An Approach to Business Process Modeling Emphasizing the Early Design Phases

Sebastian Mauser, Robin Bergenthum, Jörg Desel, Andreas Klett

Department of Applied Computer Science, Catholic University of Eichstätt-Ingolstadt
sebastian.mauser, robin.bergenthum, joerg.desel,
andreas.klett@ku-eichstaett.de

Abstract. This paper proposes an approach to formal business process modeling emphasizing the early design phases. That means, the focus is on gathering requirements of a business process in an informal environment. First, methods to systematically elicit all requirements are discussed. Then, it is suggested to formally model and validate the elicited requirements before integrating them to a formal business process model and verifying the model w.r.t. the formal requirements. The approach is inspired by techniques which have proven successful in the area of software requirements engineering. The key technique is the application of scenarios to bridge the gap between the informal view on the process by practitioners and the formal business process model.

1 Introduction

Business process modeling is an important part of many software development projects [1, 2] because software is often applied in the context of business processes. But the number and variety of purposes, business process models are used for, is growing. Business process modeling and management has attracted increasing attention going beyond software engineering in recent years [3, 4]. Process models are more and more used for pure organizational purposes like mere documentation, process reorganization and optimization, certification, activity-based costing or human resource planing. Business process models are also applied as input to workflow systems to control and monitor the proper execution of work items.

In this context, it is very important that the models are valid, i.e. that they correctly and completely represent the relevant aspects of the underlying real-world business process. However, although the need for valid process models increases, there usually is little effort on guaranteeing validity in practice. Mostly, process models are constructed ad hoc – usually in workshops – without detailed documentation of the different requirements of the users involved. Also in theory, most approaches to business process design and according tools assume validity of models and concentrate on analysis (e.g. soundness tests) and optimization issues. But analysis and optimization of invalid models is useless and decisions based on invalid models or execution of invalid models will cause errors.

We faced the problem of generating valid process models in a recent industrial project with the purchasing department of the AUDI AG. Correct modeling of processes is very important for AUDI, because in the recent years documentation of business processes more and more became a major part of the requirements for a TÜV (short for Technischer Überwachungs-Verein, Technical Inspection Association in English) certification for German automobile manufacturers. The increasing importance of valid business process models caused AUDI to ask for our academic support in modeling. The practical problems in such a large company make the modeling of valid business processes really difficult. The processes are typically inter-divisional such that the processes of the purchasing department have impact on the whole company. They are supported by a heterogeneous system landscape and include many media breaks. Within our project we developed an approach of how to come to valid process models in such a complex setting.

To design valid business process models significant attention should be paid to early phases of business process design, i.e. to the question how to systematically gather information about a business process in an informal environment. This part of business process modeling has not yet been sufficiently considered in the literature. But we claim that similar problems have been tackled in the field of requirements engineering for software systems. Therefore, the core idea of our approach is to adopt findings of requirements engineering to the area of business process modeling, in particular w.r.t. the early phases of the design process.

In requirements engineering, scenario based specifications proved to be a successful starting point. Consequently, our approach also starts with elicitation of scenarios which are single process instances of a business process model. Our approach suggests to then formalize and validate the process instances up to a level of preciseness and completeness such that formal methods can be applied in the follow-up steps of integrating the scenarios to a formal process model, e.g. an Event-Driven Process Chain (EPC) or a work flow Petri net, and of verifying the model w.r.t. the scenarios.

The approach is developed within our industrial project but we claim that in principle it can be applied for business process modeling in general. We want to present a first proposal for a respective modeling approach in this workshop paper. Still, we are collecting further experiences in the ongoing project.

The paper is structured as follows: The following section provides a rough survey on the state-of-the-art of requirements engineering in software development, with a particular focus on scenarios. In the sequel, some of these ideas are adopted to early phases of business process modeling. Section 3 provides a comparative study of literature concerning the early phases of business process modeling. Section 4 describes our modeling approach. It is structured in several subsections, referring to the different phases of our approach. Finally, Section 5 provides some concluding remarks.

2 Software Requirements Engineering: A Short Review

In software engineering, significant attention has been paid to conceptual modeling bridging the gap between informal information about the information system to be implemented and the final implementation. Main approaches have been structured analysis and structured design, developed in the late 1970's, and object-oriented analysis and design, starting in the late 1980's [5]. In the 1990's it was generally accepted that requirements engineering [6] – the elicitation, documentation and validation of requirements of a system - is a fundamental aspect of software development and thus requirements engineering emerged as a field of study in its own right.

Information system analysts discovered that faulty requirements analysis was a major reason for project failure or unsatisfactory software systems and that the costs of errors grows exponentially with progressing time in the development process, see e.g. [7, 6]. Therefore, in many cases improving the quality of requirements by using more structured approaches and formal models to elicit and articulate user and domain requirements is likely to both improve the quality of delivered information systems and reduce the costs of system development.

In early stages of requirements engineering, user oriented specification models are desired to describe the required behavior of a complex system from the user's viewpoint while for implementation of the system, integrated state-based system models are necessary [5]. For intuitive user oriented behavioral specifications, scenarios, firstly introduced by Jacobson's use cases [8], proved to be the key concept. Domain experts know scenarios of a complex system to be modeled better than the system as a whole. Thus, starting with scenarios helps to gather system specifications which are valid, i.e. they faithfully reflect the real system requirements. Important advantages of using scenarios at the beginning of the requirements engineering process include the view of the system from the viewpoint of users, the ease of understanding (by different groups of stakeholders), the possibility to write partial specifications and to incrementally extend specifications, easy abstraction possibilities, short feedback cycles, the possibilities to directly derive test cases, the documentation of user-oriented requirements and the possibility to derive scenarios from log files recorded by information systems [9, 10]. However, scenarios cannot capture the entire desired behavior of a system in a structured fashion [5]. Therefore, the final phases of requirements engineering dealing with implementation, final system design and documentation, analysis, simulation or optimization issues, require an integrated state-based model regarding the complete reactivity of (each component of) the system. Since we are interested in the early phases of system design in this paper, we focus on the scenario view of a system in the following.

In the literature the topic of modeling software systems by means of scenarios has received much attention over the past years, see e.g. [6, 5]. Popular scenario notations include e.g. the ITU standard of Message Sequence Charts, the UML diagrams suitable to model scenarios, namely Sequence Diagrams, Communication Diagrams, Activity Diagrams and Interaction Overview Diagrams, as well as Live Sequence Charts, Scenario Trees, Use Case Trees or Chisel Diagrams [10]. But such diagrams can often hardly be used and understood by typical users [11]. On the other hand, the complexity of natural language specifications of typical users in real world situations can often hardly be handled by developers [12]. Scenario modeling approaches accounting for such problems are presented in [6, 13]. Notable approaches in requirements engineering developed to generally bridge the gap between natural language specifications and the variety of conceptual modeling languages are the KCPM [14, 13] (Klagenfurt conceptual pre-design model) and the information modeling approach of [12].

Having finally modeled the requirements of a system by scenarios, the next challenge is to come from the scenario view of a system to a state-based system model, which is closer to design and implementation. Also for this problem several methodologies have been proposed, see e.g. [15, 10, 6, 13, 5].

3 Requirements Engineering in Business Process Modeling

We in this paper claim that for modeling valid business processes ideas from software requirements engineering are appropriate. But domain specific problems require the application of (partly) new techniques and approaches in the field of business processes. For instance, when modeling a software system the scope is usually clearly focused around this system and it always has to be kept track of implementation issues [6]. On the other hand, a business process model often has a larger scope, including many systems and even crossing organizational boundaries, and it often includes many implementation independent parts such as interactions between humans [4]. Moreover, the focus of software modeling [5, 15, 6] is on components or objects, communication (dependencies) between components and the distinction between inter- and intra-object behavior, while the emphasis of business process modeling [3, 4] is on global activities (where modularity comes into play by appropriate refinement and composition concepts), dependencies through pre- and post-conditions of activities, and resources for activities.

Detailed ideas how to elicit requirements on a business process were for the first time raised in the articles [1, 16], where some initial work on this topic has been done by adapting the KCPM [14] approach. There are several further articles mentioning the suitability of requirements engineering activities for the actual design of business process models, see e.g. [17–19], but they do neither go into details concerning this topic nor do they discuss a copious application of requirements engineering techniques in particular for the first phases of business process modeling.

Apart from the two highlighted articles, so far, only few research focused on the early phases of business process modeling can be found. There are a lot of methodologies for the modeling of business processes, but, as far as we know, they rarely provide elaborate systematic approaches to gather the information necessary for developing a process model. Notable examples of business process modeling procedures covering some aspects concerned with gathering process requirements include the approaches presented in [2, 18, 19] and some of the approaches mentioned in the survey paper [17]. In particular, the ARIS approach [20, 21] is very successful in practice. But also these approaches lack a detailed discussion of problems and concrete methods concerned with this topic. Rather, as it is true for most other modeling approaches, their focus is on problems relevant in later modeling stages such as the discussion of process modeling techniques and how to apply the techniques to capture certain behavior. In this paper, we are interested in the first phases of modeling, namely how to find the behavior that should be modeled in an informal environment. Only if this behavior is correctly elicited, a model implementing the behavior can be expected to faithfully represent the intended business process. Although, except from [1, 16], we found no comprehensive methods in literature dealing with the early stages on the way to come to a valid business process model, the mentioned business process modeling procedures [2, 18, 19, 17, 20, 21] present some valuable ideas on this topic. Moreover, there are several helpful strategies and assisting procedures supporting the elicitation of information about a process, e.g. the papers [22–26] apply the user view of scenarios in the context of business process design, many papers such as [27, 28] discuss how to formally integrate different views on a process, the surveys [3, 4] include some early modeling strategies, and finally there are several user-oriented modeling techniques (see e.g. [29] for some recent trends) such as design principles (top-down, bottom-up and inside-out approaches), ideas for the management of modeling activities (e.g. terminology, conventions, process model governance and ownership), tool support for several modeling activities (see e.g. http://bpmn.org), reference models (best practices), patterns (http://www.workflowpatterns.com) and modeling guidelines (quality factors).

4 Modeling Approach

In this section we present our comprehensive approach to model business processes. The approach is inspired by the concepts of scenario-based requirements engineering, i.e. we suggest focusing on scenarios of a business process before designing an integrated process model. Looking at scenarios to specify the behavior of a business process has similar advantages as for the software engineering domain, in particular user-oriented intuitive modeling is supported. The starting point of our approach is distributed knowledge about a business process in an informal real-life environment. The aim is to first develop a comprehensive formal specification of the business process by scenarios and some other types of requirements artifacts. The single formal artifacts can easily be checked for correctness according to the real-life requirements ensuring a valid specification. Then the artifacts are integrated into a business process model given by some modeling language and the generated process model is verified w.r.t. the artifacts. It is important to mention that integration and verification heavily benefit from having a valid formal specification, because this allows (semi-) automatic generation of a process model from the specification, e.g. by Petri net synthesis [24, 26] or merging procedures [27, 28], and formal verification whether a process model fulfills the specification is possible. Altogether, the construction of complex process models behaving valid according to the requirements is supported.

Besides the influences from the requirements engineering domain mentioned in Section 2, our methodology adopted several ideas from the articles on business process modeling cited in Section 3, in particular, from the highly related work of [1, 16]. But in contrast to [1, 16], our approach focuses on scenarios, it is seen independently from the software engineering domain and it is less technical but more detailed in the concepts of the first modeling stages. In general, the difference to all other process modeling approaches is that the methodology of this paper concentrates on the early modeling phases of gathering all relevant information in an informal environment and of the transition from the informal setting to more and more complex formal models.

Our approach is divided into the five phases elicitation, formalization, validation, integration and verification (see Figure 1) and the additional orthogonal phase of information management. The first three phases are inspired by the three dimensions of requirements engineering and the respective requirements engineering activities suggested in [6].

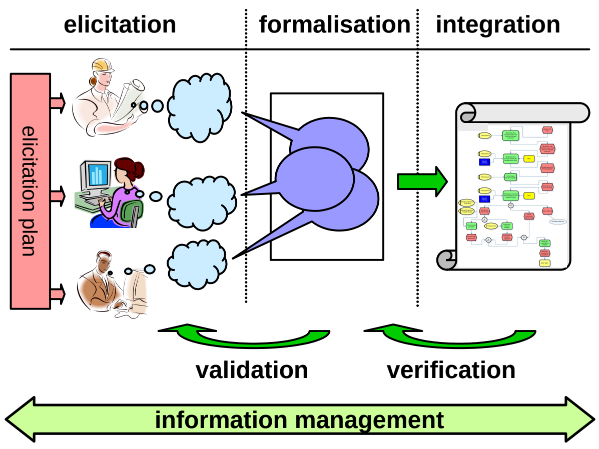
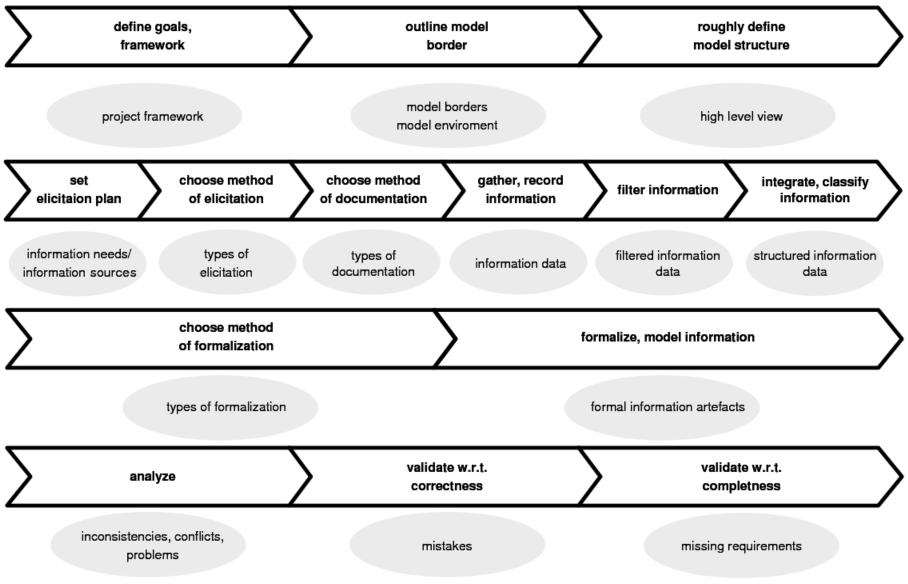


Fig. 1. An approach for business process modeling.

The focus in the elicitation phase is on gathering information. The main problem is to relate and combine the information collected from different information providers in an informal environment into a valid database of knowledge. Therefore, an elicitation plan which determines appropriate strategies for the elicitation procedure has to be assembled. The information collected according to the elicitation plan has to be filtered and documented, yielding a collection of pieces of information. The pieces have to be formalized to information artifacts in the next phase. This enables validation in a follow-up feedback phase. The valid information artifacts document the process requirements which can be integrated into a process model. The process model is finally verified w.r.t. the documented requirements. During all these five phases, it is necessary to manage the progress of information retrieval and to organize all gathered information in the orthogonal phase of information management.

Remark that, while we describe our approach as a sequence of five phases together with one parallel phase, for applying the approach we do not suggest to adhere strictly on the given sequential ordering of phases. On the one hand it is not always possible to generate a process model in one run, such that phases have to be iterated, i.e. it is necessary to repeat phases when information is missing. On the other hand, sometimes it is helpful to move to a next phase, in particular having elicited certain information items, it can be useful to directly formalize and validate them before further proceeding with the elicitation phase.

Next we explain each of these six phases. Thereby, we concentrate on the elicitation, formalization and validation phases and discuss them in more detail. Figure 2 shows all necessary steps of these phases together with the resulting objects. The first two lines refine the elicitation phase, the third line refines the formalization phase and the last line refines the validation phase. The model incorporates ideas from different domains concerned with information retrieval (e.g. [30, 14, 12]).



define goals, framework
outline model border
roughly define model structure
project framework
model borders model enviroment
high level view
set elicitaion plan
choose method of elicitation
choose method of documentation
gather, record information
filter information
integrate, classify information
information needs/ information sources
types of elicitation
types of documentation
information data
filtered information data
structured information data
choose method of formalization
formalize, model information
types of formalization
formal information artefacts
analyze
validate w.r.t. correctness
validate w.r.t. completness
inconsistencies, conflicts, problems
mistakes
missing requirements

Fig. 2. Steps in the elicitation, formalization and validation phase.

4.1 Elicitation

The first three steps at the very beginning of the elicitation phase are the basis of the next six core elicitation steps. When modeling a business process, the starting point is to define scope and aim of the project. It is necessary to set up the project framework which surely influences all decisions made in later steps. Next, the outline of the process has to be defined. This clarifies the border of the process together with the environment and its interfaces. Now, that the business process is set to its context, a first rough structure of the process including aims, related organizational structures and involved documents and systems has to be identified, preferably with the help of a domain expert having a high level view on the process. This helps to get an overview of the information which has to be elicited, i.e. the information needs, and on potential information sources. To actually set up an elicitation plan, it is important to gather more detailed knowledge about available information providers and existing documents describing the process. Such a plan organizes the choice of people or documents which may provide detailed information about parts of the process. For each information to be collected the elicitation plan contains a list of information providers and documents. Gathering information often leads to the following specific problems: the information contains redundancies and repetitions, homonyms and synonyms, exceptional cases to be handled, implicit information or confusions between schema and instance level. To tackle these problems, an adequate elicitation method together with a harmonized documentation method has to be chosen. Since this is an important decision to be made, we will provide a suggestion for both. After these choices have been made, the next step is to gather and record all the information according to the specified elicitation methods.

This normally leads to a large set of loosely arranged information pieces, collected in different kinds of documents, which have to be filtered in the next step of the elicitation process. Filtering corrects all the above listed problems identified in the collected set of information items. The last step integrates and classifies the collected knowledge, i.e. the loose pieces of filtered information are merged and ordered in a structured way according to the chosen documentation method. This concludes the elicitation phase.

Before describing the follow-up formalization phase, we tackle the problem of choosing an appropriate elicitation and documentation method. Often there exist several kinds of documents to be considered, such as working instructions, already existing process models, intra-net information or even theses about parts of the process. Also, exist a lot of methods for elicitation of requirements from the information providers such as interviews, monitoring, logging, rollplays, discussing, questionnaires, meetings, etc [6, 31]. Practical experience suggests to first elicit all adequate documents to get a good overview of the process before starting to consult information providers. Eliciting from information providers needs considerable effort such that a good previous knowledge about the process is desired. After having elicited documents we suggest to interview information providers focused on discussing scenarios. The interviews should be guided by the following framework: After a short round of introduction, the aim of the interview has to be explained to the information provider. This includes level of abstraction, borders, environment and, if already available, interfaces of the process to be modeled. We found it very helpful to shortly introduce the concept of scenarios before the actual interview, because information given by the information provider then was much better structured. Introducing already existing process models to illustrate the sort of the aspired model has similar positive effects. After that, firstly, single scenario instances or even real live examples should be elicited and documented as structured text. Together with the information provider, then a scenario schema is deduced from the scenario instances and documented in a precast scenario form. These forms include entries such as name, description, information provider, activities, events, ordering relation, variations and exceptional cases, pre- and post-conditions, goals and results, success factors and responsibilities. Having filled out such a scenario form, we ask for details about single important activities, events, involved systems, business objects or actors and document them in similar precast forms as well. Although our suggestion here is to focus on scenarios, sometimes it is helpful to discuss whole process fragments which can be done in a similar way. Process fragments are scenarios containing alternatives or loops. Some information providers experienced in modeling may find it easier to describe complex process structures directly in terms of such process fragments. Generally, within the interviews it is important to always mind completeness and clarity of each information recorded and to accomplish the interviews in an appropriate intensity (see e.g. [31]).

The precast forms are already part of our documentation method. We not only use the forms in the interviews, but each type of form defines a template for storing information. As far as possible we insert each gathered information to such a form and port the forms into a database. This yields one table in the database for each type of template. Information not fitting any template is stored in an additional table of the database as structured text together with general specifications such as information provider. Such an organization of information allows to generate different perspectives on the stored requirements through different database queries and search functionalities. It is also possible to automatically produce requirements documents following certain standards.

The main building blocks of a process model are activities, events and different kinds of objects. Therefore, to prepare the requirements for process modeling, we classify the information collected in the tables of the database in a similar way as described in the ARIS [21] methodology. Each information has to be classified into one out of three different views described in Figure 3 (the ARIS methodology suggests the views organization, data, function and one control view relating the first three). Every information about activities or events is stored in a process view. Information about objects is divided into a data and an organizational view. Resource and data objects as well as systems and applications are assigned to the data view and objects concerned with responsibilities and access rights are assigned to the organizational view. Each kind of template naturally belongs to one of these views. Scenarios, activities and events belong to the process view, business objects and systems to the data view and actors to the organizational view. Each information stored in the general purpose table explicitly has to be assigned to a view. Note that in our approach the process view is the dominant one and the relations between the views are naturally included within the process view (instead of a separate control view), e.g. by systems or roles associated to activities (as already given by the templates).

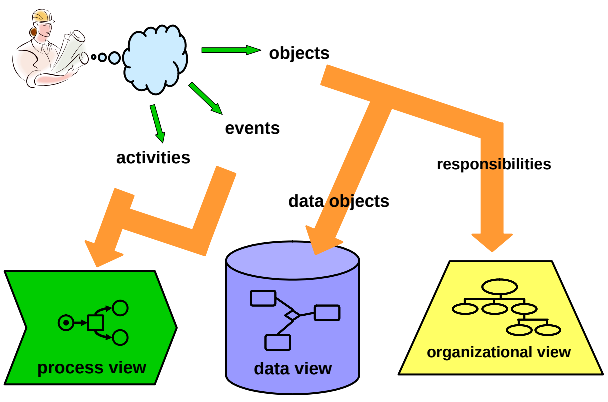


Fig. 3. Three different views to formalize data.

Additionally to the requirements database, we suggest to build a dictionary (similar to an approach described in [14]) to define a consistent language for activities, events and objects used in the documentation. To allow a quick matching of information gathered from different sources, each item in this dictionary is given with a short description (similar to a glossary) and a list of possible synonyms (similar to a thesaurus). In order to navigate through the dictionary we allow to add relations between items yielding a thesaurus-like representation of the domain specific vocabulary of the business process at hand. In particular, a hierarchical ordering of the items is useful to find notions used in the dictionary for certain objects. Similar as in the case of object oriented modeling, it is useful to refine the hierarchy by distinguishing an is-a, a part-of and a property-of relation if possible. Additionally, an association relation to generally link related items is important. Via a graphical representation it is possible to feedback such a dictionary to the information providers. A tool nicely visualizing the relations of the items, having intuitive hide/show functionalities and a search functionality is necessary here. Finally, it is important to link the dictionary to the process information stored in the database, e.g. by adding hyperlinks between the dictionary and respective notions occurring in the entries of the database.

4.2 Formalization

The next two steps depicted in Figure 2 are allocated in the formalization phase. This phase is necessary to get precise requirements of the intended process model. Using informal or semiformal models instead of formal ones for specifying requirements can lead to misunderstandings between author and recipient of a model.

There is a rich variety of different formalisms to choose from. The choice may depend on the documentation method, the target process modeling language or surrounding conditions in the enterprize. Generally, graphical modeling languages should be preferred. The structured pieces of information gathered in the elicitation phase can be translated in an appropriate formal representation. Each formal model of a requirement is called information artifact. The formal models should be linked with the respective documented information.

Our suggestion is to formalize scenarios similar to instance EPCs [32, 22], but allow both events and activities (functions) which not necessarily strictly alternate. That means, it is possible to specify any ordering between activities and events such that any kind of scenario specification can be regarded. In particular, it is possible to consider partial orders of activities called runs or partial orders of events called lifelines. Process fragments (if elicited) can be formalized similarly by adding alternatives to the scenarios. To express process fragments it is also possible to already use the target process modeling language or to simply specify a respective set of scenarios for each fragment. The other way round, having a highly related set of scenarios, it may be helpful to directly fuse them to one process fragment, if that is easy, in order to account for their strong connection. Additionally, we use formalization concepts for pre- and post-conditions, relations between activities or events, invariants or behavioral restrictions (which might be derived from elicited business rules) and triggers in the process view. For the data view we use ER-diagrams and related concepts within the UML, and for the organizational view organigrams or group and role concepts are applied.

4.3 Validation

After some information artifacts have been prepared, their validation is started. This is done in three steps. Before we take a closer look at them, we discuss some basics about validation. Validation of the formalized requirements is a necessary phase in our approach because it is easier to check the requirement artifacts than checking a whole process model. When the validation phase takes place at an early stage of the whole modeling procedure, mistakes recognized at such a stage can be clarified with less effort. As a consequence, validation is a task that also takes place in the first two phases of elicitation and formalization, e.g. the preparative high-level model, the elicitation plan and the filled out precast forms have to be validated (using the same methods applied in our actual validation phase). But the main validation phase of our approach, which we discuss in this subsection, is accomplished using the information artifacts after the phases of elicitation and formalization. This is because at this stage the requirements have to be correct and complete to allow a reasonable further processing in the integration phase. Therefore, a validation-quality-goal which has to be passed by every single information is applied.

The first step of validation, called analysis, solely deals with the documented information not involving any information provider. In this analysis step the unambiguous formal representation of the information artifacts enables to check for inconsistencies (e.g. some precondition never appears as a postcondition), conflicts (only occurs when there are more than one information provider for specific information) and similar problems in the requirements. Concerning conflicts, it is important to identify, analyze and solve them in this early stadium. They are documented because it might be that the same controversial subject reappears later on. Concerning inconsistencies, even automated or semi-automated analysis methods are applied for analyzing formal artifacts, e.g. matching preconditions and postconditions, checking every artifact if it is formalized with a correct syntax or analyzing patterns. Besides examining the formal representations, we also investigate the applied precast forms. These have to be checked to contain no empty fields and that the content of the fields has the right form. Altogether, in this first step of validation we discover problems and lack of clarity within the information artifacts. But the problems and unclear parts can only be resolved with additional information from the information providers. That is the reason why the analysis takes place before the other two steps of validation, namely validation w.r.t. correctness and validation w.r.t. completeness (see Figure 2). Within these two steps, one returns to the information providers, now knowing further points to discuss.

In the step of validation w.r.t. correctness, besides trying to resolve conflicts, inconsistencies and similar problems, it is checked in detail whether the artifacts faithfully model the intended requirements. The main goal of this step is to eliminate mistakes coming from misunderstandings during the elicitation phase and mistakes coming from a faulty transfer from informal requirements to formal artifacts during the formalization phase. Therefore, the information artifacts are discussed with the corresponding information provider, i.e. it is asked if they express exactly what the provider meant. For the discussion, we lean on standard validation techniques from software engineering [6] such as inspections, reviews or walkthroughs. Due to their concreteness and clearness, our main modeling concept of scenarios is very well suited for such discussions. If necessary, the artifacts can be discussed with different information providers (having different perspectives on one topic) or even external specialists, so that the final information artifacts can really be regarded as correct. This part of the validation w.r.t. correctness is performed in collaborative meetings or one-on-one interviews. As assisting techniques for this step of validation we first use perspective based reading (the information provider has to concentrate on a special point of view or role while reading an artifact) to reveal problems [6]. Moreover, we apply automatic approaches for validating formal information artifacts w.r.t. correctness. It is very useful to simulate the given scenarios or to test them towards performance. Even, first prototypes or process fragments are synthesized from the scenarios (and additional artifacts). Automatically analyzing them as well as feedbacking them to information providers often leads to new insights whether the respective input scenarios are correct or not.

The last step in the validation phase is the validation w.r.t. completeness. This step should not be seen independently from the former validation step. That means, often both validation steps take place together, e.g. within the same discussions with information providers, and often the same validation techniques, e.g. discussion techniques, are applied. But, nevertheless, we have distinguished this part of validation from validation w.r.t. correctness, because there is a different focus, namely to test whether the gathered information are not yet complete. We use the following validation techniques specially tailored to find missing information: First, examining existing scenarios, process fragments or prototypes as well as unfolding process fragments or prototypes yields hints or inspirations towards additional scenarios. Second, matching equal states of different scenarios or finding structural dependencies between different scenarios indicate that other combinations of parts of the scenario should be considered. Third, it is important to check if every single context aspect is taken care of, e.g. if all interfaces, stakeholders and objects of the environment are covered by and fit to the specified scenarios. In addition to these three techniques, one also has to simply ask the information providers if they can suggest further providers which might contribute additional information.

4.4 Information Management

Parallel to the five other phases depicted in Figure 1, there is an information management phase, providing the necessary infrastructure for storing, relating and updating all the documents and data, as well as monitoring the progress of the steps described above. Regarding the infrastructure, an appropriate tool management, data management and file system has to be established. Concerning the monitoring of the progress of the activities (note that this task is sometimes also considered a part of validation [6]) a simple to-do list is suggested. In this list every row represents one special information and every column represents the activities to be performed for one single information. The rows can be taken directly from the elicitation plan. The columns should not only contain the main activities elicitation, formalization, validation, integration and verification, but also more precise steps like “make appointment”, “prepare appointment”, “filter information”, and “classify information” are very useful. In particular for the validation phase, a detailed consideration of the progress, even regarding a subdivision of the different applied validation techniques, is necessary. Using such a to-do list concept we suggest to not only keep quantity aspects in mind, since this could mislead to just superficially perform the activities such that they can be marked in the list as done. A respective list has a quality meaning as well. So, when marking a task as done one should check the quality of its execution and of its results. For certain tasks, it is even helpful to supplement the main to-do list by additional detailed checklists [6].

4.5 Integration and Verification

As depicted in Figure 1, the next phases of our modeling approach are the integration of the gathered requirements into a formal process model, e.g. an EPC or a Petri net, and verification of the model w.r.t. the requirements. For the integration, we suggest a semi-automated approach. Given the formal requirements as an input, automatic synthesis methods can suggest model components to a user and automatic test methods can on the fly check the correctness of each component added by hand. That means, the process is designed by a modeling expert, based on the requirements, assisted by a program proposing components to be added and constantly checking for inconsistencies between the designed process model and the requirements. Such a method is faster and less error prone than modeling without algorithmic support. There are also advantages of a semi-automated approach [10] compared to a fully automated approach. Firstly, being involved into the modeling process increases the understanding of the model. This is very important when the model should be extended or revised. Secondly, during the modeling process ambiguous situations can explicitly be solved by the user, and he is explicitly confronted with problems occurring when translating the requirements into the process modeling language.

In the verification phase reliable formal requirements are a basis to apply verification methods which check whether the process model correctly reflects the specified requirements such as testing methods to check the executability of specified scenarios, unfolding methods to check wether the process model has additional non-specified behavior and model checking methods for the verification of formalized business rules. Besides, there are specification independent correctness criteria for process models, such as the absence of deadlocks, certain soundness criteria or structural properties, which also have to be checked by formal methods in the verification phase.

5 Conclusion

The described approach of emphasizing the early design phases of business process modeling together with a consequent documentation and formalization of the information gathered has many benefits. Most important is that as explained throughout the paper the approach heavily supports the generation of valid models in an informal environment. Furthermore, the elicited requirements are documented in such a way that it is possible to get a detailed understanding of them at any time, e.g. if the business process has to be changed or expanded one can build on the existing requirements. Especially, in the case that the business process model is for statutory warranties, its requirements have to be traceable. Moreover, the formal approach enables reliable validation of the information collected and verification and testing of the constructed process model. Formal requirements also support the application of formal methods such as synthesis in the integration phase.

By now, within the AUDI project, we followed our modeling approach for example business processes and found the results most promising. We also developed and applied prototype tools supporting the early phases in our process modeling approach. From the experiences of our project we in particular discovered that in practice there are several difficulties that have to be regarded. There are legal constraints, e.g. it can happen that information providers are not allowed to be stored together with the given information, and organizational constraints, e.g. information providers are often not available. Most important is the factor of time and cost such that a tradeoff between the invested effort made in the early phases of process modeling and the related cost has to be found. Still, as learned from software engineering for important process models we think that emphasizing early design phases always pays off.

The integration and verification phases of our process modeling approach still have to be elaborated. This is the focus of our further theoretical research. In particular, we plan to support the integration phase by adjusting methods known from Petri net synthesis and to develop verification methods by adapting the theories of testing executability of scenarios and calculating unfoldings of Petri nets (algorithms in all these areas are implemented in our toolset VipTool [24, 25], see http://viptool.ku-eichstaett.de). Concerning further practical work, we plan a comprehensive evaluation of the modeling approach going beyond our current industrial project.

References

1. Mayr, H.C., Kop, C., Esberger, D.: Business Process Modeling and Requirements Modeling. In: ICDS 2007, IEEE (2007) 8–14
2. Oestereich, B.: Objektorientierte Geschäftsprozess-Modellierung und modellgetriebene Softwareentwicklung. HMD - Praxis Wirtschaftsinformatik **241** (2005)
3. Aalst, W., Hee, K.: Workflow Management: Models, Methods, and Systems. MIT Press (2002)
4. Weske, M.: Business Process Management – Concepts, Languages and Architectures. Springer (2007)
5. Harel, D., Marelly, R.: Come, Let's Play: Scenario-Based Programming Using LSCs and the Play-Engine. Springer (2003)
6. Pohl, K.: Requirements Engineering. dpunkt (2008)
7. Faulk, S.: Software Requirements: A Tutorial. In: Software Engineering, IEEE (1995) 82–101
8. Jacobson, I.: Object-Oriented Software Engineering: A Use Case Driven Approach. Addison-Wesley (1992)
9. Glinz, M.: Improving the Quality of Requirements with Scenarios. In: Second World Congress on Software Quality, Yokohama (2000) 55–60
10. Amyot, D., Eberlein, A.: An Evaluation of Scenario Notations and Construction Approaches for Telecommunication Systems Development. Telecommunication Systems **24**(1) (2003) 61–94
11. Moody, D.L.: Cognitive load effects on end user understanding of conceptual models: An experimental analysis. In: ADBIS, LNCS 3255, Springer (2004) 129–143
12. Frederiks, P.J.M., van der Weide, T.P.: Information modeling: The process and the required competencies of its participants. Data Knowl. Eng. **58**(1) (2006) 4–20
13. Fliedl, G., Kop, C., Mayr, H.C.: From Textual Scenarios to a Conceptual Schema. Data Knowl. Eng. **55**(1) (2005) 20–37
14. Mayr, H.C., Kop, C.: A User Centered Approach to Requirements Modeling. In: Modellierung 2002, LNI 12, GI (2002) 75–86
15. Liang, H., Dingel, J., Diskin, Z.: A Comparative Survey of Scenario-Based to State-Based Model Synthesis Approaches. In: SCESM 2006, ACM (2006) 5–12

16. Salbrechter, A., Mayr, H.C., Kop, C.: Mapping Pre-Designed Business Process Models to UML. In: IASTED Conf. on Software Engineering and Applications 2004, IASTED/ACTA Press (2004) 400–405
17. Holten, R., Striemer, R., Weske, M.: Vergleich von Anstzen zur Entwicklung von Workflow-Anwendungen. In: Software Management 97. (1997) 258–274
18. Weske, M., Goesmann, T., Holten, R., Striemer, R.: A Reference Model for Workflow Application Development Processes. In: WACC, ACM (1999) 1–10
19. Castela, N., Tribolet, J.M., Guerra, A., Lopes, E.R.: Survey, Analysis and Validation of Information for Business Process Modeling. In: ICEIS, Ciudad Real (2002) 803–806
20. Scheer, A.W.: Architecture of Integrated Information Systems: Foundations of Enterprise-Modeling. Springer (1992)
21. Scheer, A.W., Nüttgens, M.: ARIS Architecture and Reference Models for Business Process Management. In: BPM 2000, LNCS 1806, Springer (2000) 376–389
22. Dongen, B., Aalst, W.: Multi-Phase Process Mining: Aggregating Instance Graphs into EPC's and Petri Nets. In: 2nd Workshop on Applications of Petri Nets to Coordination, Workflow and Business Process Management, Petri Nets 2005, Miami (2005) 35–58
23. Desel, J.: From Human Knowledge to Process Models. In: UNISCON 2008, LNBIP 5, Springer (2008) 84–95
24. Bergenthum, R., Desel, J., Lorenz, R., Mauser, S.: Synthesis of Petri Nets from Scenarios with VipTool. In: Petri Nets 2008, LNCS 5062, Springer (2008) 388–398
25. Desel, J., Juhás, G., Lorenz, R., Neumair, C.: Modelling and Validation with Viptool. In: BPM 2003, LNCS 2678, Springer (2003) 380–389
26. Bergenthum, R., Desel, J., Mauser, S., Lorenz, R.: Construction of Process Models from Example Runs. In: ToPNoC, Springer (to appear in 2009)
27. van Hee, K.M., Sidorova, N., Somers, L.J., Voorhoeve, M.: Consistency in model integration. Data Knowl. Eng. **56**(1) (2006) 4–22
28. Mendling, J., Simon, C.: Business process design by view integration. In: BPM Workshops, LNCS 4103, Springer (2006) 55–64
29. Recker, J.: Process Modeling in the 21st Century. BPTrends **3**(5) (2006) 1–6
30. Bernhard, J., Jodin, D., Hömberg, K., Kuhnt, S., Schürmann, C., Wenzel, S.: Vorgehensmodell zur Informationsgewinnung Prozessschritte und Methodennutzung, Technical Report 06008. Sonderforschungsbereich 559, Modellierung großer Netze in der Logistik, Universität Dortmund (2007)
31. Hömberg, K., Jodin, D., Leppin, M.: Methoden der Informations- und Datenerhebung, Technical Report 04002. Sonderforschungsbereich 559, Modellierung großer Netze in der Logistik, Universität Dortmund (2004)
32. Scheer: IDS Scheer: ARIS Process Performance Manager. http://www.ids-scheer.com.