Goal-oriented Processes with GPMN

Kai Jander, Lars Braubach, Alexander Pokahr, Winfried Lamersdorf
University of Hamburg, Germany
jander, braubach, pokahr@informatik.uni-hamburg.de

Karl-Josef Wack
Daimler Group Research, Germany
karl-josef.wack@daimler.com

Business process management is a challenging task that requires business processes being described, executed, monitored and continuously enhanced. This process management lifecycle requires business as well as IT people working together, whereby the view on business process is quite different on both sides. One important means for bridging the gap between both consists in having a modeling notation that can be easily understood but also has a precise semantics and can be used as a basis for workflow execution. Although existing approaches like BPMN and EPCs aim at being such as notation they are already very activity oriented and do not consider the underlying motivations of processes. Introducing the goal oriented process modeling notation (GPMN) a new language is presented that has the objective of bringing together both sides by establishing higher-level modeling concepts for workflows. This results in an increased intelligibility of workflow descriptions for business people and greater consideration for the way processes are described on the business side. The core idea of the approach consists in introducing different kinds of goals and goal relationships in addition to the established activity-centered behavior model. The applicability of the approach is further illustrated with an example workflow from Daimler AG.

1. Introduction

Business processes form a challenging research area, in which the business and IT sides have to be conceptually and methodically aligned in order to achieve their full potential [8]. The business side typically focuses on the elicitation of business processes, their management and controlling as well as their optimization while the IT side has to deal with their simulation, execution as well as real-time monitoring. Currently, the artifacts produced from methods at the business side are only partially adopted by IT so that a conceptual gap still exists. Few approaches have tried to address business process management (BPM) at a holistic level, which is a prerequisite for effective and efficient BPM. One example for such a holistic approach addressing the whole BPM cycle, ranging from design to execution and finally enhancement, is the ARIS house of business engineering [23], also used at Daimler AG.

In practice, experience has shown that modeling means offered by ARIS, focusing on event process chains (EPCs), are insufficient for some processes at Daimler AG. While processes have been documented with EPCs, they have not been directly adopted by the workflow participants. One major issue is the strong focus on activities and their ordering, found in nearly all modeling languages including BPMN (business process modeling notation) [16]. As the processes considered at Daimler AG can frequently change, the abstractness of the process descriptions is essential for their long-term usefulness.

In this paper, we propose an approach based on the notion of process goals. The use of process goals aims at achieving a higher degree of abstractness in the process models by employing goals to describe what is to be achieved instead of how it should be done. The means for achieving a goal can then be described on a finer grained level using traditional activity oriented workflow languages. In addition, as goals play an important conceptual role on the business side, e.g. in well known process elicitation methods like business score cards [8], policy deployment [8] and with respect to process monitoring and evaluation metrics (key performance indicators, KPIs), they form an ideal basis for enhancing the integration level between both. Thus, it will be shown how the approach fits into the BPM lifecycle and how a developer benefits from using the proposed goal-driven method.

The next section will discuss related work with respect to workflow modeling approaches. Thereafter, in Section 3, the BPM lifecycle will be introduced and the goal-oriented modeling approach and language will be presented in the context of the Go4Flex project and motivated using the lifecycle. The following Section 4 will establish the tools implemented to support the BPM lifecycle throughout all its phases. Section 5 further illustrates the approach by explaining selected concepts using an example workflow from Daimler AG and finally, Section 6 concludes the paper and highlights aspects of current and future work.

2. Related Work

In the literature many different description languages for workflows can be found. They can be coarsely divided into domain centred modeling approaches and execution oriented approaches. Example of the former group are e.g. EPCs [23], BPMN [16] and YAWL (yet another workflow language) [27], and of the latter group especially BPEL (business process execution language) [15], petri nets [18] and also rule based approaches like ECA (event condition action) [12]. The difference between both kinds of approaches is mainly the level of details supported by the language. Execution languages allow specifying more technical details and can thus be used to describe workflows that can be automatically executed. Yet, there is no fixed border between both, because on the one hand some of the more technical languages also offer comprehensible modeling concepts and on the other hand also mechanisms have been devised for converting a modeling language to an executable counterpart, e.g. from BPMN to BPEL [17] or from EPCs to rules [12].

In order to assess the usefulness of these approaches it is necessary to understand the different perspectives that are relevant during workflow modeling. Based on the early work of Curtis et al. [7], List and Korherr [13] proposed a holistic view on business processes consisting of five perspectives. The *functional view* focuses on the concrete execution of tasks and typically considers concepts for describing subprocesses and atomic tasks. By using the related *behavior view* the sequencing of these elements is controlled. In many cases workflow control patterns like sequence, AND- or XOR splits and joins [26] are used for that purpose. The *informational view* is related to the process related data elements that are on the one hand necessary for task processing and on the other also produced by tasks. These data elements represent simple information as well as complex objects and business products or services and are often called process resources. In the *organizational perspective* it is highlighted who processes the tasks and how the distribution of tasks can be handled. Hence, in this view especially concepts like actors, roles and organizational units play an important role. Finally, the *context perspective* adds an overview or meta perspective to the process, which may contain important process characteristics like the process goals and their performance metrics in form of e.g. KPIs.

Please note that the first four perspectives are quite commonly agreed upon and also build the fundamental building block of the ARIS house of business engineering [23]. New is the recent addition of the context perspective, which is until now not adequately reflected and connected to the concepts of the other perspectives. Today, most modeling approaches still focus too much on the functional and behavior perspective, which can e.g. be observed in the BPMN, YAWL and BPEL languages. Organizational aspects are often reduced to the issue of task distribution and for the informational perspective established techniques like the entity relationship model (ERM) are utilized. An exception is the holistic ARIS approach, which pinpoints the right direction but is very heavyweight and offers dozens of modeling diagrams failing to achieve minimality in the sense of the parsimony principle [14] (introduce only as many concepts as really needed). Additionally, at the heart of the approach is the EPC behavior description, which is a fine-grained task and event based modeling approach without goals.

Missing in all aforementioned approaches is the explicit modeling of the process goals and relating them to the context as well as to the behavior view, i.e. goals should be used to understand what a process is used for and also directly steer the process execution. In addition the performance metrics, often using KPIs, form another important brick of the context perspective that currently gains much practical attention through the advent of real-time business activity monitoring (BAM) tools. Therefore, several BPMN tools try to offer solutions for the KPI modeling and process linking but fail altogether to link the KPI results to the underlying process goals, simply because they are not represented. Similar to our research objective is the approach of [20], who use the user requirements notation (URN) together with use case maps (UCM) and the goal-oriented requirements language (GRL) for modeling the context perspective. Also the idea of goal, KPI and process linking has been pushed forward in their approach. One main difference is that we use goals for functional and non-functional process aspects and hence make goals an integral conceptual element for the context and the behavior perspective. i.e. the goal modeling is considered as starting point for the process definitions according to the goal-context method of developed by Daimler AG [5], which has also been implemented in a commercial tool [6]. This tool also includes a goal-oriented language called GO-BPMN. However, Go4Flex differs from GO-BPMN by closely following BDI agent semantics and having a more dynamic approach regarding the goal hierarchy.

3. Concepts

The focus of the Go4Flex project is on complex and dynamic processes, which are currently only poorly supported by the prevailing activity-centered approaches for process description. Studies have shown that people usually require more than the sequential ordering to explain why certain activities are done in a process [4]. E.g. an explanation for performing a “ship goods” activity can be that the payment was successfully processed (activity sequence). Alternative and equally valid explanations include that shipment of goods is required for an order to be completed (decomposition) or that fast track shipment was chosen because the recipient is a premium customer (alternative selection). The various ways for explaining activities are well researched in the area of human practical decision making and have led to theoretical models for explaining human behavior that is based on intuitive common-sense notions like beliefs and intentions [1]. These models already have been successfully transferred into software frameworks for implementing intelligent software agents [19]. The approach of the Go4Flex project is to make use of these existing theoretical models and software technical interpretations from the agent area to facilitate a better treatment of highly complex and adaptive processes.

One core concept of BPM is the BPM lifecycle, which describes the development and refinement methodology for business processes and workflows. A software system which supports such a lifecycle is generally referred to as *Business Process Management (BPM) Suite* or *BPM System*. Many variations of such a lifecycle can be found in literature[25,28,31], however, they all tend to include a design and implementation phase, a deployment and execution phase and a monitoring and analysis phase which feeds back into the design of the workflow model. The generic lifecycle in Figure 1 includes a design phase for planning the process, an implementation phase for generating an executable workflow model, an execution phase in which the workflow is deployed and execution is performed either directly or in a simulation environment, a monitoring phase which overlaps with the execution during which runtime information about the workflow is gathered and finally an analysis phase of the gathered information which provides input for the next design cycle.

Exploiting the full potential of a folk-psychologically oriented process modelling approach requires a unified set of concepts, methods and tools for all lifecycle phases, which together form a full BPM suite capable of supporting a workflow engineer in every step of the cycle. As a result, the Go4Flex approach leads to improvements regarding the management of complex, adaptove processes, that manifest in the different lifecycle phases. E.g. during process elicitation in the analysis phase,

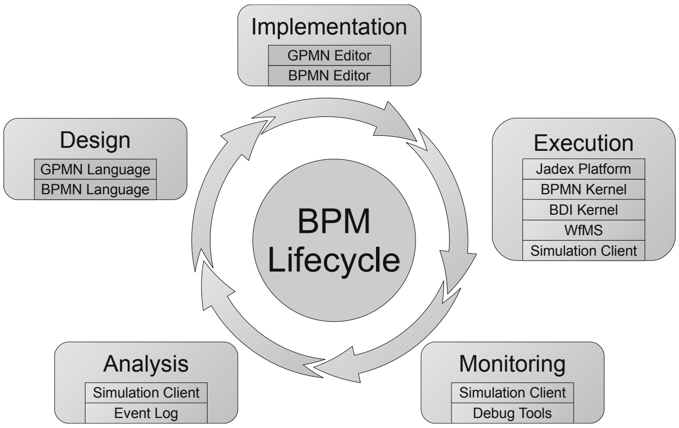

Fig. 1. Go4Flex BPM lifecycle

workshops done with domain experts at Daimler have shown that it is useful to follow a goal-oriented process refinement approach like KAOS [29] by asking “why?” and “how?” questions about process activities instead of focussing on single steps and their interdependencies directly and during process enactment in the execution phase, an adaptive workflow engine can choose from a set of process alternatives based on the dynamic context of a process instance. The following sections will present the vision and current state of the Go4Flex BPM suite by giving basic consideration about process modelling languages, details about rational agent models and introducing the goal oriented modelling notation GPMN.

3.1. *Language Considerations*

Languages are required to express the results of both the design and implementation phase of the BPM lifecycle. Ideally, the language used during the design phase can be used seamlessly during the implementation phase by merely adding additional technical detail. As has been mentioned, a number of workflow languages are already available, however, the focus tends to be on procedural execution. When considering the requirements for a language representing workflows used in Go4Flex, it is helpful to differentiate between the goal level and the plan level. The plan level can the thought of relatively simple and concrete actions which describe plans working towards a particular goal. A procedural approach for their implementation is therefore sufficient to provide the needed expressiveness at the plan level. A large number of established languages supporting a procedural approach are already available, reaching from general purpose programming languages like Java to specialized workflow languages.

Since the area of application is workflow design and implementation, it is advantageous to use an established and recognized language in this field. We have therefore decided to use popular workflow language BPMN as the language for implementing plans on the plan level. However, this is strictly a decision based on practical considerations like familiarity within the community and practical experience with the language in the field and it is not a hard requirement to use BPMN. It is therefore technically conceivable to use also other procedural languages for implementing the plan level.

Unlike the plan level the goal level is not an established concept for workflows yet. However, the concept of goal-plan hierarchies are known from the area of BDI agents. It would therefore be desirable to use a language which makes use of goal-plan hierarchies and can thus also be executed using existing technical infrastructure for BDI agents.

Since a part of the people involved in developing workflows are business experts who may lack technical background, it would be helpful for the language to have a graphical representation similar to the one offered by BPMN. This may facilitate accessibility to non-technical personnel by offering a graphical view on goals and available plans without considering technical details. In addition, it helps technical personnel to quickly acquire a high-level overview of the workflow without spending a considerable amount of time reading source code.

As a result of these considerations we have developed a language called Goal-oriented Process Modeling Notation *(GPMN)* which is used to describe the goal hierarchies in Go4Flex workflows. The language includes both a graphical representation and a text-based representation in XML. The following section will introduce this language in detail and describe both the elements used in the graphical representation of the language as well as their representation in XML.

3.2. *BDI Agent Foundations*

Go4Flex is based on the notion of declarative goals [3,32]. On the one hand goals allow capturing the reason for executing process activities. This facilitates taking a top-down perspective on processes that starts from high-level business goals instead of having to focus on low-level activities. On the other hand goals are an ideal concept for understanding the process context (why something is done and how good it is done). Thus we envision goals as one fundamental conceptual entity for both perspectives. Besides goals, the approach is conceived to integrate well with established concepts, e.g. by reusing available BPM concepts and techniques.

The representation and execution semantics for the goal level has been directly adapted from the notion of goals in mentalistic belief-desire-intention (BDI) agents [32]. In general, a BDI agent in its thinking follows a process called *practical reasoning*, i.e. reasoning that leads towards actions and is not only concerned with knowledge refinement. This process can be understood as two-staged activity consisting of *goal deliberation* and *means-end reasoning*, which an agent continuously performs. In the first phase an agent decides about which goals it wants to pursue and in the latter phase it selects the means, i.e. plans for accomplishing a chosen goal.

As goals are a core concept of practical reasoning it is important to understand the characteristics and nature of goals. As goals are meant to express the motiva-

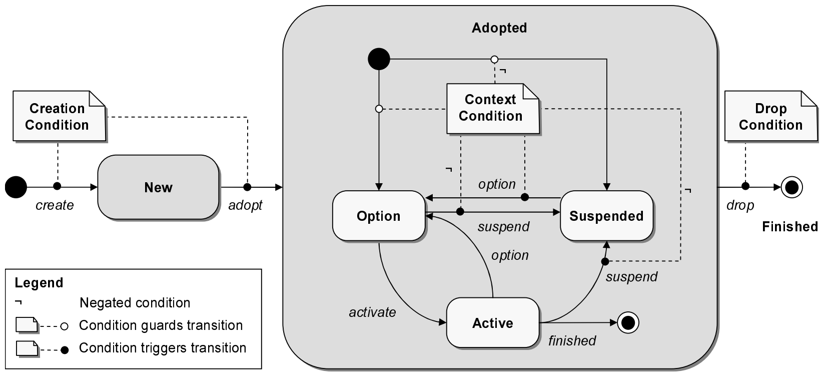

Fig. 2. Basic goal lifecycle

tional attitudes of an actor, be it an agent or a process, it is important to understand them as first-class modeling and execution concepts. Some important characteristics of goals are that they are *persistent*, *producible/terminable, suspendable* and may have *variable durations* [22]2. Persistent means that goals exist over a period of time. Especially, in dynamic environments it is important for an agent to commit to its goals and only give them up for good reasons. Hence, the persistence of goals serves for stability in an agent's behavior [21]. Furthermore, an agent should be able to produce goals when chances occur in the environment and also drop existing ones when it considers them as achieved or possibly unachievable. In situations where an agent cannot pursue a goal any longer for arbitrary reasons it should also be possible to suspend this goal and come back to it at a later point in time. This allows to work on goals in stages and work that has already been done for goal achievement may not be lost as it may be the case if the goal would have been dropped. Also the duration of a goal is of importance for deciding how and when a goal is pursued, e.g. by distinguishing between short term goals and long term strategic goals.

To support all these characteristics of goals a general goal life cycle has been developed [32] (cf. Fig. 2). Basically, this lifecycle distinguishes between goals that are *new*, *adopted* or *dropped*. The transition between new and adopted states means that the agent becomes aware of a goal and may start acting towards its achievement. Most interestingly, the adopted state further divides the set of goals according to the three possible substates *option*, *active* and *suspended* in order to support the goal deliberation and means-end reasoning phases. Thereby, the means-end reasoning of the agent operates on active goals only, i.e. only active goals can lead to the execution of plans and actions. Moreover, suspended goals are goals, which currently cannot be pursued due to an invalid context condition, while options are those goals, which the agent's deliberation mechanism has decided not to pursue (e.g. in favor of other more important goals). Although each goal can only be in exactly one of the substates at any point in time, the state of a goal can change, e.g. when changes happen to beliefs or other goals of an agent. In addition to the states of a goal Fig. 2

also highlights how transitions between states and substates may occur. The general idea is that declarative goals may have goal conditions which may trigger specific state changes. A *creation condition* is responsible for goal inception (new state) and immediately afterwards also goal adoption. Furthermore, a context condition is responsible for monitoring if it is valid for the agent to pursue a given goal. In case the condition gets violated it will trigger a suspension of the goal. If the context is valid again at some later point in time the condition will make the goal an option, i.e. it will not be made active automatically. The reason is that switching between active and option states is the task of the goal deliberation mechanism, which should finally determine the actively pursued goal set of an agent.

Building on the generic characteristics and lifecycle of goals introduced above different functional *goal kinds* have been proposed in order to simplify the description of varying goal semantics. The most important goal kinds are *achieve*, *maintain*, *query* and *perform* goals. These goal kinds map to different application use cases and thus help in naturally modeling the problem at hand. Achievement goals are the most common goal kind, which aim at the establishment of a user defined world state (declaratively expressed as a *target condition*). The goal executes plans until its condition is met or no more plans are available. On the other hand, a maintenance goal can be used to ensure that a specific world state is preserved. In case of a violation the goal tries to re-establish the condition by executing as many plans as needed. A *maintain condition* is used to declare in which cases the world state is considered to be violated and the goal should become active and start plan processing. A query goal is used for information retrieval and only executes plans when the requested piece of information is not already available in the process context. Finally, perform goals are the simplest form of goals, which have a procedural semantics and are directly connected to executing possibly several plans. A detailed description of these goal kinds can be found in [3].

3.3. *Goal-oriented Process Modeling Notation (GPMN)*

The objective of the Goal-oriented Process Modeling Notation is to establish a language to describe BDI-like goal hierarchies which can be used to develop goal-oriented workflows. Since the language is supposed to address both the design and implementation phase of the BPM lifecycle, additional requirements must be met by this language. During the design phase, the business process itself is analyzed and the workflow team attempts to define the business goals of the process. This effort is greatly aided if the language provides a graphical representation, which would also help during the implementation phase to keep an overview of the overall process.

The graphical represention of the elements used in GPMN are shown in Figure 3. It consists of four types of node elements which can be connected using four types of edges. However, since each of the edge types defines a specific relationship between particular node elements, edge connections of a particular type are only

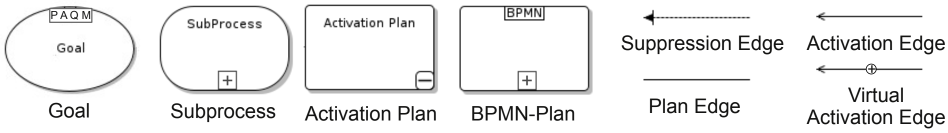

Fig. 3. GPMN language elements

allowed where the connection has well-defined semantics.

When designing a goal-oriented process, the workflow team starts with a general top-level *goal* which defines the business purpose of the process. For example, if the process deals with incoming loan application, a sensible top-level goal for the process would be to finish processing a new loan application. The goal can be of any of the four kinds described in Section 3.2. The abstract top-level goal is then further refined by dividing it into several, more concrete subgoals. The set of subgoals should be complete in the sense that if all the objectives of the subgoals are met, they, together, meet the objective of their parent goal.

The subgoals themselves can again be divided into subgoals as well, continuing the hierarchy to more detailed levels. When a subgoal becomes sufficiently concrete to the point that it can be reached with a relatively simple set of activities, the hierarchy has reached the plan level and the activities can be implemented as *BPMN plans*. If additional flexibility is required, a goal can be provided with multiple plans which are used as options for actions which can be tried to meet the objectives of the goal. BPMN plans are associated with goals using *plan edges*.

Since the decision which of the currently active goals should be pursued should be based on the current circumstances of the process, these circumstances need to be recorded and available. This is accomplished by the *process context* which contains information such as business documents, available resources and information about strategic business decisions regarding the process. Since the context is a global element which is always available to the complete process, it does not require a graphical representation. Instead, the context consists of a table of typed and named properties associated with the process diagram and defined by the workflow engineer. These properties can be set by the workflow with the appropriate information or documents during execution.

Since goals need to be divided into subgoals, there needs to be a way to associate goals with other goals. However, while most subgoals in workflows can be pursued in parallel, there may be dependencies among the subgoals which require them to be pursued sequentially in a certain order. This means that instead of activating all subgoals at once (parallel goal), there is a defined order that requires the activation of the next subgoal only after the previous subgoal has finished (sequential goal).

It is therefore desirable to allow the subgoal activation semantics to be configurable. This is accomplished using *activation plans* which are predefined plans dealing with the activation of subgoals. Like BPMN plans, these plans can be connected with goals using plan edges and provide an option for actions. The activation

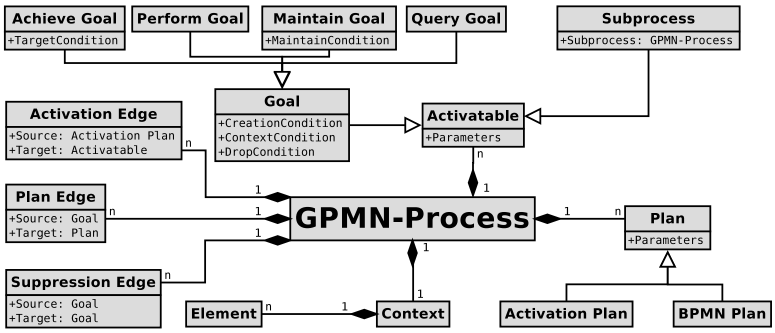

Fig. 4. GPMN meta-model

plan can then be connected with other goals using *activation edges* to indicate which goals the activation plan is adopting as subgoals. The activation plan can then be configured to implement the parallel and sequential activation modes for their parent goal.

Since the activation plan is a technical detail which is only important is certain cases, it can be hidden by replacing the plan edge-activation plan-activation edge pattern with a *virtual activation edge*[11]. This edge directly connects two goals, representing the goal-subgoal relationship. However, this edge is merely a shorthand for the activation plan pattern and is represented as this pattern in the execution model.

In addition, GPMN also supports *suppression edges* as an option to influence the goal reasoning process beyond conditions. If a suppression edge connects one goal with another, the first goal suppresses the other. This means in cases where both goals are adopted, the second goal is put in the option state until the first goal is either complete or dropped.

The final element of GPMN is the subprocess. Subprocesses allow the workflow engineer to decompose a process into multiple diagram. The subprocess can be either external or internal. Internal subprocesses are integrated into the same process instance as their parent process, thus allowing them to share the process context of the parent process. External subprocesses on the other hand are executed as separate processes, allowing for parallel execution but providing each process with a separate context, allowing communication only using messages or parameter passing.

The relationship between the GPMN elements are part of the GPMN meta-model shown in Figure 4. The starting point for the model is the GPMN-Process, which can contain a single context with any number of context elements and any number of edges, plans or activatable elements. The activatable elements are further divided into subprocess on the one hand and goals on the other. Goals can be defined to be of any of the four types defined previously.

The next set of elements are the plans, which currently includes both the BPMN plans with associated BPMN diagrams and activation plans for subgoal activation. The relationship between these elements can be expressed using one of the three edge types, however, the edges are restricted in what types of elements they can connect. Activation edges always have an activation plan as their source element, but their target can be any activatable element. In most cases this will be one of the goals, however, it can also target a subprocess, meaning the source activation plan will start the subprocess instead of adopting a goal.

Plan edges are used to associate plans with goals. Using multiple plan edges, several option can be provided for a single goal, enabling means-end reasoning. Note that plan edges can connect both activation plans or BPMN plans. Since the suppression edges influence goal reasoning, they can only have goals as both source and target. Finally, both activatable elements like goals and subprocesses as well as plans include the option of passing parameters.

As a result, the meta-model defines both the elements of the language GPMN themselves as well as the constraints in their use such as restrictions in the use of edges. The specific semantics of each elements is derived from the underlying BDI semantics which together allow a workflow engineer to develop the goal hierarchies of goal-oriented workflows.

4. Tools and Implementation

This section introduces key tools and components used to turn the concepts of Go4Flex and GPMN into a usable software environment. This environment aims to provide the necessary parts needed to support the implementation of workflows used in real-world organizations and support the remaining phases of the BPMN lifecycle, starting with the implementation phase and ending with the monitoring and analysis phases. Concretely, in the next sections the workflow editors for GPMN and BPMN, the execution environment including the workflow management system and monitoring and analysis tools will be presented.

4.1. *Workflow Editors*

An essential component for supporting the implementation phase of the BPM lifecycle are the tools for modeling new workflow models and modifying existing ones. The goal of such editors is the implementation of technically robust and semantically well-defined workflows allowing them to be executed on a workflow engine and being accessible even to non-technical personnel, which includes people with a business or management background. The goal-oriented workflows presented here are employ two different workflow languages: The first language is the goal-oriented part of the workflow, GPMN, which allows the workflow engineer to model a goal hierarchy of the process and decompose the workflow into small, concrete coals. The second language is BPMN, which is used to define the concrete set of actions for implementing a plan used to work towards a sufficiently refined goal.

As a result of having two languages, two editors are required to support both of them. Since we are also aiming at tool integration and use of already available tools for project management, both of the editors are implemented as plug-ins for

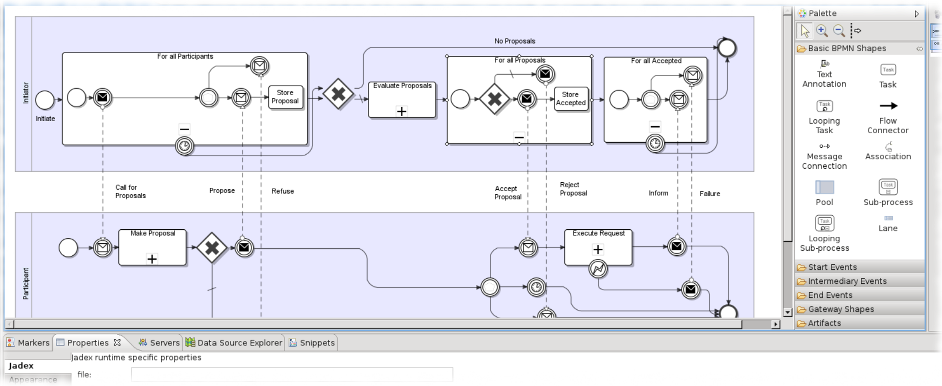

Fig. 5. The extended BPMN editor, showing a contract net protocol implementation being edited

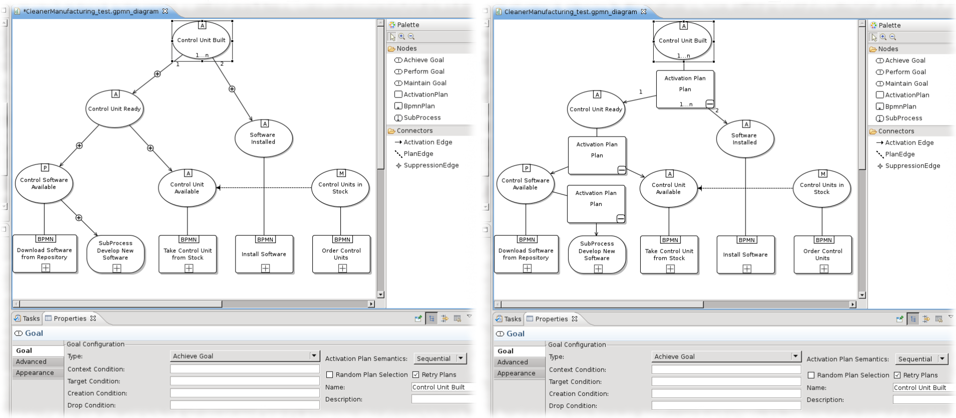

Fig. 6. The GPMN editor, demonstrating a business-oriented view where activation plans are hidden (left) and the full technical view (right)

the integrated development environment Eclipse.[a] The Eclipse project itself already provides a BPMN editor implementation, which already provides the means for graphical manipulation of BPMN elements. We have therefore extended this editor with some additional features in order to allow it to create BPMN workflows which are executable by our BPMN interpreter. These features primarily deal with data flow such as passing of parameters (see Fig. 5). The use of this editor is not solely restricted towards the creation of plans for GPMN workflows, but it can also be used to create standalone BPMN workflows.

The editor used to create GPMN goal hierarchies is also implemented as a plug-in for Eclipse, however, unlike the BPMN editor it had to be implemented from scratch. Aside from the enforcement of basic editing constraints based on the meta-

[a]http://www.eclipse.org/

model, such as allowing plan edges to only originate from goals and terminate with plans, the editor also supports additional features supporting the workflow engineer during design and implementation. For example, if the workflow engineer connects two goals, an activation plan is automatically inserted if necessary. Furthermore, the editor allows the user to adjust the visibility of certain elements such as activation plans, allowing the customization of the view for different groups of users (see Fig. 6). For example, business user may want to have a stronger focus on goals while technical users are concerned with the execution semantics of activation plans.

Both editors feature strict separation of graphical layout and semantical model. While most graphical elements like goals also feature a semantical representation, others can be a representation of multiple semantic elements or merely express a relationship between the semantic elements. In addition, the graphical layout contains information which concern the graphical representation only. This includes information about element sizes, font sizes and element locations. Therefore, the editors generate two different XML files, one representing the graphical layout which is only used by the editor and a second file containing the semantical model which is also used by the workflow engine to execute the workflow.

4.2. *Workflow Engine and Workflow Management System*

The next BPM lifecycle phase after implementation is the execution of the workflows generated by the editors. This requires an execution environment which allows them to be executed in a controlled manner. This *workflow engine* generates an active instance of the workflow model and manages the workflow state, which, in case of GPMN workflows, include the workflow context. Since our approach is motivated by features of BDI agents, our implementation of GPMN workflows employs BDI agents as representation of GPMN workflows. As a BDI implementation we employ the Jadex BDI agent system which we have enhanced with a BPMN interpreter to allow the execution of BPMN plans.

In addition to the use of Jadex as workflow engine, further components are needed to build a full workflow management system (WfMS). This includes the deployment of workflow models, controlled instantion and termination of workflow instances and, critically, the generation and distribution of work items to participants in the workflow. Such a workflow management system has been implemented as a Jadex application based on the reference model of the Workflow Management Coalition (WfMC)[9]. The system itself provides work item management, including a distribution system based on roles, role management, security and administrative features and workflow model deployment.

In addition, a basic workflow client application in included with the WfMS which allows users to interact with the workflow management system. The application uses the parameters requested by the work item to automatically generate a graphical form for the user to enter the requested data. This is done automatically and the appropriate GUI components are used for the relevant data types. For example, if

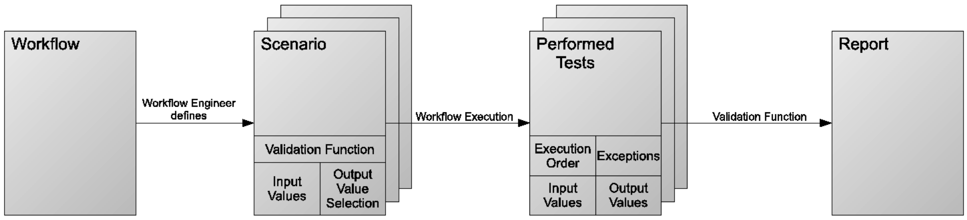

Fig. 7. Approach for using the simulation tool

a document is requested, the user is provided with an upload interface.

The workflow management system and the workflow engine is the basic infrastructure for the execution phase of the BPM lifecycle. This includes the use of the system in a production environment as well as test environments. For production use as well as simple tests, the workflow client allows workflow participants to interact with the system during execution.

4.3. *Simulation, Monitoring and Analysis*

The last phases of the BPM lifecycle are the monitoring and analysis phases. The goal of these phases is to inspect the workflows while they are executing, analyzing their behavior and using the results to improve the original workflow model. These phases are supported using a simulation tool and a set of monitoring tools. The simulation tool provide the means for controlled execution of the workflows for test purposes and behaves like workflow client application[10].

The execution behavior of the simulation tool depends on the configuration by the workflow engineer. Based on an analysis of the workflow model, the tool determines the information provided by workflow participants during execution. This is used by the engineer to develop *test scenarios* for the workflow (cf. Fig. 7). Test scenarios consist of sets of input values, a selection for output values and a validation function. The input values are the values that are usually provided by the workflow participants. The output value selection restricts the recording of workflow output to a smaller set, since not every output of the workflow is relevant to the test at hand. Finally, the validation function provides the means to determine whether a scenario has been successfully executed. The validation function can be both provided in a limited fashion using a graphical user interface or in a broader manner programmatically.

During this execution, the tool records information in addition to the selected output values such as process instantiations, goal adoptions and any exceptions occuring during execution. These events are recorded in an event log. This log is then used to generate a scenario report allowing the workflow engineer to review and analyze report as well as the the log in order to find unusual behavior of the workflow or outright errors in the workflow model.

In addition to the simulation tool, a number of tools originally used to monitor

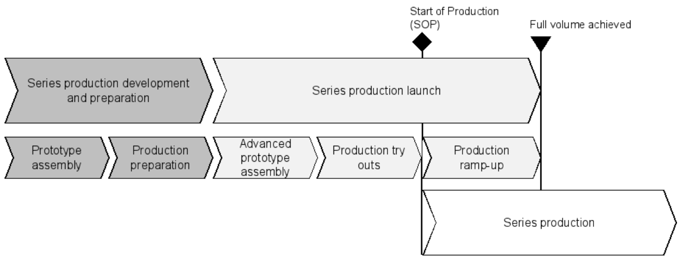

Fig. 8. Phases of series production launch

BDI agents have been adopted or extended to support GPMN and BPMN workflows and provide additional information during execution. This includes an overview of workflow instances to review currently active process. A ComAnalyzer tool for monitoring interprocess communication is also available. A process debugger lets the workflow engineer suspend processes and proceed execution in small steps while monitoring the process behavior. This is aided by a process introspector which shows the internal state of the process such as adopted goals, executing plans and the current value of context elements. These tools can also be used alongside the simulation tool for additional information about the workflow behavior.

This toolset enable the user to monitor and analyze workflows and provide automatic means for execution in a controlled manner. Furthermore, the automatic tests done by the simulation tool enable immediate feedback when the workflow misbehaves. This is particular important in later iterations of the BPM lifecycle when changes have been applied to the workflow and it is necessary if the rest of the workflow still behaves as expected and defined in the test scenarios.

5. Production Preparation Workflow

The Go4Flex goal concept has been used in several different areas at Daimler AG. This section presents an example workflow used in the preparation stages of the production of a new line of car. The first part consists of a general overview of the process and the context in which it is used at Daimler AG. The second part will contains a simplified model of that process in GPMN. This example will illustrate the practical application of the modeling concepts presented in the previous sections.[b]

5.1. *Overview and Process Context*

In general, production planning in the automotive industry is a collaborative issue. Many different planning departments are involved. Product development, Body in White planning, Factory Layout planning, Logistics and Assembly planning are typical planning departments in the automotive industry. The proposed example

[b]The original workflow has been made abstract due to business secrecy reasons.

workflow “production preparation” is part of the series production launch process at Daimler. This process contains the following different phases (cf. Figure 8):
1. Phase: Series production development and preparation (Prototype assembly, production preparation)
2. Phase: Series production launch (Advanced prototype assembly, production try outs, production ramp-up)
3. Phase: Series production

In the first phase, prototypes will be assembled and the production will be prepared. The subsequent phase series production launch deals with assembling prototypes under conditions of series production. With the start of production (SOP), the first product for a costumer will be produced and the production ramp-up as well as the series production will start. The phase from SOP to the point “full volume achieved” is defined as production ramp-up. After achieving the full volume, the production ramp-up phase ends[24].

The production preparation phase is applied to ensure an efficient production ramp-up in assembly. The production preparation process is an iterative process where many planning departments come together to improve processes for series manufacturing. Therefore, physical prototypes are used. In so called station preparation workshops, the physical prototypes will be assembled step by step for each station. Basis for this assembly is the current planning state of an operation list. The operation list is a complete list of all necessary assembly operations. The main part of final assembly in automotive industry is done by workers manually and the careful considerations of prototype cars affect the later assembly processes. Each planning department has a specific view on the performed processes. Therefore, lots of different specific criteria have to be validated through these workshops. Wack et. al describes four main topics of validation aspects during production preparation[30]:
- Production oriented product validation
- Product oriented process validation
- Production oriented process validation
- Validation of resources

Production oriented product validation describes the validation of product focused on production view, like composability of product. Product oriented process validation is related to value adding processes which affect the product directly, like screwing or twisting together. The validation of production oriented processes ensures that necessary processes for the production are efficient. For example, worker paths have to be short as possible and logistic routes have to be optimized. Finally, the validation of resources focussed on uses tools, cargo carriers etc. The Validation of these partially interdependent aspects is done in the production preparation phase to ensure an efficient production ramp-up in assembly.

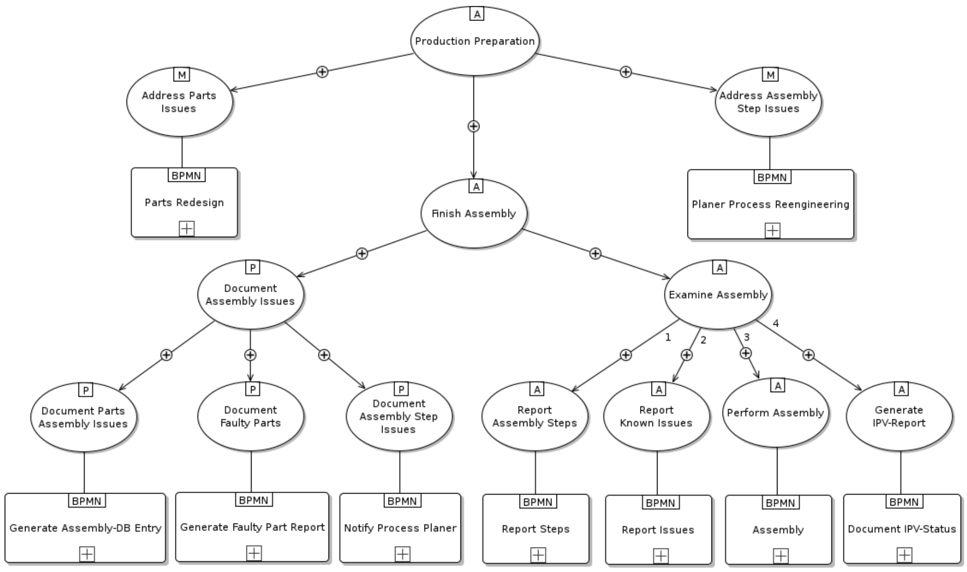

Fig. 9. Partial model for the product preparation process

5.2. *Process Model*

The production preparation is a process, where groups from multiple areas of development gather to review the vehicle assembly. The production preparation process involves a large set of activities surrounding the preparation of a new product line, which is a very complex process that includes a large number of participants. Here we will present partial model of these activities with a focus on some of the activities involved in the assembly. Since the modeling approach of GPMN involves a top down approach, the modeling process starts with a general goal describing overall objective of the process. In this case, the objective of the process is finishing the product preparation for the new product line. Since this goal has a clear objective that can be reached by finishing the production preparation, it is an achieve goal.

This first goal starts the goal hierarchy of the process as can be seen in Figure 9. The goal itself is then decomposed into subgoals, which together are aimed at accomplishing the overall goal. In this process, two different aspects have to be considered. First, the assembly has to be performed in order to identify issues with components and parts of the vehicle and to identify issues regarding the production process itself. Afterwards, the issues found during the assembly have to be addressed and resolved. Parts and components can be faulty either by being complicated to install in the vehicle or by including faults which prevent their installation altogether. Faults in the production process include aspects such as erroneous order installation activities such as installing a part which will obstruct the installation of a later part, but also process optimizations like redundant steps.

Since the assembly has a reachable objective, it as modeled as a achieve goal as well. The process itself is accomplished using two subgoals. The right branch of this sub-hierarchy describes the examination of the assembly by experts. This goal is an achieve goal called "Finish Assembly" with the objective of performing and examining the assembly process. This is done in four steps, which, unlike previous goals, are performed in sequential order. First, the assembly steps are read out to the experts in the process in order to give an overview of the assembly and to allow a first examination of the assembly itself. The first production issues can be identified by the experts in the process at this point. The next step is to report issues that have already been found and are currently being worked on. This avoids identifying issues a second time. The next step is the assembly itself, where a worker will assemble the listed parts. The experts will review the assembly and identify and discuss potential problems that could impact production. Finally, the experts will present the current state of the production process in a report, indicating whether the production process has reached a satisfactory state or if there are still issues which require intervention. All of these goals are modeled as achieve goals.

The left branch of the "Finish Assembly" sub-hierarchy deals with the documentation of issues that are found during the assembly. Its subgoals primarily deal with the various issues that need to be recorded and record them in the appropriate databases. Since this is a continuous process running alongside the assembly itself, it is modeled to be performed parallel to the assembly. Unlike the assembly itself, there is no clear objective that can be finished, such as finishing the assembly. Therefore, the goals in this hierarchy are modeled as perform goals. Production issues are continuously gathered while the assembly is being done. A drop condition in the documentation hierarchy goals will cause the documentation goals to be dropped once the assembly sub-hierarchy finishes, thus stopping the documentation of further production issues.

The issues found during the assembly process need to be resolved at a later point by the relevant developer or group of developers. For this reason, the process also includes two maintain goals which are subgoals of the overarching "Production Preparation" goal, running parallel to the assembly process. The maintain conditions of those goals aim at keeping the list of unresolved issues empty. If this condition gets violated, for example by adding a new issue during the assembly, the attached plan becomes active and performs the relevant activity to resolve this issue. For issues involving the production process, a process reengineering task is performed to change the production process. For faulty parts, a part redesign will improve the part by allowing easier installation or resolving issues preventing installation.

The workflow demonstrates how GPMN can be used to implement an important real world process. Some aspects in particular highlight the advantages of the approach. For example, the use of maintain goals to process the production issues is a feature that is not easily reproduced in an integrated workflow using procedural approaches like BPMN, requiring the use of subprocessing or other workarounds. Also, the use of perform goals to continuously document a number of production issues of an unknown quantity is easily implemented in GPMN using drop conditions but much harder to do in other workflow languages. Finally, the goal-oriented workflow immediately shows documentation, the assembly process itself and the processing of issues as important, high-level goals, documenting the business purpose of the workflow instead of focusing on particular activities.

6. Conclusion

This paper presented an approach for modeling goal-oriented business processes. It has been argued that the goal oriented approach has definite advantages over traditional procedural approach by exploiting concepts from multi-agent systems for improving the conceptual underpinnings. One fundamental building block of Go4Flex is the usage of declarative goals, which increase the understandability and abstractness of workflow descriptions. On the one hand functional goals help representing the underlying reasons for executing a process and on the other hand non-functional goals support the understanding of the process metrics.

In order to explore the new concepts workflow modeling, the paper presented a BPM lifecycle and described both methods, concepts and tools to support it. This includes languages and editors for the first parts, execution and management systems for the next parts and simulation, monitoring and analysis tools for the final parts of the lifecycle.

Future work will further develop and systematize the available modeling concepts. In addition, we aim to apply more multi-agent concepts to workflows, such as applying organizational strcutures and team behavior.

Acknowledgement: We would like to thank the DFG for supporting the technology transfer project Go4Flex.

References

1. M. Bratman. *Intention, Plans, and Practical Reason.* Harvard University Press, 1987.
2. L. Braubach and A. Pokahr. Representing long-term and interest bdi goals. In *Proc. of (ProMAS-7)*, pages 29–43. IFAAMAS Foundation, 5 2009.
3. L. Braubach, A. Pokahr, D. Moldt, and W. Lamersdorf. Goal Representation for BDI Agent Systems. In *Proc. of (ProMAS 2004)*, pages 44–65. Springer, 2005.
4. J. Broekens, M. Harbers, K. Hindriks, K. van den Bosch, C. Jonker, and J.-J. Meyer. Do you get it? user-evaluated explainable bdi agents. In *MATES 2010*, pages 28–39. Springer, 2010.
5. B. Burmeister, M. Arnold, F. Copaciu, and G. Rimassa. Bdi-agents for agile goal-oriented business processes. In *AAMAS '08*, pages 37–44. IFAAMAS, 2008.
6. M. Calisti and D. Greenwood. Goal-oriented autonomic process modeling and execution for next generation networks. In *Proc. of MACE '08*. Springer, 2008.
7. B. Curtis, M. Kellner, and J. Over. Process modeling. *Com. ACM*, 35(9):75–90, 1992.
8. W. Sesselmann H. J. Schmelzer. *Geschäftsprozessmanagement in der Praxis.* Hanser Fachbuchverlag, 2008.
9. D. Hollingsworth. *Workflow Management System Reference Model.* Workflow Management Coalition, 1995.
10. K. Jander, L. Braubach, A. Pokahr, and W. Lamersdorf. Validation of agile workflows using simulation. In *LADS 2010*, pages 41–47. CEUR Workshop Proceedings, 8 2010.

11. K. Jander and W. Lamersdorf. Gpmn-edit: High-level and goal-oriented workflow modeling. In *WowKiVS 2011*. Electronic Communications of the EASST, 2011.
12. G. Knolmayer, R. Endl, and M. Pfahrer. Modeling processes and workflows by business rules. In *Business Process Management, Models, Techniques, and Empirical Studies*, pages 16–29, London, UK, 2000. Springer-Verlag.
13. B. List and B. Korherr. An evaluation of conceptual business process modelling languages. In *Proc. of SAC '06*, pages 1532–1539. ACM, 2006.
14. A. Newell. *Unified Theories of Cognition*. Harvard University Press, 1990.
15. OASIS. *Web Services Business Process Execution Language (WSPBEL) Specification*, version 2.0 edition, 2007.
16. Object Management Group (OMG). *Business Process Modeling Notation (BPMN) Specification*, version 1.1 edition, February 2008.
17. C. Ouyang, M. Dumas, A. ter Hofstede, and W. van der Aalst. From bpmn process models to bpel web services. In *Proc. of ICWS '06*, pages 285–292. IEEE, 2006.
18. C.A. Petri. *Kommunikation mit Automaten*. PhD thesis, Institut fur Instrumentelle Mathematik, Bonn, 1962.
19. A. Pokahr, L. Braubach, and W. Lamersdorf. Jadex: A BDI Reasoning Engine. In R. Bordini, M. Dastani, J. Dix, and A. El Fallah Seghrouchni, editors, *Multi-Agent Programming: Languages, Platforms and Applications*, pages 149–174. Springer, 2005.
20. A. Pourshahid, D. Amyot, L. Peyton, S. Ghanavati, P. Chen, M. Weiss, and A. Forster. Business process management with the user requirements notation. *Electronic Commerce Research*, 9(4):269–316, 2009.
21. A. Rao and M. Georgeff. BDI Agents: from theory to practice. In V. Lesser, editor, *Proceedings of the 1st International Conference on Multi-Agent Systems (ICMAS 1995)*, pages 312–319. MIT Press, 1995.
22. A. S. Rao and M. P. Georgeff. An abstract architecture for rational agents. In *Proceedings KR 1992*, pages 439–449, 1992.
23. A.-W. Scheer and M. Nüttgens. Aris architecture and reference models for business process management. In *Business Process Management*. Springer, 2000.
24. G. Schuh, W. Stölzle, and F. Straube. *Anlaufmanagement in der Automobilindustrie erfolgreich umsetzen*. Springer-Verlag, 2008.
25. P. Seshan. *Process-Centric Architecture for Enterprise Software Systems*. CRC Press, 2010.
26. W. M. P. van Der Aalst, A. H. M. Ter Hofstede, B. Kiepuszewski, and A. P. Barros. Workflow patterns. *Distrib. Parallel Databases*, 14(1):5–51, 2003.
27. W. M. P. van der Aalst and A. H. M. ter Hofstede. Yawl: yet another workflow language. *Information Systems*, 30(4):245–275, June 2005.
28. W. M. P. van der Aalst, A. H. M. ter Hofstede, and M. Weske. Business process management: A survey. In *Business Process Management*, 2003.
29. A. van Lamsweerde. Goal-Oriented Requirements Engineering: A Guided Tour. In *Proceedings of the 9th International Joint Conference on Requirements Engineering (RE 2001)*, pages 249–263. IEEE Press, 2001.
30. K.-J. Wack, T. Bär, and S. Straßburger. Grenzen einer digitalen Absicherung des Produktionsanlaufs. In *Proceedings ASIM*, 2010.
31. M. Weske. *Business Process Management Concepts, Languages, Architectures*. Springer Verlag, 2007.
32. M. Winikoff, L. Padgham, J. Harland, and J. Thangarajah. Declarative & Procedural Goals in Intelligent Agent Systems. In *Proc. of KR 2002*. Morgan Kaufmann Publishers, 2002.