid-und-Neu-Str. 10-14
76131 Karlsruhe, Germany
nstojano@fzi.de

Abstract. In this paper we present a novel approach for the verification of a business process configuration. The approach is formal, so that logic mechanisms are used in the verification process. The approach has been applied in the scope of the project ATHENA for the verification of cross organizational business processes.

Keywords: Formal verification, Business process modeling, Business configuration, Cross-organizational business processes

1 Introduction

Today most enterprises do not implement their software applications from scratch on their own. They rather decide to buy pre-built IT-Solutions from software vendors, where the software applications are built on top of it. This is especially the case in the area of ERP Software. The software has to be adapted in such a way that the implemented business processes there meet the needs and requirements of the customer enterprises. Usually, same business processes differ from company to company as a result of different and changing business environments. Changing business environment is caused by frequently changing internal local business practices and capabilities of an enterprise, its partner ecosystem and the local legal regulations. That means that customers have to configure the functions in the purchased solutions accordingly. Business configuration is part of every implementation project for customers who have bought an ERP product. Soffer et al. describe in [1] configuration as an alignment process of adapting the enterprise system to the needs of an enterprise. Typically in praxis, technology and application consultants take care of an initial business configuration, and the customer business department is responsible for maintenance.

At the same time, most enterprises focus on specific parts of their business process and depend on partners in a market to perform the additional parts of the process required to achieve a complete end-to-end business process. A common business

paradigm is that of service outsourcing, in which an enterprise focuses on its core business process and has secondary process parts enacted on its behalf by service provider organizations.

In the EU-Project ATHENA [2], these kinds of business processes are called Cross-organizational Business Processes (CBPs) [3], i.e. processes that cross two or more enterprises. The ATHENA project deals with the problem of interoperability between enterprise information systems. Solutions to problems associated with CBPs are the main goal of the project. Support for the semi-automatic modeling and automatic execution of these processes are the focus of study in the different research groups, which investigate the problem at different levels: business, technical and execution.

We state that in real world situations not fully configured ready to run CBPs are delivered by ERP vendors to customer enterprises, since the vendors aim to cover the requirements of their different customers. These requirements are first known when introducing the CBPs at a customer company. By business configuration of CBPs we particularly mean in this paper that the customer company is forced to integrate the set of available external business functionalities provided by partners and internal systems in not configured CBPs in such a way, that it fulfills the enterprises situation. Nowadays, both the interface to provide business functionalities provided by partners and the interface to internal systems of enterprises are exposed mostly as services, even as web services. The task of business level adaptation of CBPs according to the available set of existing web services is a costly and time consuming task in most non trivial cases, since it is dependent on frequently changing business environment of a customer. A rule based approach is required to express the existing business environment of an enterprise in terms of business rules to influent the instantiation and execution of a business process.

One challenging need in this context is to provide a mechanism to ensure that CBPs are configured and upgraded consistently. The vision is a top-down deployment of business level requirements on an enterprise application spanning over different processes and a bottom-up verification of already configured processes making sure they steadily fulfill the business level requirements.

In this paper we address the problem of verification of a configured CBP whether it meets the requirements of an enterprise. On requirements level we focus on the mapping between a process step and the available web services. We will see by an example that there are different mapping variants dependent on each enterprise demands how web services are mapped in the same CBP. We will express these dependencies as declarative rules specific for each enterprise representing its business requirements in this scope. We propose to use semantic technologies based on SWRL [4] for representing the configuration constraints on the CBPs. We will use the reasoning technologies provided in Description Logics [5] to inference the configuration of a CBP according a knowledge-base in terms of SWRL-rules, thus we will be able to verify the current CBP Configuration of each enterprise.

This paper is structured as follows: First we take a look at the related work in this area. Following we introduce the concepts developed in ATHENA in the area of business processes collaboration. In the next chapter we introduce a motivating use case for illustrating the need for verification of CBPs. Based on the introduced

scenario we show a solution approach how to semantically verify the configuration of CBPs. Finally we present the planed research and conclude.

2 Related Work

Business Process Execution Language for Web Services (WSBPEL) [6] aims to provide an XML based language for describing business processes and how web services are composed together. ATHENA provides the tool Maestro, which is used for orchestrating business processes. They can be enacted by Nehmiah, a Process Execution Engine. We argue that both approaches, using Maestro or a BPEL-based description of a process does not solve the problem of different enterprise-dependent process configurations since they can be characterized as static in that sense that once a process is implemented, it runs always in the same way and is not aware of changing business environments.

As stated clearly in [5], "Configuration" can be considered as one successful domain for knowledge-based applications built using Description Logics, which includes application that support the design of complex systems created by combining multiple components. While there is industry-wide accepted related work in the area of DL-based product configuration and verification [7] [8], there is less industry-wide accepted research work done in the area of configuration and verification of process-based software applications.

In [9] an approach is introduced to express configurable EPCs (CEPCs). The semantic of business level requirements on business process, which is matter of frequent change, is here explicitly modeled in CEPCs and not separately outside of the business processes, as required by our approach.

There are several works done in the area of integrating business rules and service compositions [10] [11]. While the business rules there address more the runtime behavior of processes, we concentrate more on the design time of processes and offer a verification mechanism for the processes whether they fulfill a set of predefined SWRL-rules. In [12] a formal framework is provided to represent business requirements and goals of an organization and how they can be operationalized in terms of BPEL-Processes. The verification mechanism in this framework is based on model checking.

Finally in [13] with TOVE-Ontology an extension of First-Order-Logic (Calculus Logic) is provided to verify if business processes in a producing enterprise fulfill ISO9000 Quality Norms. In this ontology concepts are provided to describe business processes of an enterprise, but the ontology does not take collaborative aspect of business processes and web service integration into account.

3 Business Process Concepts and Tools in ATHENA

3.1 CBPs

A concept is developed in ATHENA to classify process types pursuing different goals. Processes are divided in three levels of abstraction: a level suited for business analysts, an intermediate level suited for process analysts, and a level suited for IT-experts. At this last level the processes may be executed by computer systems. Furthermore, ATHENA presents a concept to model cross-organizational processes without having to reveal internal, private information of enterprises. This concept includes three different process types that vary in their degree of providing information about a single enterprise and in their degree of providing information about the whole collaborative process:

- Cross-Organizational Business Process: This process type is intended to explain the whole collaborative process and contains mainly abstract information about the roles the involved enterprises play
- Private Process: This process type is used only internally by an enterprise and contains all information regarded as necessary by internal users
- View Process: This process type hides sensitive information contained in the private process of an enterprise and provides partners with information on how to interact with the enterprise owning this private process.

Based on these concepts, modeling tools were enhanced to support collaborative business processes on each level; the approach of having three different levels of abstraction was implemented, too. Thus it is possible to model processes at the business analyst level and to transform them to the technically detailed BPDM-models used in Maestro. Apart from this, based on formal operators a method was developed to enable horizontal transformation. Thus automatic transformation from view process to private processes and vice versa is supported by the Maestro tool.

To complete the model framework, a modeling procedure was established that identified three possible procedures for the creation of views and CBPs. In a *bottom-up* approach each company starts with the identification of their private processes and the creation of interaction-specific views which then are combined into CBPs. In a *top-down* approach, the partners start identifying a common picture of the interaction in terms of a CBP model. Each partner then creates its views according to the process steps that will be executed. As a last step the partners have to define their private processes. The third scenario ("*middle-out*") is that of one partner starting with its private processes and offering a view process to its partners. The partners can link this process-based interface to their internal processes via view processes. This corresponds to a bottom-up approach for one partner and a top-down for the others.

3.2 Business Processes vs. Services

In ATHENA a Tool called Gabriel is provided to bridge the gap between the business process world in terms of CBPs and the SOA world in terms of web services provided by business partners. Gabriel is responsible for defining and enacting the concept of business process task particularly as web services. Maestro allows a user to model business processes as a set of tasks and dependencies between those tasks. Nehemiah allows the execution of business processes modelled in Maestro; such processes are exported from Maestro to Nehemiah via a business process repository, which therefore provides a link between modelling time and runtime. Because both those tools were originally aimed at simulating business process execution, Maestro and Nehemiah do not focus on modelling and enacting the business action that should be performed for a given task. With the aim of cleanly separating modelling and enactment, Gabriel has two distinct parts. The modelling side allows defining so-called task profiles, which are a set of attributes that describe what action has to be performed for a given task. Such an action can be user interaction or service invocation. Task profiles as well as organisational data are stored in a repository that connects modelling time and runtime. When the process enactment engine (Nehemiah) notifies Gabriel that a task is available, Gabriel looks up the task profile that was associated with the corresponding task model in Maestro, and based on the type of profile, puts the task at the top of some appropriate principal's task list, or causes a web service invocation related to that task.

4 Scenario: Integration of Carrier Web Services in the Shipping Process

Many enterprises with a need for shipping functionality use various online-services and tools provided by companies in the transportation industry to facilitate the shipping process for their customers. For instance United Parcel Service of America (UPS) provides several on-line XML Tools [14] or FedEx offers various APIs for integrating their business functionality into a client shipping application [15]. These Tools can be integrated into the Order-To-Cash-Process of a shipper. Integration software vendors have been addressing this issue by providing Shipping applications, but they are mostly carrier dependent and force shippers to implement a specific business process pattern supported by the solution. The shipping process at a shipper company, as well as the problems in its configurations is described in following subsections.

4.1 Description of collaborative Shipping Process

An Order-to-Cash process at a shipper company is generally constituted by following process steps:

A **Sales Order** contains information on ordered Goods, shipping address etc. and is the starting point for the process. The Sales Order can be changed as long as there

is no subsequent Delivery processing started. Until then all fields are changeable. The **Delivery** is created from a Sales Order. Depending on customer specific criteria (e.g. delivery dates / shipping mode) several Deliveries can be created from one Sales Order. Merging of several Sales Orders into one Delivery is possible. From the Delivery the **Picking and Packing** Information is created (Pick List / Handling Units). During this Phase quantities are confirmed. The information is written back to the Delivery. Depending on the physical availability of goods the Delivery can change several times (e.g. only a partial shipment is possible, thus correction of delivery is needed). From a Delivery a **Shipment** can be entered. Several Deliveries can be aggregated to one Shipment. The Shipment is the foundation for the manifest / shipping instructions. After the goods are shipped, a commercial **invoice** for the customer is created. The shipper will be invoiced by the selected carrier and is therefore out of the scope of the shipping process. During **After Sales**, tasks related to eventual Return Management are done.

The Shipping Process is handled by several services. In this paper we will concentrate on core shipping services provided by a carrier as web services:

- **Generate Routing Code:** The routing code is the carrier specific representation of the route over which a parcel is shipped.
- **Calculate Rate:** Rate calculation takes as input the selected shipment type, the route and some more information on the parcel and calculates the rate for the shipment.
- **Generate Label:** This service creates the label for the parcel in the carrier specific format.
- **Manifest:** In this service the current shipment is added to the manifest for the day and sent to the carrier.

4.2 Shipping Process Configuration Variants

Considering the abstract shipping process previously described, the process is instantiated and executed in such a way that in each process step different internal systems/services or external carrier services are called.

In case of a shipping process realized through a purchased solution at a shipper company, the business configuration of the solution according to the requirements of that enterprise has direct impact on the way, at which process step or activity which service(s) is called. Different variants occur as web services can be called from different process activities of the standard order-to-cash process executed at the shipper. Furthermore, there are particular call dependencies between some of the services. For instance, a routing code is required to calculate the rate and the label can only be printed after the rate has been determined.

To exemplify this situation, three different business situations for three different enterprises are described, which lead to different configurations of the core-shipping services calculate rate, generate routing code and generate label web services of carriers into the same standard order-to-cash-process of each shipper. This is a result of a real world analysis of different shipping companies regarding their interoperability requirements.

- **Process Variant 1:** In this case, the shipping condition selection, routing code calculation and rate calculation are all done during sales order. This is possible if the total weight of the parcels is already known during sales order entry. After the goods have been picked and packed a label is printed and finally the manifest is generated. As the manifest is not generated after each individual shipment this results in adding a new entry to the manifest for the daily manifest.

- **Process Variant 2:** In this scenario the shipper only selects the carrier and the desired shipping condition during sales order entry. Routing code, rate calculation and label generation are performed after the goods have been packed. This is either the case if the shipper does not need rate estimations during sales order entry or if the shipping process starting with picking and packing is run by a different system.

- **Process Variant 3:** If the shipper wants to use the option of combining several sales orders in one shipment the routing code and the rate cannot be calculated until this combination has been done. Thus, the services for calculating the routing code and the rate are called during the delivery step. This is possible if the final weight of the freight is already known during delivery. Label and manifest are still called after packing.

Figure 1 illustrates the 3 different scenarios:

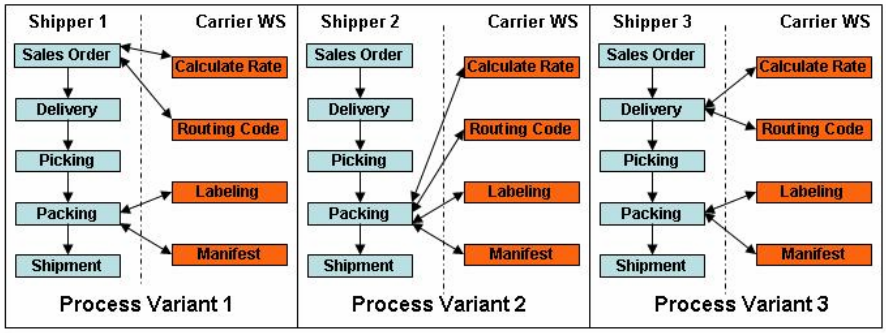

Fig. 1 Different Process Variants in the Scenario

We can see that there are 3 different variants how external web services can be integrated into the same business process. Usually the process is purchased by a customer enterprise (the shipper) and may be preconfigured in one of these variants. In this case only a technical configuration is necessary in terms of registering the technical endpoints of the provided carrier web services in the solution. In other cases the shipping process is not configured at all and must be first configured according to the appropriate process variant reflecting the current business demands of that customer enterprise.

In both cases there is no way to verify if the current configuration in fact satisfies the business level requirements defined by stakeholders of that process, which are often non-technical persons. This is insofar critical, since either the business level requirements may change after the process is already configured or the business process may be reengineered on a technical level. In both cases a mechanism is required to bridge the gap between the technical and business level in terms of to verify whether the technical implementation of the business process always reflects the business level requirements.

It is obvious that in a shipping process not only the carrier web services matter. For example during sales order activity first the stock management systems will be queried to make sure that the ordered goods are already available. If not, the subsequent processes in Supply Relationship Management system will be triggered to order the missing goods. The same applies after the goods are shipped. At this stage the invoicing processes will be triggered to send the customer an invoice according to the applicable customer invoicing process variant for that shipper enterprise. These aspects are out of scope in this work, since we concentrate only on carrier web service integration, but they are rather mentioned here to emphasize that the mentioned problematic pattern in area of different variants on carrier web service integration in a shipping process currently occurs in nearly every other business process in an enterprise.

We should additionally remark that we assume that each shipping enterprise uses one carrier company, thus a dynamic carrier selection from a market-place according to the current shipment conditions is outside of our scope.

5 Formal Verification of CBPs

In this section we present a general approach for the formal verification of a process configuration that can be used in the verification of the CBPs. The approach enables automatic discovery of all parts of a business process that do not satisfy a predefined business requirement. This process we will call inconsistency detection.

We base our work on the semantic description of a business process introduced in [16]. Very briefly, a process is a sequence of activities connected through several types of connectors (join, split, switch). For each activity a set of properties can be defined, like input, output, assigned resources. Finally, for each activity a set of (web) services can be bound.

Although the approach is general, we focus on the type of business requirement mentioned in section 4.2. More precisely, requirements for the service binding of a process can be defined as:

$$M: \text{Property}(A) \rightarrow \text{Property}(WS)^{n}$$

, where
A is the set of activities from a business process
WS is a set of web services

Property(x) is a function that retrieves characteristics of an entity x. A characteristic is defined according to the underlying process model.

Note that the given mapping directly expresses the original constraints mentioned in section 4.2:

$$A \rightarrow WS^{n} \subset Property(A) \rightarrow Property(Ws)^{n}$$

5.1 Formal method for Inconsistency Detection

Verification of process configuration is realized using formal methods. These methods seek to establish a logical proof that a system works correctly, i.e. that it is correctly configured. A formal approach provides:

(1) a modelling language to describe the system;
(2) a specification language to describe the correctness requirements; and
(3) an analysis technique to verify that the system meets its specification.

The model describes the possible behaviours of the system, and the specification describes the desired behaviours of the system. The statement the model P satisfies the specification α is now a logical statement, to be proved or disproved using the analysis technique.

Since the goal of the inconsistency detection is to check whether a service description satisfy the required specification, it can be treated as a formal verification problem in which a modeling language to describe a system is defined through the above mentioned process model, a specification language corresponds to the consistency constraints that must be preserved and an analysis technique can be treated as inference process. In the rest of this section we give more details about last two issues.

5.2 Compliance representation

To formally prove the correctness of a model, the first decision is about what claims to prove. In our case, the claim is that there is no violation of the requirements regarding binding of services. It means that the system for process verification has to return an error value in the case that an activity does not comply to a predefined binding. That can be formally described as follows:

$$isCompliant(X) \leftarrow \neg ErrorBinding(X)$$

$$ErrorBinding(X) \leftarrow Activity(X) \wedge WebService(Y) \wedge Binding(X, Y) \wedge \neg M(Property(X), Property(Y))$$

From the implementation point of view, these constraints can be formally represented as DL-safe rules and KAON2[1] engine is used to evaluate these rules in the process of model verification.

[1] http://kaon2.semanticweb.org/

5.3 An approach for inconsistency detection / model verification

One of the main advantages of the proposed model, in which everything is defined rigorously and precisely, is the possibility to verify the service descriptions formally. In other words, process verification can be done by using formal methods. Formal methods are those that provide a rigorous mathematical guarantee that a large system conforms to a specification. Formal methods can be roughly classified as:

(1) Proof-theoretic: a suitable deductive system is used, and correctness proofs are built using a theorem prover, and

(2) Model-theoretic: a model of the run-time behaviour of the system is built, and this model is checked for the required properties.

In this work we apply the first method, since once we have a service description plus the formally defined consistency constraints we can automatically prove whether these constraints are satisfied in the service description with the help of the reasoning. The KAON2 inference engine is used, since it implements the proof-theory for DL and DL-safe rules. By performing an efficient exploration of the possible inconsistencies that can be built in the service description, the system is able to verify all the consistency constraints defined for the proposed process model.

The set of the consistency constraints as well as a description of the concrete service are inputs to the KAON2 inference engine that is used to automatically verify whether the service description satisfies the consistency. Practically, a trace of the answer to a query is considered as a model that reflects how different pieces of a service description are put together to generate the answer. If the KAON2 verifies that the consistency constrains are fulfilled (i.e. there is no answer), then the service description is consistent. Otherwise, the KAON2 provides explanation about causes of problems, since it can identify the conditions under which the problem occurs.

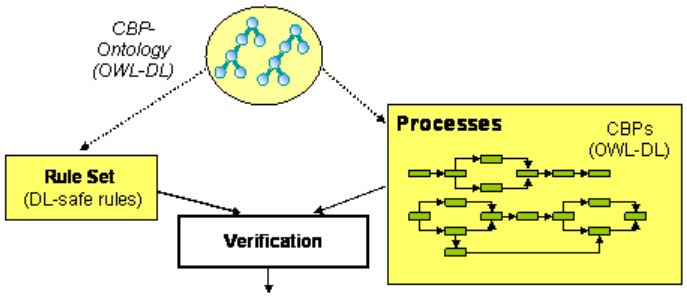

Fig. 2 The illustration of the verification process

Figure 2 summarizes the verification process: The CBP-ontology will be used for the description of the validation rules and the existing processes. The verification module will apply these rules on a concrete process instance in order to determine the validity of the process configuration.

6 Future Research and Conclusion

Changing business environments force software application to be adapted frequently. One challenge is to verify the behavior of a software application if it still reflects the business environments, in which it is executed. In this paper we saw by an example how the business model of a shipper enterprise influences the shipping process composition in terms of the way the carrier web services are integrated into the shipping process. We proposed to use SWRL as a knowledge representation form about the interaction pattern between a shipper and a carrier company. With help of those SWRL-rules the process composition can be verified if it fulfills the required interaction pattern.

By having a formal representation model expressing each enterprise's collaboration configuration for business processes we will be enabled to automatically derive an orchestrated CBP based on that model. A non technical person in an enterprise will be enabled to express and verify the way business processes are composed through available internal or external services if an appropriate user interface is provided to express the rules in an easy way hiding the technical complexity behind those formal models. A very obvious and valuable use case would be then the ability to verify the interoperability configuration of a process with help of reasoning technologies.

As mentioned in the introduction of this paper, one aspect of frequently changing business environments for enterprises is the fulfilling of the growing count of different regulatory compliance requirements. We find nearly in every business area various regulatory requirements, such as Sarbanes Oxley Act, Basel II, HIPAA, ISO 9000, to count a few. One direction of our future work is to research how the content of different regulatory compliance requirements can be captured with semantic technologies and their impact of business processes can be handled more automatically on process-based software applications. Our research will go towards providing a knowledge-based compliance-aware architecture.

Acknowledgment

The research presented in this paper was partially funded by the EC in the projects ATHENA (IST 507312) and SAKE (IST 027128)

References

1. Soffer, P.; Golany, B.; Dori, D.: ERP modeling: a comprehensive approach. Information Systems 28(6) (2003) 673-690
2. ATHENA European Integrated Project, http://www.athena-ip.org/
3. Greiner, U., Lippe, S., Kahl, T., Ziemann, J., Jkel, F.W.: Designing and implementing

crossorganizational business processes - description and application of a modeling framework. In: Interoperability for Enterprise Software and Applications Conference I-ESA. (2006)
4. Horrocks, I., Patel-Schneider, P.F., Boley, H., Tabet, S., Grosof, B., Dean, M.: SWRL: A Semantic Web rule language combining OWL and RuleML, version 0.5 of 19 november 2003, http://www.daml.org/2003/11/swrl/ (2003)
5. Baader, F., Calvanese, D., McGuinnes, D., Nardi, D., and Patel-Schneider, P. The Description Logic Handbook: Theory, Implementation and Applications. Cambridge University Press. (2003)
6. Thatte, S.: Business process execution language for web services version 1.1(2003) (2003)
7. Jon R. Wright, Elia S. Weixelbaum, Gregg T. Vesonder, Karen E. Brown, Stephen R. Palmer, Jay I. Berman, and Harry H. Moore. A knowledge-based con-
figurator that supports sales, engineering, and manufacturing at AT&T network systems. AI Magazine, 14(3) (1993) 69–80
8. Nestor Rychtyckyj. DLMS: An evaluation of KL-ONE in the automobile industry. In Proc. of the 5th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR'96) (1996) 588–596
9. Rosemann, M. and van der Aalst, W. A Configurable Reference Modelling Language. Information Systems, In Press (2005)
10. Charfi, A. and Mezini, M. Hybrid Web Service Composition: Business Processes Meet Business Rules. In Proceedings of the 2nd International Conference on Service Oriented Computing (2004)
11. Orriens, B., Yang J. and Papazoglou M. P. A Framework for Business Rule Driven Service Composition. In Proceedings of the Fourth International Workshop on Conceptual Modeling Approaches for e-Business Dealing with Business Volatility (2003)
12. R. Kazhamiakin, M. Pistore, M. Roveri. A framework for integrating Business Processes and Business Requirements, 9th International IEEE Enterprise Distributed Object Computing Conference (EDOC) (2004)
13. Kim, Henry M. and Fox, Mark S. "Using Enterprise Reference Models for Automated ISO 9000 Compliance Evaluation", In: Proceedings of 35th Hawaii International Conference on Systems Science (HICSS) (2002)
14. UPS Online Tools, http://www.ups.com/content/us/en/bussol/offering/technology/automated_shipping/online_tools.html, Retrieved January 19, 2006
15. FedEx Ship Manager API, http://fedex.com/us/solutions/fsmapi.html; FedEx Return Manager API, http://fedex.com/us/solutions/netreturnapi.html; Retrieved January 19, 2006
16. Stojanovic L., Abecker A., Apostolou D., Mentzas G., Studer R. The role of semantics in e-government service model verification and evolution, Semantic Web meets eGovernment, AAAI Spring Symposia, AAAI. (2006)