Towards Context-aware Workflows

Matthias Wieland[1], Oliver Kopp[1], Daniela Nicklas[2], Frank Leymann[1]

[1] Institute of Architecture of Application Systems, University of Stuttgart,
Universitätsstraße 38, 70569 Stuttgart, Germany
{wielanms, koppor, leymann}@iaas.uni-stuttgart.de
[2] Institute of Parallel and Distributed Systems, University of Stuttgart,
Universitätsstraße 38, 70569 Stuttgart, Germany
danickla@informatik.uni-stuttgart.de

Abstract. Context-aware applications adapt their behavior based on changes of the physical world, which is often obtained with a huge amount of sensors. Thus, the development of such applications is cumbersome, in particular the implementation of their often complex control flow. To ease the development of context-aware applications we present the concept of context-aware workflows. Thereafter we present an implementation of these concepts based on a standard workflow language. Context-aware workflows are not only interesting for the development of context-aware applications, but also enable workflow technology to be applied in new domains that are process oriented and yet not supported by workflow systems like production processes in the manufacturing industry. The concept of context-aware workflows is a first approach that enables modeling and execution of technical production processes with workflow systems normally used for business processes.

Keywords: workflow systems, context-aware systems, ubiquitous systems, workflow modeling, development of context-aware applications, BPEL, Nexus

1 Introduction

Workflow technology gained major influence in many enterprises and within the software industry. It provides methodologies and corresponding products to support modeling, execution, and management of business processes that have been carried out manually and through a diversity of non-integrated systems before. The integration of these non-integrated systems is possible by employing workflow management systems. The flow of papers became a flow of electronic documents, and the management got deeper insight on how their business really worked, which leads often to major improvements using business process reengineering. And best of all, with modeling their processes (instead of hard-coding them into their systems), the enterprises were enabled to change their processes more easily; they became more flexible and adaptive, to survive within a more and more dynamic market [1].

However, there are major application domains where processes are not supported by workflow technology yet. The domain we are focusing on are production environments in factories. The manufacturing or maintenance processes executed there are

planned using a production planning system (PPS). The layout of the resulting assembly line and the collaboration of technical machinery controlled by workers implement the production processes. We call these processes technical processes, since they are executed by technical machinery such as assembly lines, robots or other machines. A difficulty of technical processes is that they are crossing the boundary to the physical world. Here, manual or paper written coordination, and again, non-integrated systems (e.g., production planning and control systems) are current state-of-the-art for controlling and executing the corresponding processes. A fundamental difference between traditional business processes and technical processes is that within technical processes events signaled by the real physical world are of particular importance. Since miniature sensor technology and wireless communication becomes ubiquitous, it is possible to capture and observe more and more of these real world events. This leads to the idea of a Smart Factory, which allows the (automatic) collection and distribution of information and knowledge to all work places based on physical events [2].

Our vision is that it should be possible to build an extended workflow management system using the results of the Smart Factory to automatically execute and control technical processes. This leads to the same amount of flexibility enterprises gained by introducing workflow management systems. Furthermore, technical processes are enabled to easily interact with the back office, bridging the gap between business and production ("business-production-gap") [3].

What is missing to make this vision a reality? First of all, current workflow technology needs to be able to handle information about the physical world. This kind of information is often referred to as **context**: "*Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves*." [4] An application or in our case, a workflow that considers context is called context-aware which [4] defines as: "*A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task.*" To achieve this, workflow meta-models should allow the modeling of context in workflows and the use of context to control the flow between activities. The next step is to enhance a workflow management system to use context information in workflows, i.e. to become aware of the corresponding process meta-models. For this, the workflow management system is coupled with a context-provisioning platform. Finally, the usefulness of this technology is evaluated in a prototype and real world scenarios.

The paper is organized as follows: Section 2 gives an overview of the state-of–the-art of the relevant topics. Section 3 describes an example using context-aware workflows to implement technical processes executed in production environments. It shows a technical process carried out in the manufacturing industry. For a better understanding of the example we briefly describe a suitable context-provisioning platform (the Nexus Platform [5, 6]) and describe its basic context access patterns. Section 4 introduces three basic concepts on how to integrate context information into workflows. Since it is not possible to implement these concepts using standard workflows, Section 5 shows how to extend a workflow language (WS-BPEL 2.0 [7]) to implement context-aware workflows. Finally, Section 6 concludes and discusses future directions of this vision.

2 Related Work

The Workflow Management Coalition [8] defines workflows as "The computerized facilitation or automation of a business process, in whole or part" [9]. That means the execution of workflows is fully computerized. In contrast, business processes describe the progress of work done in a company in a form that is not directly executable by workflow systems. Business processes are modeled using graphical notations. The current important graphical notations are UML activity diagrams [10], the Business Process Modeling Notation (BPMN) [11], and event-driven process chains (EPCs) [13]. None of them contain explicit elements for modeling context-aware processes. We use BPMN for the specification of the process models.

For the automated execution with a computer, the processes must be expressed as workflows using a workflow execution language. In the past, following different languages were proposed for workflow execution: BPML [14], XPDL [15], YAWL [16], and WS-BPEL [7]. Since WS-BPEL (or BPEL for short) became the standard in this area, we use it for the implementation and execution of our context-aware workflows. Note that BPEL also supports collaboration of humans with the BPEL4People extension [17] or vendor specific solutions. For this purpose, a task management system distributes work tasks to human participants. Workflow systems (called BPEL-engine) such as IBM WebSphere Process Server, Microsoft BizTalk Server, etc. execute the BPEL workflow models.

There is no workflow language available for modeling context-aware workflows; however, some location aware approaches exist: xBPEL [18] is a BPEL extension for modeling mobile participants in workflows. The PerCollab system executes xBPEL and allows integration of people into BPEL workflows without constraining the users to their desktop PC. The WHAM System [19] supports mobile workforce and applications in workflow environments based on IBM products. These solutions are not generic enough to enable modeling of context-aware workflows, since they use only one type of context data: the location. However, other context data types such as time or identity of an object are also important for context-aware workflows.

Although context changes could be regarded as exceptions to the normal workflow and handled with database triggers and event-condition-action rules like in [32], we consider context as the normal case for workflows in real, physical environments. It is an integral part of the process and should be represented as first class citizen in the process modeling.

The main problem of today's workflow languages is that the execution context of a workflow has no direct influence on the design of the processes. However, it is an important requirement that a process reacts flexible to context changes and adapts to a changed environment.

2.1 Context-aware Information Systems

In the domain of context-aware applications, context models serve as an abstraction layer between applications and the technical infrastructure that provides the context data (e.g., sensors, databases, or other event sources). Typically, these models are managed in some kind of infrastructure to relieve the often small and mobile applica-

tions from that task and to share context information between applications. Also, a rather loose coupling between context management and the applications eases the maintenance and evolution of the whole system and again, provides more flexibility. In the past years, research and development of context management systems was quite active: there are rather simple, widget-like frameworks for sensor information like the Context Toolkit [21]; support for smart environments like Aura [22] or Gaia [23]; development frameworks with a separation between lower-level context (observation and facts) and higher-level context (information derived with rules) [24]; up to federated context-management platforms aiming at a efficient provisioning of context information within a global scope like the Nexus Platform [6] (to name a few). The latter supports the necessary basic access patterns to context information. Therefore, we describe Nexus in the next chapter in detail and use it as an example for our extensions without losing generality. This means that the binding to the Nexus Platform as context management system is only one possibility among others and therefore not fixed.

3 Prototype in the Smart Factory Production Environment

The example in Section 3.2 shows how context-aware workflows can improve efficiency of production processes, increase shop floor outcome through advanced labor division, and integrate production processes with business processes without media disruption.

The basis for context-aware workflows is context information such as the position of workers, tools, machines, transport carts, the stock amount of available spare parts, and also the state of all objects (i.e. the actual and maximal usage time of a tool). All these context data is available in the Smart Factory [2]. Context data is sensed via RFID tags mounted to the tools and Ubisense tags [25] carried by transport carts and workers. The context data is managed by a context management system developed in the Nexus project [26]. The analysis of this implemented system allows us to model the processes executed in the Smart Factory. It is not possible to implement the processes without usage of context information; hence the control flow is affected by context data. Context information drives the workflow forward by observing the state of the factory. If a worker has accomplished a task the workflow is able to detect that by observing the state of the environmental context. Upon that the workflow can proceed with the next workflow step automatically. The integration of humans into workflows is done using normal methods of workflow systems for human task management [1]. To deal with context the following extensions are needed: The selection of a worker (staff assignment) is based on his current position and the position where the task has to be executed. Furthermore, context information must be provided in the description of the human tasks. Thus, the workers don't have to search for the nearest tool to use, but rather get informed by the workflow about the exact position where they can find the tool.

The machine maintenance process is one of the Smart Factory processes and we use it as example in this paper. This process makes sure, that the tools used in the production machines are replaced before their remaining usage time runs out. We re-

alized the machine maintenance process in a prototype. In this prototype, the context influences and the integration of the workflow with the Nexus Platform were hard coded. It was cumbersome to make changes in this inflexible hard coded system. This showed us that modeling the prototype processes using workflows that are context-aware would be a better option. Thus, the prototype delivered the basic requirements for context-aware workflow modeling (see Section 4). In addition, we derived advanced requirements; their realization is described in Section 6 which summarizes future work. Section 3.2 presents a simplified version of the machine maintenance process containing all basic requirements. Since the Nexus Platform provides the context information in this example, we briefly describe the Nexus Platform in the next section.

3.1 Context Management with the Nexus Platform

The goal of the Nexus context management platform is to integrate various local context models to a common view for applications. The Nexus project [5] developed the platform over the last four years. It uses a federation approach and bases on the so-called Augmented World Model (AWM) that serves as a common, yet extensible integration schema [27]. The AWM is object-based and covers four different types of context information: geographical context (map data), dynamic context (sensor data), information context (documents and virtual information), and technical context (sensors, networks, devices, etc.). For the Smart Factory prototype, we developed a context model of the production floor that contained the location and the current status of machines, tools, and mobile agents. This domain-specific context model is an extension of the basic objects of the AWM, such as "SpatialObject" or "MobileObject". The data of the context model is updated by sensors and managed by the Nexus infrastructure. To interact with this infrastructure, the Nexus platform supports three access patterns:

- *Context Queries*: The Augmented World Query Language (AWQL) [28] can be used to declare a subset of the available context information. It supports object selection based on attribute values, spatial predicates, and attribute filtering. For efficient processing in distributed environments complex queries (like, e.g., spatial analyses or joins) are not supported. The result set of such a query is returned in the Augmented World Modeling Language (AWML) [28], an XML format that uses the Geographic Markup Language [29] for spatio-temporal representations.
- *Context Events*: Applications can register context events, like OnEnterArea (a mobile object enters a spatial area) or OnMeeting (two mobile objects meet in the same area) [30]. These events are observed by the platform, even if the corresponding context objects are managed by different systems. Note that this requires distributed observation algorithms. If the context event occurs, the Nexus platform notifies the application, thus enabling publish-subscribe-mechanisms.
- *Context Services*: Finally, context-aware applications can be deployed as value added services in the Nexus platform, like a Navigation Service (providing routing information) or a Map Service (providing results to context queries in a graphical representation). These services can be found and bound using standard web service technology described in [12].

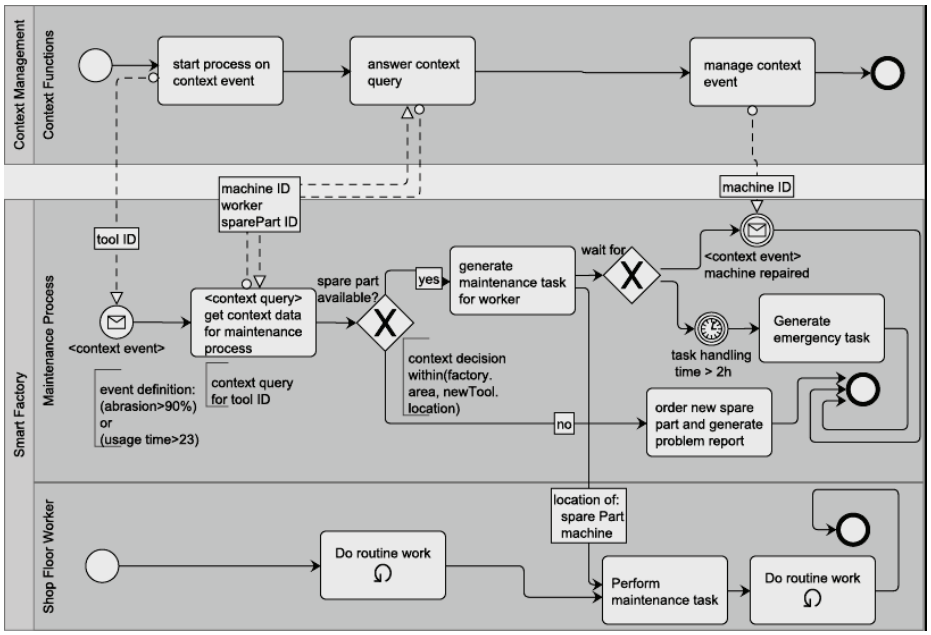

Fig. 1. Machine maintenance process modeled using BPMN [OMG06]

In the remainder of this paper we will show how these basic access patterns can be exploited to enable context-aware workflow modeling.

3.2 The Machine Maintenance Process in the Smart Factory

The process presented in Fig. 1 models a simplified version of the maintenance of a machine in the Smart Factory. For simplicity, we only consider tools mounted to machines as objects of maintenance. The state and position of the tools are monitored in real-time. When the residual term of a tool falls below a limit, the tool has to be replaced by a new one. In that process following context information is needed: tools in the factory (abrasion, position, type); position of all workers; position of the machine to be maintained.

The maintenance process is started by a **context event** that is triggered as soon as the *usage time or abrasion of a tool* exceeds a given limit. Then all necessary context data about the actual environment is queried (**context query**) and stored into internal variables. After this, branching based on context is done (**context decision**). If a spare part is available within the factory area, a worker gets the task to pick it up. Therefore, the worker is provided with all data necessary: e.g. *the exact current location of the spare part*. This is necessary if there are different storage locations or if somebody has misplaced it. Then the process waits for two hours for a **context event** signaling the successful *repair of the machine*. If the event does not occur in time, an emergency task is generated. If no spare part was available in the previous branching a new one is ordered. To purchase a new spare part a business process of the purchasing

department is started. This is done without media disruption because both technical and business processes are executed using BPEL workflows.

The context acquisition and management in this example is not simple. The locations of all tools, workers, and machines must be tracked, and their states must be monitored. Events must be observed and queries must be answered rapidly. These requirements show that a context management system, like the Nexus Platform, is needed.

4 Basic Concepts for Modeling Context-awareness

The example in Fig. 1 shows the three new concepts we introduce to enable modeling of context-aware workflows: context events, context query, and context decisions. Context events and context query base on the corresponding Nexus access patterns. Context decision is a workflow specific concept.

4.1 Context Event

Context events allow a workflow to wait asynchronously for a special environment state. The listening for the event is done in parallel to the normal workflow. As soon as the environment is in the awaited state, the context management system produces an event notification. After that, the workflow system delivers the event notification to the concrete workflow instance. By defining event handlers, the workflow can react on the occurring event in a predefined way. A context event is defined by an event description language and must be registered at the context management platform for observation. This results from the infinite amount of possible events occurring in the environment. Only registered events are observed by the context management system. In doing so, the information overflow can be handled. In the Smart Factory example, the process waits for the machine to be repaired. This is signaled by an event waiting for the state change of the machine's "isWorking" attribute. As soon as the machine got repaired, the attribute's value is changing and the event is thrown. For instantiation of the process, also a context event is used. This event is specified on all tools available in the factory and specified to observe if any tool has to be replaced.

Context events should firstly be used if a running process wants to wait for something to happen in the real world. Secondly, context events can be used to start new processes as a result of happenings in reality.

4.2 Context Query

A context query allows synchronous query oriented access of context data with a query language. The query language has to support object selection based on spatial predicates and filtering of the results. In the example, a context query is used to get the position of the worker, spare part, or the state of a machine in the Smart Factory. The objects of interest are queried at the context management platform and are in-

jected into the internal workflow data. The attribute values of the context objects can now be used like other complex workflow variables.

A context query should be used if the process needs to read context data at a well-defined position in the process. The workflow modeler models the required context data explicitly by specifying a context query selecting only the needed objects.

4.3 Context Decision

A context decision allows the process to route process control flow based on context data using context-aware operators. The context data used by the context-aware operators can be internal process data or can be gathered by a context query from process external sources. The context-aware operators provide comparison functions for all types of primary context. Primary context is location, time, and identity [6]. One comparison function is the *within* function evaluating whether the location coordinates of an object are inside a given area. Each context decision is evaluated to true or false and thus can be used for branching in processes.

The maintenance process in the Smart Factory example uses a context decision to evaluate an exclusive or fork. That context decision evaluates if a tool object of the needed type can be found inside the factory area. If a tool is found, it is used as spare part. Otherwise, a new tool must be ordered and delivered to the factory by an external supplier.

5 Implementation of Context-aware Workflows using BPEL

This section shows how the three context extensions are implemented using a standard workflow execution language. We use WS-BPEL 2.0 [7] and its extension mechanisms to allow modeling of context-aware workflows. For that purpose we introduce new BPEL activities. We call this extended BPEL version **Context4BPEL**. The intention of Context4BPEL is the explicit modeling of context influences on workflows. This extension approach allows us to achieve a direct visibility of context influences on workflows. Besides this extension approach one other implementation method would be to provide a context provisioning web service and use it with standard BPEL invoke activities.

We decided to use the *extension approach*, because it has the *following advantages* compared to the *web service approach*: Firstly it is directly visible in the workflow model where context is used and how it affects the control flow. In contrast the web service approach would hide the context usage behind standard invoke activities. Secondly, the implementation of extended context activities allows the workflow engine to do optimization, as the engine knows the semantics of the new activities and not only invokes web services. Thirdly, the advanced concepts mentioned shortly in Section 6 can only be implemented with the extension approach. Therefore, the extension approach allows us to stay consistent in the future development. Overall, the extension approach enables the separation of concerns between control flow modeling and the modeling of context influences.

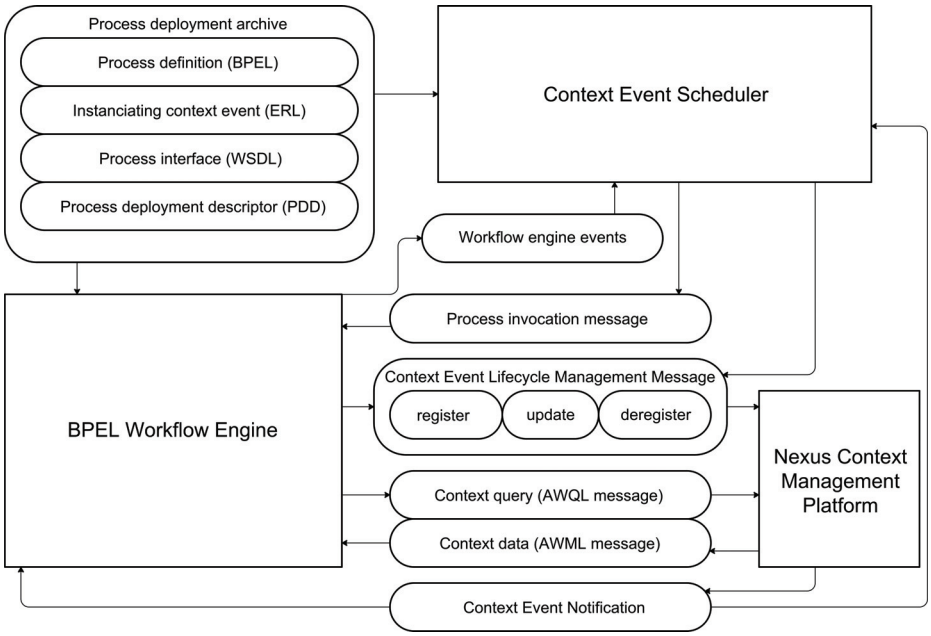

Fig. 2. Architecture of the Context4BPEL execution environment

For receiving asynchronous messages from the context management platform, we define a standard partner link "ContextPlatformLink", where such interactions take place, since existing inbound message activities (receive, pick, event handler) can be used for this task.

The proposed context extensions can be implemented using different context management systems. We opted for the Nexus Platform, because it supports all required access patterns.

5.1 Architecture

Fig. 2 shows the architecture of the Context4BPEL execution environment using FMC notation [20]. A rectangle denotes an active component and an oval is denoting a passive component containing data. The active components are the BPEL Workflow Engine, the Context Event Scheduler, and the Nexus Context Management Platform.

The BPEL Workflow Engine executes the workflows and generates an event as soon as a new workflow is installed form the Process deployment archive. The Context Event Scheduler reacts on that event and checks if the newly installed workflow contains a document specifying a context event via Event Registration Language (ERL). The ERL is a language to register events in the Nexus platform. For a detailed description and examples of the ERL please refer to [28]. If an ERL is provided the new workflow can be started by a context event (see 5.2: Instantiation of Workflows through Context Events). Therefore this event must be registered at the Nexus Context Management Platform, responsible for observing context events. As soon as the

registered context event occurs, the Context Event Scheduler makes sure that the corresponding workflow is started. A running workflow instance can directly register context events (see 5.2: Workflow controlled Context Events) and query context at the Nexus Platform.

In the following, we describe the two different kinds of context events, how context can be queried, and how context decisions on both internal and external context data can be made.

5.2 Context Event

Context events can be distinguished in two types: (i) Workflow instantiation events and (ii) Workflow notification events. The first type is controlled externally by the Context Event Scheduler. The function of this type of context events is the initiation of new workflow instances. The second type of context events is controlled by the running workflow instances. Their function is the event-based navigation of the internal process control flow. In the following, their implementation is described.

(i) Instantiation of Workflows through Context Events. Typically, workflows are started by user actions or direct invocation. For context-aware workflows, it must be possible to start them by the occurrence of a defined context state. To implement this, an ERL document containing the specification of the spatial event that should cause the instantiation of the workflow must be provided. This document is added to the process deployment archive. After the deployment of the process, the Context Event Scheduler uses the ERL document to register a spatial event at the Context Management Platform. As soon as the event is fired, the process instantiation operation is called. To identify the endpoint of the operation the process deployment descriptor (PDD) is used (see Fig. 2). The operation takes the event notification message as input. Thus, the process knows from the start what spatial event and which concrete spatial object caused the process instantiation.

Example for a workflow instantiation caused by a context event:

```
<variable name="contextEventData"
          messageType="nexus:ENL"/>
<receive  partnerLink="ContextPlatformLink"
          operation="instantiateOnContextEvent"
          variable="contextEventData"
          createInstance="yes" />
```

(ii) Workflow Controlled Context Events. Workflow controlled context events have to be registered and deregistered by the workflow itself. To allow a workflow the management of context events following new BPEL activities are introduced. The registration of a spatial event is done with **registerSpatialEvent**. As input data an ERL message must be given. The output of the activity is an **event id** stored in the given variable. The event id is important for a unique identification of an event and is needed to specify events to be changed or deregistered. The update of a context event is done with the activity **updateSpatialEvent**. As input data, the update needs an event update message. This message contains the event id identifying the event. De-

registration of the event only needs the event id as input variable and can be executed by the activity **deregisterSpatialEvent.** These extensions are for the **management of the lifecycle** of spatial events.

Context4BPEL activities for the lifecycle management of context events:

```
<c4b:registerSpatialEvent variable="eventID">
  <nexus:eventRegistrationMessage>
    <nexus:erl> ERL-message </nexus:erl>
  </nexus:eventRegistrationMessage>
  <correlations ... />
</c4b:registerSpatialEvent>

<c4b:updateSpatialEvent>
  <nexus:eventRegistrationMessage>
    <nexus:erl> ERL-message </nexus:erl>
  </nexus:eventRegistrationMessage>
</c4b:updateSpatialEvent>

<c4b:deregisterSpatialEvent eventID="eventID">
</c4b:deregisterSpatialEvent>
```

The **notification of occurrence** of events is done with any BPEL inbound message activity. Inbound message activities are pick, receive, and event handlers. The inbound message activity must take the event notification (ENL) as an input value. Note that the workflow modeler has to take care about the correlation of the asynchronous messages.

Example for receiving a notification of a context event:

```
<variable name="contextEventNotification"
          messageType="nexus:ENL"/>
<pick>
  <onMessage partnerLink="ContextPlatformLink"
          operation="incomingContextMessage"
          variable="contextEventNotification">
    <correlations ... />
  </onMessage>
</pick>
```

5.3 Context Query

With a context query, a workflow can retrieve objects in the context model. The new BPEL activity **queryContext** implements that functionality. The workflow has to wait until the execution is completed and a result is available to work with. Context queries are expressed in AWQL (Augmented World Query Language), the resulting objects of context queries are serialized in AWML (Augmented World Modeling Language) [28]. Note that in an AWQL or AWML message definition data of process variables can be accessed using $varname.

Definition of variables for storing context query results:

```
<variable name="contextData"
          messageType="nexus:AWML" />
```

Context4BPEL activity for context data retrieval:

```
<c4b:queryContext outputVariable="contextData">
  <c4b:ContextQueryMessage>
    <nexus:awql> AWQL-message </nexus:awql>
  </c4b:ContextQueryMessage>
</c4b:queryContext>
```

5.4 Context Based Transition Conditions (Context Decision)

Context data can be used in transition conditions. It is possible to use (i) replicated internal context data stored in a workflow variable or (ii) up to date external context data stored in the context management system. The following describes both types of context decisions. Note that, for both kinds of context based transitions it is not necessary to extend the BPEL standard. Instead, we define new functions for the XPath expression language [31] and implement them using the extension concepts of XPath.

(i) Context Operator Based Decisions on Internal Context Data. To use context data to evaluate context based workflow transitions context data must first be available in a workflow variable. Then a context operator can use the data for evaluation of transition conditions. For the realization of these operators, we define new context-aware XPath functions. The Smart Factory example (see Fig. 1) uses the function *context:within(area, location)* to detect whether a new tool is available on the factory area. It is possible to define other context functions for location, time, and identity in the same way.

Definition of variables containing context data:

```
<variable name="factory"
          messageType="nexus:awml" />
<variable name="newTool"
          messageType="nexus:awml" />
```

Example for a context decision on internal context data:

```
<source linkName="NCName">
  <transitionCondition>
     c4b:within($factory/area, $newTool/location)
  </transitionCondition>
</source>
```

(ii) Query Based Decisions on External Context Data. To base the transition condition on external context data, a context query is used. The query specifies a situation that is used to evaluate the transition condition. That means if the situation exists at the moment of the decision, the transition condition is true. As immediate consequence, the workflow can proceed with the next activity. The situation specification is similar to a context query and can express complex context states. The evaluation al-

gorithm checks whether the result of the query is empty and returns the corresponding Boolean value. For example, consider a query checking whether a worker is near a given machine. The function **awql:getSituation** evaluates the query and returns a corresponding Boolean value.

Definition of a variable containing a situation evaluation query:

```
<variable name="workerNearMachine"
          messageType="nexus:awql" />
```

Example for a transition condition based on external context data:

```
<source linkName="NCName">
  <transitionCondition>
     c4b:getSituation($workerNearMachine) = true()
  </transitionCondition>
</source>
```

6 Conclusion and Outlook

We introduced concepts for modeling context-aware workflows and showed how to realize them extending BPEL and using the Nexus Platform. Our motivation to develop modeling concepts for context-aware workflows resulted from an early prototype of the context-aware machine-maintenance workflow in the Smart Factory. This prototype showed that there is a great need for a generic workflow-based control of technical processes. Normally, technical processes are planned using a production planning system (PPS) and are afterwards implemented in the assembly line layout. At the production time, a standard workflow system cannot support the technical processes. For example, a standard workflow system is not aware of the location of the tools, workers, and the machine states. However, that context information is available in context-aware workflows and the technical processes can be adapted to changed environments automatically.

In addition, we learned from the prototype that using context-aware workflows to implement context-aware application is a promising approach. Mainly, the modeling of the context-aware workflows was easier and faster than the manual programming of a context-aware application. Furthermore, the basic workflow activities used to model context-aware workflows are qualified for easy reuse in new projects because of their well-defined interfaces and functionality.

As future work, we plan to research if the defined concepts are sufficient for the modeling of context-aware workflows or if further concepts are needed. Such further concepts can be (i) a context scope and (ii) a transparent context query. A context scope assures that a defined context is in effect during the execution of activities belonging to a context scope. A transparent context query guarantees a certain freshness of context data without an explicit execution of a context query by the workflow.

Furthermore, our experiences are encouraging the implementation of a generic prototype for a context-aware workflow system supporting Context4BPEL. In addition, we plan to make an evaluation of the prototype with a broad range of technical processes as future work.

Acknowledgments

We are grateful to Dieter H. Roller and Andrea Wöhr for a detailed review of an earlier version of this paper.

The Nexus project was funded 1999-2002 by the German Research Association (DFG) under the grant 200989 and is continued as collaborative Research Center (SFB) 627 since 2003.

References

All links were followed on 2007-04-24.

1. F. Leymann; D. Roller: *Production Workflow - Concepts and Techniques*. PTR Prentice Hall, 2000.
2. E. Westkaemper; L. Jendoubi; M. Eissele; T. Ertl: *Smart Factory – Bridging the gap between digital planning and reality*. Proceedings of the 38th CIRP International Seminar on Manufacturing Systems, 2005
3. M. Wieland; F. Leymann; L. Jendoubi; D. Nicklas; F. Dürr: *Task-orientierte Anwendungen in einer Smart Factory*. 1. Konferenz Mobilität und Mobile Informationssysteme (MMS 2006), Lecture Notes in Informatics (LNI), P-76, 2006 *(in German)*
4. A. K. Dey: *Understanding and Using Context*. Personal Ubiquitous Computing, 5(1), pp. 4-7, Springer-Verlag, 2001.
5. Project Homepage, *SFB 627: Nexus Project*, http://www.nexus.uni-stuttgart.de
6. M. Grossmann; M. Bauer; N. Hönle; U.-P. Käppeler; D. Nicklas; T. Schwarz: *Efficiently Managing Context Information for Large-scale Scenarios.* In: Proceedings of the 3rd IEEE Conference on Pervasive Computing and Communications (PerCom) 2005
7. OASIS, Web *Services Business Process Execution Language Version 2.0*. http://docs.oasis-open.org/wsbpel/2.0/CS01/wsbpel-v2.0-CS01.html
8. *Workflow Management Coalition* (WfMC), http://www.wfmc.org
9. D. Hollingsworth: *The Workflow Reference Model*. Workflow Management Coalition
10. Object Management Group: *Unified Modeling Language: Superstructure*. Version 2.0. formal/05-07-04 http://www.omg.org/docs/formal/05-07-04.pdf
11. Object Management Group: *Business Process Modeling Notation*. Final Adopted Specification. http://www.omg.org/cgi-bin/doc?dtc/2006-02-01, February 1st, 2006
12. S. Weerawarana; P. Curbera; F. Leymann; T. Storey; D. Ferguson: *Web Services Platform Architecture*. Prentice Hall, 2005
13. A.-W. Scheer; O. Thomas; O. Adam: *Process Modeling Using Event-Driven Process Chains*. Wiley-Interscience, 2005.
14. A. Arkin: *Business Process Modeling Language*. Version 1.0. http://www.bpmi.org, November 2002.
15. Workflow Management Coalition: *Workflow Process Definition Interface–XML Process Definition Language*. Document Number WfMC-TC-1025, October 25, 2002, Version 1.0.
16. W.M.P. van der Aalst; A.H.M. ter Hofstede: *YAWL: Yet Another Workflow Language.* Information Systems , 30(4):245-275, 2005
17. IBM, SAP AG: *WS-BPEL Extension for People*. http://www-128.ibm.com/developer works/webservices/library/specification/ws-bpel4people/
18. D. Chakraborty; H. Lei: *Pervasive Enablement of Bussiness Processes*. In: Proceedings of the Second IEEE International Conference on Pervasive Computing and Communications (PerCom), 2004

19. J. Jing; K. E. Huff; B. Hurwitz; H. Sinha; B. Robinson; M. Feblowitz: *WHAM: Supporting Mobile Workforce and Applications in Workflow Environments.* RIDE 2000: 31-38
20. A. Knopfel; B. Grone; P. Tabeling: *Fundamental Modeling Concepts: Effective Communication of IT Systems*. Wiley, 2006
21. A. K. Dey; G. D. Abowd; D. Salber: *A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-aware Applications*. In: Human-Computer Interaction, Vol. 16, No. 2-4, S. 97-166. Lawrence Erlbaum Associates, Inc., Mahwah, New Jersey, USA. 2001
22. G. Judd; P. Steenkiste: *Providing Contextual Information to Pervasive Computing Applications.* In: Proceedings of the First IEEE International Conference on Pervasive Computing and Communications (PerCom), 2003
23. M. Roman; R.H. Campbell: *GAIA: Enabling Active Spaces*. Proceedings of the 9th ACM SIGOPS European Workshop, Kolding, Denmark, 2000
24. K. Henricksen; J. Indulska: *A Software Engineering Framework for Context-aware Pervasive Computing*. Proceedings of IEEE International Conference on Pervasive Computing and Communications (PerCom), 2004
25. Precise Real-time Location System, *Ubisense*, http://www.ubisense.de/
26. M. Bauer; L. Jendoubi; O. Siemoneit: *Smart Factory – Mobile Computing in Production Environments.* In: Proceedings of the MobiSys 2004 Workshop on Applications of Mobile Embedded Systems (WAMES 2004)
27. D. Nicklas; B. Mitschang: *On building location aware applications using an open platform based on the NEXUS Augmented World Model*. In: Rumpe, Bernhard (ed.); France, Robert (ed.): Software and Systems Modeling. Vol. 3(4), Berlin Heidelberg: Springer-Verlag, 2004
28. M. Bauer; F. Dürr; J. Geiger; M. Grossmann; N. Hönle; J. Joswig; D. Nicklas; T. Schwarz: *Information Management and Exchange in the Nexus Platform*. Universität Stuttgart, Technical Report No. 2004/04
29. S. Cox; A. Cuthbert; R. Lake; R. Martell (Eds): *Geographic MarkUp Language (GML 2.0)*. OpenGIS® Implementation Specification, OGC Document Number: 01-029,20 February 2001, http://www.opengis.net/gml/01-029/GML2.html
30. M. Bauer; K. Rothermel: *An Architecture for Observing Physical World Events*. In: Proceedings of the 11th International Conference on Parallel and Distributed Systems: ICPADS 2005; Fukuoka, Japan, July 20-22, 2005.
31. W3C: *XML Path Language (XPath)*, Version 1.0, http://www.w3.org/TR/xpath
32. F. Casati; S. Ceri; S. Paraboschi; G. Pozzi: *Specification and Implementation of Exceptions in Workflow Management Systems.* ACM Transactions on Database Systems 24(3), 1999