Distributed Event Processing for Goal-Oriented Workflows

Kai Jander, Lars Braubach, and W. Lamersdorf

Abstract. Goal-oriented workflows enable workflow designers to easily include a high degree of flexibility while implementing and automating business process. In addition, modeling is based on the business goals of the organization instead of actions, allowing for better alignment with those goals. Combined with a distributed workflow management system, this allows for a highly agile workflow environment that can adapt to the business situation while maintaining the structure of a solid workflow model. However, one of the drawback of such processes is a lack of attribution of the actions of the workflow to specific goals. As a result, improvements to the monitoring side of process management are necessary in order to make such associations clearer and allow easier workflow analysis and reengineering. This paper presents an approach for a component-based event system that introduces a degree of structure to the events, enabling the association of events with the workflow model, facilitating real-time monitoring of goal-oriented process and process drill-down analysis.

1 Introduction and Motivation

In business process management, workflows represent the partially automatized part of a business process implemented to be executed on a computer system. While they play an important part of business automation, traditional workflow models such as the Business Process Model and Notation (BPMN) [13] often lack good support for flexibility in a multitude of business situations, requiring the modeler to include numerous branches for every conceivable situation. This deficit has lead to a number of different approaches like ADEPT [12] attempting to attenuate those limitations. Part of the difficulties stem from the fact that most approaches to workflows are

Kai Jander · Lars Braubach · W. Lamersdorf
Distributed Systems and Information Systems Group, University of Hamburg, Germany
e-mail: {jander,braubach}@informatik.uni-hamburg.de

activity-based, which means they are focused on a specific order of actions with branching explicitely inserted at certain points. While this is a fairly intuitive approach for modeling workflows, it does not directly provide business reasons for activities, which can result in the inclusion of unnecessary or unwanted activities that merely exist for technical rather than business reasons in the workflow.

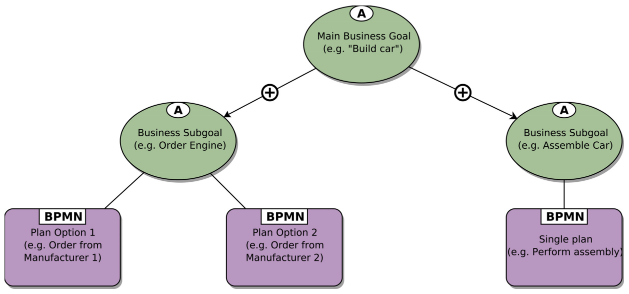
A
Main Business Goal (e.g. "Build car")
A
Business Subgoal (e.g. Order Engine)
A
Business Subgoal (e.g. Assemble Car)
BPMN
Plan Option 1 (e.g. Order from Manufacturer 1)
BPMN
Plan Option 2 (e.g. Order from Manufacturer 2)
BPMN
Single plan (e.g. Perform assembly)

Fig. 1 Example of a simplified goal-oriented process

An interesting concept for addressing both concerns are goal-oriented workflows [10]. In this approach, Belief-Desire-Intention (BDI) agents [4] are used to model workflows. The BDI approach allows the workflows to be based on business goals (see Figure 1), starting with the overall goal the business aims to accomplish with the workflow which is then subdivided into multiple subgoals with various types of interactions such as priorities and inhibitions. These subgoals taken together represent the top-level goal by completely covering all aspects of that goal. The subgoals themselves can again be further decomposed until the goals are plain enough to be achievable by a simple action-based workflow. If multiple solutions for a goal are possible, multiple plans can be attached to the goal and a plan is chosen at runtime based on attached conditions and attributes. Key for both the conditions and attributes of goals and plans is the existence of a workflow context, which not only holds the internal state of the workflow but also contains information about the current business situation, allowing this information to be accessed by goal and plan conditions.

The methodology of generating GPMN models is similar to Hierarchical task network (HTN) [5] planning approaches, however, while the goal-structure of GPMN processes often end up being hierarchical, they do have to be. In addition, GPMN-modelling is not used for feasibility analysis or ahead-of-time planing. Rather, it allows process engineers to specify business contingencies and competing objectives for long-running processes in dynamic environments such as R&D processes at Daimler AG [9]. Instead, the resulting agent model can be directly executed by a BDI interpreter, which uses a BPMN interpreter for concrete plans. The reasoning of BDI is split between the goal deliberation cycle, which decides which of potentially several conflicting goals to follow and the means-end reasoning which choses pre-defined plans to achieve selected goals. The BDI interpreter reasoning engine uses an approach called Easy Deliberation [16] to resolve this cycle.

While this top-down and business-driven approach towards workflow modeling is very intuitive and flexible, its adaptive behavior during runtime means that actions are not always attributable to the goals [8]. In addition, goal-based workflows are also often used in dynamic environments where not only the workflows but also the workflow management system are distributed to increase flexibility and robustness [11]. As a result, a distributed monitoring system is necessary that links the actions performed in the workflow with the goals and plans representing the business reasons.

Based on the requirements for a distributed workflow management system, distributed workflow execution and cohesion between goals and action, the proposed system should be able to meet the following objectives:

- Goal-Action Cohesion
  - The primary goal of the approach is providing event information that allow attribution of concrete business actions with business goals in a workflow.
  - The user must be able to perform a “drill-down” analysis to trace causes of events from the most detailed to the most high-level goals.
- Distributed Workflow Management
  - The system must be adaptable to a wide variety of business infrastructures, including transient and mobile systems and must respond in a robust fashion to changes in that infrastructure.
  - As a result, the system must be distributed, redundant, robust and adaptable. Communication must be kept low for efficiency.

In this paper we present a component- and service-based approach attempting to solve these objectives. It supports monitoring distributed goal-based workflows using a hierarchical structure of events combined with a distributable monitoring system for gathering and redestribution of the events to the workflow clients.

2 Related Work

The approach presented in this paper primarily touches two important area of research. The first is the area of *Business Activity Monitoring (BAM)*. This area concerns itself how business transactions happening within a company can be recorded and processed, either retroactively or in realtime. The most common approach to this challenge is to record business transactions, either specifically for monitoring purposes or incidentally as part of regular business record keeping, in a *data warehouse* [4]. In addition, *extract, transform, load (ETL)* processes can be used to extract additional data from other sources within the business [18]. The resulting data can be utilized in two ways: Complex analysis can be performed and long-term statistics can be gained by applying data mining techniques to the data available in the data warehouse [1]. This offers the user an in-depth and long-term perspective of the perfomance of the organization. However, it can require substantial time and computation to process the available data and thus may lag behind the current development of the business. For example, *online analytical processing (OLAP)* allows for multidimensional analysis of transactions and currently available data within the organization [2], but the data must already be available in a structured fashion, for example in a data warehouse. The data is processed to allow the user to approach it from multiple perspectives using multidimensional analysis [17] by forming structures such as OLAP cubes.

Alternatively, realtime monitoring can be achieved using *complex event processing (CEP)* [6]. This approach gathers and processes events as a stream and generates useful information for the BAM system. While aspects of this approach are similar to the approach presented here, it is focused on the processing of data, rather than providing a stronger coherence between the operational and strategic level by aligning actions and goals.

The resulting information can then be used to display information in a dashboard specific to the interest to the business user. Statistics and indicators are provided to allow the user to monitor the actual performance of the business and compare it with previously strategically defined *key performance indicators (KPI)*, which represent quantifiable values. However, the relationship between the processed data and the KPIs is implicit and cannot be derived from the data warehouse itself. As a result, the dashboard functionality of BAM systems often need to be customized to restore this relationship, which can only be partially compensated through standardization of common KPIs or interpretation of standardized charts on the dashboard. Furthermore, BAM focuses primarily on statistical data and thus does not provide an easy way for attributing transactions of the business with particular strategic goals. In contrast, the approach presented here focuses on attributing actions and tasks to business goals, while the generation of statistics is less of a concern. In addition, data mining solutions often focus on a centralized data warehouse solution while the monitoring system presented here provides distributed realtime monitoring.

The second area concerns itself with distributed event systems [14]. A common approach here is to employ *event brokers*, which receive published events from *event publishers* and then distribute them among the other brokers within the system, making them available to *event subscribers* throughout the system. Similar to this approach, we use one or more monitoring component instance to implement the broker functionality and distribute our goal-based events as part of a larger distributed workflow management system. The number of such brokers used can be chosen by the user of the system, with additional brokers increasing the redundancy and thus the resilience of the overall system and increasing the event processing performance by bundling event transfers.

3 Event Structure

The targeted goal-oriented workflows are based on the Jadex Active Components [19] approach. This approach offers a number of different active component types such as micro agents, BDI agents, BPMN workflows and the goal-oriented GPMN workflows, which are modeled as goal-plan hierarchies but are translated to BDI agents for execution.

Events generally denote an occurence associated with an *event source*. An event source can be any entity or part of an entity within the distributed system such as an active component itself or some part of the component. In order to broadly distinguish events based on sources such as components, goals and plans, a event source can be attached to the event. A number of default categories are already provided for all active component types, including the component category for events emitted by component instances itself, the execution category for the execution of components steps, the service category for component services and the property category for component property. More specialized categories are available for action-oriented and goal-oriented workflows. Event source categories for activities, goals and plans are available in addition to the workflow context fact category. The latter category is also an example of an event source category where the modification event applies.

While the exact nature of the event can vary wildly, a number of categories can be identified that not only define the relationship with the event source and between events over time. For example, BPMN offers three broad classes of event types: Start events, which mark the start of a process, end events, which denote the process end and finally intermediate events which can occur while the process is running.

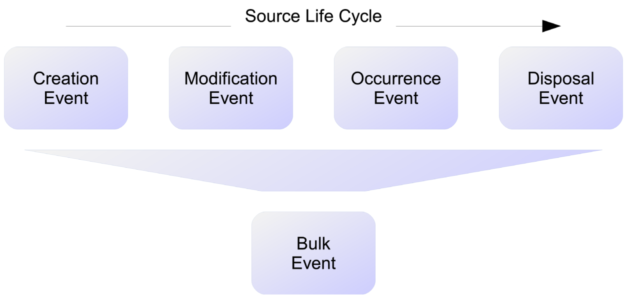

Fig. 2 Event types used over the event source life cycle

Similar to this, we have based our system on five different types of events based on the life cycle of the source (see Figure 2). The first type are *creation events*, which are issued when the event source is first created, providing similar semantics as BPMN start events. The counterpart for end events are *disposal events*, issued when the life cycle of the event source has ended and the source has been disposed. During the life cycle, two additional types of events can occur: When the state of the source changes it results in a *modification event*, while the other event type is the *occurrence event*, which simply denotes a point in time when an action took place. For example, an occurrence event is generated when an agent receives an external message or an internal user event is triggered. Finally, a special type of event, the bulk event, only applies when multiple events are aggregated into a single event for efficiency when transferring events over the distributed system.

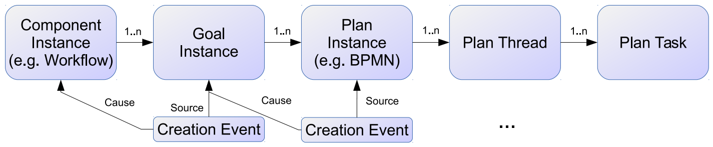

Fig. 3 Events can have both sources and causes based on the static component model and their runtime instances

Since the active component model uses a static component model, the component hierarchy can be used to denote both the source and the *cause* of an event (cf. Fig. 3). For example, a goal-oriented workflow component, when executed, consists of a component instance. If this component instance adopts a new goal instance, a creation event is issued. The source of this creation event is the newly-created goal instance, however, the cause of the event is the component instance which adopted the goal. Once the means-end reasoning decides on a plan to perform in order to reach the goal, a plan instance is created and another creation event is issued. In this case, the source is the plan instance while the cause of the event is the goal instance that triggered the means-end reasoning. This chain can be traced down the model hierarchy down to tasks within BPMN-based plans.

Since instances have unique identifiers, the events merely have to include the cause and source identifiers, minimizing the size of each event. Nevertheless, once the events have been gathered, the cause and effect chain can be recreated by cross-referencing them with other events. In addition, events that are delayed due to the distributed nature of the system can be identified and estimates can be given. For example, if the creation event of the plan instance is missing, the lower boundary for the creation time of the plan instance is the creation time of its cause, i.e. the goal instance. This can at the very least provide the user with an adequate approximation until the missing events arrive.

Each event also requires a timestamp, identifying when the event occurred, especially in relationship to other events. In distributed systems, simple timestamps often pose an issue since it is difficult to reliably synchronize clocks between nodes. Other approaches such as vector clocks increase both complexity and data volume. However, in the case of goal-oriented workflows as presented here, the problem is reduced since each individual workflow instance runs on a single node and its events are therefore internally consistent. Only if the cause chains crosses node barriers, for example due to service calls, caution has to be applied regarding synchronization. Nevertheless, small inconsistencies can be corrected to a certain degree using the same approach as mentioned above regarding missing events.

4 Monitoring Architecture and Implementation

The monitoring mechanism has been implemented using a monitoring service (IMonitoringService), which allows producers to publish events and consumers to subscribe for certain types of events. In Fig. 4 an overview of the architecture is depicted. It can be seen that in each platform a specific monitoring component realizes the monitoring service. Each component that creates (internally or intentionally) events, automatically publishes them to the corresponding service. For this purpose each component searches for an IMonitoringService when an event has to be published. In case a service is found the binding is fixed and the event is forwarded, if not the component stores the unavailability of the service and the event is dismissed. It will try to search for the service again when a new event occurs and after some specified time interval has elapsed. Event consumers can subscribe at the monitoring service using an optional event filter. This mechanism allows for reducing the network load as only events are transferred which pass the filter test.

The global monitoring infrastructure is set up as a peer to peer infrastructure formed by multiple monitoring components realizing an information exchange protocol. Knowing that events are reported locally from the event sources, each monitoring service searches for all other monitoring services and forwards local events to all remote services. In this way all monitoring services internally build up a globally consistent event state. For scalability reasons, the monitoring components only hold a certain amount of event in memory and dismiss older events. If longer lasting book keeping is necessary they can also be configured using a distributed database to store older events.

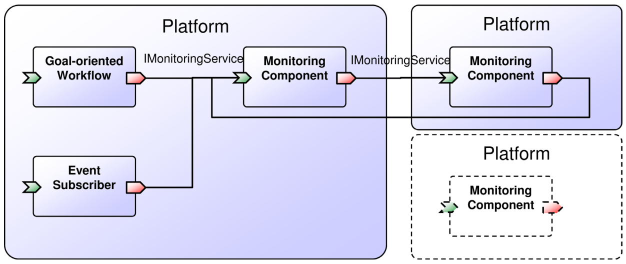

Fig. 4 Monitoring infrastructure

4.1 Workflow Environment

The monitoring service is further used to supplement a distributed workflow management system [11] implemented as active components.

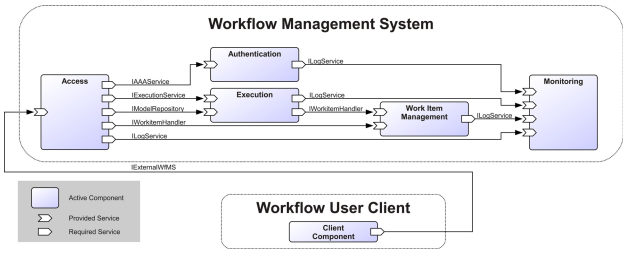

Fig. 5 Event types used over the event source life cycle

The system is largely derived from the Workflow Management Coalition Reference Model [7], where the monitoring service represents the monitoring subsystems of a traditional workflow management system. The workflow management system consists of multiple active components similar to the monitoring service which use service calls to exchange information.

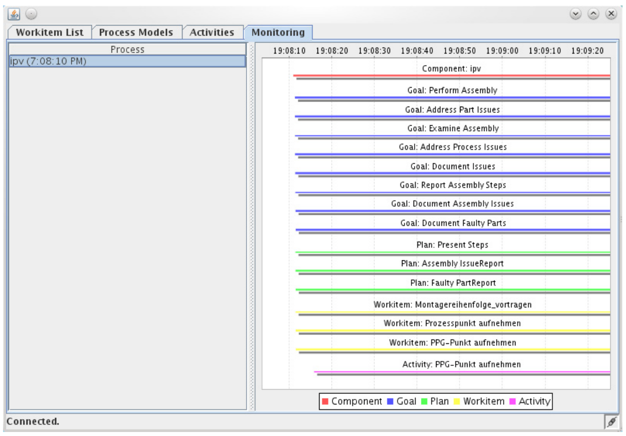
Workitem List
Process Models
Activities
Monitoring
Process
ipv (7:08:10 PM)
19:08:10
19:08:20
19:08:30
19:08:40
19:08:50
19:09:00
19:09:10
19:09:20
Component: ipv
Goal: Perform Assembly
Goal: Address Part Issues
Goal: Examine Assembly
Goal: Address Process Issues
Goal: Document Issues
Goal: Report Assembly Steps
Goal: Document Assembly Issues
Goal: Document Faulty Parts
Plan: Present Steps
Plan: Assembly IssueReport
Plan: Faulty PartReport
Workitem: Montagereihenfolge_vortragen
Workitem: Prozesspunkt aufnehmen
Workitem: PPG-Punkt aufnehmen
Activity: PPG-Punkt aufnehmen
🟥 Component 🟦 Goal 🟩 Plan 🟨 Workitem 🟪 Activity
Connected.

Fig. 6 Gantt chart monitoring of a goal-oriented workflow in the workflow client

Aside from the monitoring, other components are available to provide the rest of the workflow management functionality. The access component manages the access to workflow management functionality for workflow clients. The authentication component verifies the credentials of users and specifies their access rights. The execution component stores workflow models and launches workflow instances. Finally, the work item component holds work items representing human tasks for users to perform. Each component can be replicated to provide robustness and scalability.

The information gained by the monitoring component can be used to provide an overview of a goal-oriented workflow instance. As shown in Fig. 6, users can select a running or past workflow instance and is presented with a Gantt chart of event sources. The user can click on individual sources and is provided with a view which includes the clicked item and sources that were created as a result of the item, providing a drilled-down view on parts of the process instance. If only partial information is available, an estimate can be shown to the user based on other event sources in the hierarchy. However, due to the chaining of references a subtree may be omitted if the cause-source hierarchy is interrupted. While this is remedied as soon as the missing information arrives, it is still a limitation of the system.

5 Evaluation and Outlook

In Section 1 we provided two areas where the system had to fulfill a set of requirements. The first area involved the cohesion between business goals and workflow actions. Here, the monitoring system is required to establish a relationship between the actions of the workflow and the business goals and allow the user to "drill down" from individual goals down to specific actions. The modeling of the goal-oriented workflows allow the events to establish a cause-source hierarchy to attribute actions to goals.

The first set of requirements was due to the required flexibility of a distributed workflow management system. The events may be generated by workflows anywhere within the distributed system, forwarded to interested nodes and the system should be robust and tolerate disappearing nodes. These three requirements were achieved by implementing the monitoring system as a event broker, forwarding all events it receives for publication to corresponding services. Unless a node is permanently disabled before it can transmit new events, all events will eventually be available to the workflow client. Low communication overhead was achieved by minimizing the amount of information stored in the events, including only references to sources and causes and letting the receiver reconstruct the chain. Furthermore, bulk events are available to bundle multiple events for transfer.

However, a remaining concern is the potential loss of all events of a specific event source. Due to the events containing only a minimum of information, this would interrupt a link between the main hierarchy tree and one of its subtrees. While there is a balance involved with regard to the event sizes, this challenge could be address by including a more information about the chain in each event by including not only the current causes but also a number of previous causes, allowing the workflow client to include the subtree and only omit the missing source.

References

1. Berry, M.J., Linoff, G.: Data Mining Techniques: For Marketing, Sales, and Customer Support. John Wiley & Sons, Inc., New York (1997)
2. Berson, A., Smith, S.J.: Data Warehousing, Data Mining, and Olap, 1st edn. McGraw-Hill, Inc., New York (1997)
3. Bratman, M.: Intention, Plans, and Practical Reason. Harvard University Press (1987)

4. Chaudhuri, S., Dayal, U.: An overview of data warehousing and olap technology. SIGMOD Rec. 26(1), 65–74 (1997)
5. Erol, K., Hendler, J., Nau, D.S.: Htn planning: Complexity and expressivity. In: AAAI, vol. 94, pp. 1123–1128 (1994)
6. Greiner, T., Düster, W., Pouatcha, F., von Ammon, R., Brandl, H.-M., Guschakowski, D.: Business activity monitoring of norisbank taking the example of the application easycredit and the future adoption of complex event processing (cep). In: Proceedings of the 4th International Symposium on Principles and Practice of Programming in Java, PPPJ 2006, pp. 237–242. ACM, New York (2006)
7. Hollingsworth, D.: Workflow Management System Reference Model. Workflow Management Coalition (1995)
8. Jander, K., Braubach, L., Pokahr, A., Lamersdorf, W.: Validation of Agile Workflows Using Simulation. In: Dastani, M., El Fallah Seghrouchni, A., Hübner, J., Leite, J. (eds.) LADS 2010. LNCS, vol. 6822, pp. 39–55. Springer, Heidelberg (2011)
9. Jander, K., Braubach, L., Pokahr, A., Lamersdorf, W., Wack, K.-J.: Goal-oriented processes with gpmn. International Journal on Artificial Intelligence Tools (IJAIT) 20(6), 1021–1041 (2011)
10. Jander, K., Lamersdorf, W.: Gpmn-edit: High-level and goal-oriented workflow modeling. In: WowKiVS 2011, vol. 37, pp. 146–157 (2011)
11. Jander, K., Lamersdorf, W.: Jadex WfMS: Distributed Workflow Management for Private Clouds. In: Conference on Networked Systems (NetSys), pp. 84–91. IEEE Xplore (2013)
12. Jennings, N., Norman, T., Faratin, P.: ADEPT: An agent-based approach to business process management. ACM SIGMOD Record 27(4), 32–39 (1998)
13. Object Management Group (OMG). Business Process Modeling Notation (BPMN) Specification, version 2.0 edition (January 2011)
14. Pietzuch, P.R., Bacon, J.: Hermes: A distributed event-based middleware architecture. In: Proceedings of the 22nd International Conference on Distributed Computing Systems, ICDCSW 2002, pp. 611–618. IEEE Computer Society, Washington, DC (2002)
15. Pokahr, A., Braubach, L.: The active components approach for distributed systems development. International Journal of Parallel, Emergent and Distributed Systems 28(4), 321–369 (2013)
16. Pokahr, A., Braubach, L., Lamersdorf, W.: A goal deliberation strategy for BDI agent systems. In: Eymann, T., Klügl, F., Lamersdorf, W., Klusch, M., Huhns, M.N. (eds.) MATES 2005. LNCS (LNAI), vol. 3550, pp. 82–93. Springer, Heidelberg (2005)
17. Vassiliadis, P., Sellis, T.: A survey of logical models for olap databases. SIGMOD Rec. 28(4), 64–69 (1999)
18. Vassiliadis, P., Simitsis, A., Skiadopoulos, S.: Conceptual modeling for etl processes. In: Proceedings of the 5th ACM International Workshop on Data Warehousing and OLAP, DOLAP 2002, pp. 14–21. ACM, New York (2002)