REQUIREMENTS FOR CONTEXT-AWARE, MOBILE WORKFLOW SYSTEMS

Carl-Fredrik Sørensen
carlfrs@idi.ntnu.no

Alf Inge Wang
alfw@idi.ntnu.no

Jon Ole Nødtvedt
nodtvej@frisurf.no

Man Hoang Nguyen
manhoang@pvv.ntnu.no

Department of Computer and Information Science, Norwegian University of Science and Technology (NTNU),
N-7491 Trondheim, Norway.Phone: +47 73 590731, Fax: +47 73 594466, http://www.mowahs.com

ABSTRACT

This paper describes novel work on context-aware workflow systems envisaged to integrate and utilise context information from pervasive environments to improve the enactment of mobile work processes. Work in dynamic, mobile environments requires workflow systems to be capable of utilising contextual events to be used in process enactment and coordination. The existing standards in workflow systems are investigated and extended with new requirements to cope with context-awareness. The concept of a *smart work process* is used to capture adaptive and context-aware process support. The requirements and design considerations for developing context-aware workflow systems are elaborated and then used to propose a new interface for handling context-awareness in workflow enactment services. This interface will serve as a link between the work environment and the workflow system.

KEYWORDS: Smart work process, mobility, context-awareness, adaptation, requirements.

1 INTRODUCTION

The evolution of wireless networks, mobile devices, and sensor networks together with the instrumentation of the physical environment bring the vision of ubiquitous computing closer to reality [15, 23]. The evolution enables exploration in many application areas involving both static and mobile actors. The MOWAHS project[1] investigates how process support can be achieved for heterogeneous devices to provide a flexible work environment where mobile work processes can be executed and shared together with artefacts belonging to these work processes.

Mobile work and the local working environment can mutually influence each other in different ways. Firstly, mobile work can change the state of its environment. Secondly, the environment state or change of state can influence activity control by: Initiating activities to be performed; controlling start, duration, delay, stop, and termination of activities; controlling which activities to be performed; changing the content or goal of an activity; and initiating exceptions in the current activities. Further, activities and change of state in the environment can together provide dynamics to reach process goals by adjusting to each other. The mobile environment can thus be perceived as a pervasive computing environment where context-awareness plays a central role. In [18], we discuss and differentiate mobile work from nomadic work, and then identifies issues related to how different types of context information can be supported and utilised in systems to support mobile work processes. We further introduced the notion of the *smart work process*; i.e., a work process that automatically adapts to changes in the environment through reasoning on context information.

The adoption of mobile computerised equipment gives the opportunity to extend existing workflow systems with support for mobile work processes and activities. Such support requires a dynamic behaviour not previously found in workflow systems. Workflow systems have generally only supported minimal dynamic behaviour. We therefore propose to extend workflow systems to also integrate and utilise context. It is therefore important to study how workflow systems can integrate and utilise context information in their support of process and activity enactment.

We have identified some challenges in creating context-aware workflow systems, e.g.: Integration of context information that is usable for process reasoning and enactment; specification of rules and components necessary for a workflow system to be reactive to context changes before and during workflow process execution; management of context exception states; usage of context information to enable more dynamic process execution with regards to selection of alternative process paths; support of situated actions and situated process reasoning; accountability of situated process and activity enactment; selection and use of relevant context sources related to the current process plan; and how to enable workflow systems to work as coordinators for cooperative situated activities.

The rest of the paper is organised as follows: Section 2 briefly presents the concept of workflow, the workflow reference model, and how exceptions are treated in workflow systems, Section 3 presents requirements for a context-aware workflow system to support smart work processes, and Section 4 briefly considers a design to fulfil some of the requirements. Further, Section 5 discusses issues related to integration of context into workflow systems, and Section 6 presents related work. Finally, Section 7 concludes the paper.

[1]This paper is part of the MObile Work Across Heterogeneous System (MOWAHS) project. The MOWAHS project was sponsored by the Norwegian Research Council's IKT 2010 programme.

2 WORKFLOW – CONCEPTS AND MODELS

Workflow technology [6] has been around for some time. Java, XML, and the World Wide Web have made it easier to define and exchange information across applications. The growth of the Web has also had a major impact on workflow, establishing a ubiquitous platform to interact and participate in workflows. Central concepts in workflow are the notions of process and activities, process enactment, exchange of information across applications, and coordination of activities and resources among the workflow participants.

An activity can be described in a *process model* by providing goals, pre- and post-conditions, invariant conditions, and the use or production of artefacts and resources [12]. Context and changes in context may affect process model elements in several ways: *Can be used as preconditions to start/stop/terminate an activity*; *provide rules for how and what to do in an activity*; *provide alternative process paths*; *trigger or create new activities*. Further, the actual work or actuations started by systems can result in changes in the physical environment (post-condition), preconditions to perform both new and (pre- or situated) planned activities, and preparation of the environment to establish specific preconditions for specific activities.

The paragraph above identifies how a process enactment service must take into account the surroundings to effectively support the enactment of mobile work processes. The process model is in addition to the process plan and goals, also affected by state changes of elements in both the physical and the computational environments. The elements in the environment may be affected by the process, but also by other factors that might be out of control from the enactment service point of view. The environment may, e.g., contain other "competing" actors, artefacts, resources and other elements causing coordination problems between actors, the physical and computational environment itself, and the process goals of the different actors.

Allen [1] segments workflow into *production*, *administrative*, *collaborative*, and *ad hoc* workflow, based on specific aspects in each of these segments. The *ad hoc workflow* features easy process definition and flexibility to make it easy for users to adapt to changing circumstances. Users own their own process, which separates this type of workflow from process workflow where the organisations own the processes.

The ad hoc workflow systems are best suited for the dynamic behaviour expected in pervasive environments. However, even an ad hoc workflow system is not capable of handling the dynamic behaviour in pervasive work environments.

Below, we will briefly present the workflow reference model. The reference model is used to identify eventually discrepancies with respect to making a workflow system context-aware.

2.1 WORKFLOW REFERENCE MODEL

The workflow reference model is developed by the Workflow Management Coalition (WfMC) [12] and identifies the interfaces and architecture within a generic workflow product structure. The reference model consists of a set of components and interfaces:

- *Workflow Enactment Service*: A software service that consists of one or more workflow engines to manage and execute particular workflow instances. The workflow management service may be centralised or functionally distributed.
- *Workflow Engine*: A software service that provides the run-time execution environment for a process instance. A workflow engine may be responsible for the whole runtime process execution, but also for only a part of it.
- *Workflow Application Programming Interface & Interchange Formats (WAPI)*[19]: WAPI incorporates specifications to enable interoperability between different components of workflow management systems and applications.
- *Interface 1: Process Definition Tools*: The interface defines a common interchange format which allows different tools to share process definitions and exchange information. The interface covers standard definitions and the interchange of such information as: Process start and termination conditions; Identification of activities within a process; Identification of data types and access paths; Definition of transition conditions and flow rules; and Information for resource allocation decisions.
- *Interface 2: Workflow Client Application*: Defines standards for workflow engines to maintain work items that a workflow client presents to the user.
- *Interface 3: Invoked Application*: Defines a standard interface allowing a workflow engine to invoke a wide variety of applications.
- *Interface 4: Workflow Interoperability* Defines a standard interface to allow workflow management systems developed by different vendors to pass work items between each other across a network.
- *Interface 5: Administration & Monitoring Tools*: Defines a standard interface to allow the activity status monitoring application of one vendor to work with other workflow enactment services.

2.2 EXCEPTION HANDLING

Exception situations arise in workflow systems as in any other computer system. It is therefore necessary to study how exceptions in a workflow system without human intervention can be handled. Exception transitions are by WfMC defined at the same level as normal transitions. Thus, transitions defined as exceptions are handled by the workflow enactment engine like any other transition.

There are also other ways of handling exceptions in workflow systems. Exceptions can be categorised into the two main types; expected and unexpected exceptions [8]. These exceptions are both to be handled by the exception handler of the workflow system. Expected exceptions are usually known in advance and can thus be modelled a priori. Expected exceptions can then be handled by the exception handler using the semantics of the workflow system. The exception handler usually uses some form of reactive processing to cope with these exceptions.

Unexpected exceptions are unpredictable, asynchronous and may require special treatment. This makes these exceptions hard to represent in workflow process definitions. Unexpected exceptions may often not be identified and will therefore usually require human intervention.

3 REQUIREMENTS FOR WORKFLOW SYSTEMS UTILISING CONTEXT INFORMATION

This section will present some of the requirements for workflow systems integrating and utilising context information. Basic workflow system requirements will not be presented in this paper, but generally the workflow system should adhere to the workflow standards as presented by the WfMC [19]. Figure 1 illustrates actors and use cases in a basic workflow system. The actors are the workflow enactment service, the workflow client application and the workflow participant.

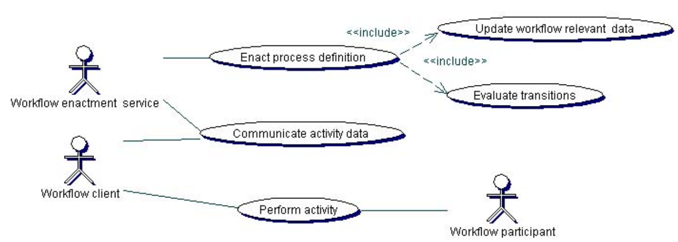

Figure 1. Use case diagram for a basic workflow system

The list below summarises the requirements for a basic workflow system:

BR1 Adhere to the WfMC standards.

BR2 Interpret and enact process definitions specified in the XPDL language.

BR3 Send and receive activities.

BR4 Evaluate transitions.

BR5 Update workflow relevant data, based on completed activities from workflow participants.

BR6 Perform concurrent enactment of processes.

3.1 CONTEXT REPRESENTATION AND RETRIEVAL

An underlying context support system is an important part of any context-aware application. The main role of such a system is to provide access to context information in the pervasive environment for context-aware applications. The design principle, *separation of concerns* between the underlying context sources and the users of such information, gives a requirement to remove low-level input handling of context information from the context-aware application. Further, is it also considered natural to provide for some level of context interpretation and aggregation in the context support system [7]. The requirement for removal of low level input handling of context information is necessary to ensure reuse of context handling code across applications. Different workflow processes, workflow clients, and other kinds of applications may thus make use of the same context sensors. Likewise, reuse of functionality related to context interpretation and aggregation is considered important. It is, however, necessary to be aware of the difficulties of presenting users with information of how the system reasons about and takes actions on behalf of the user. Bellotti et al. [3] refer to the importance of accountability and intelligibility of context-aware systems that is considered threatened when the users are not able to access and control information used in actuations. Nevertheless, reuse of context interpretations is considered superior in our case to the need for presenting users with the "reasoning" behind the actions performed. Most of the aspects related to the execution of workflow processes are done automatic and autonomously by the workflow engine. The gain of using workflow systems will be much lower by imposing manual activities more than strictly necessary. In addition, most of the workflow processes are already planned sequences of activities, making the addition of context reasoning in a workflow just another set of actions to be planned by the workflow designer. Workflow transitions are already based on rules and conditions that can be regarded as context information. However, existing workflow systems are not at the moment enabled to handle the dynamic contextual information caused by mobility.

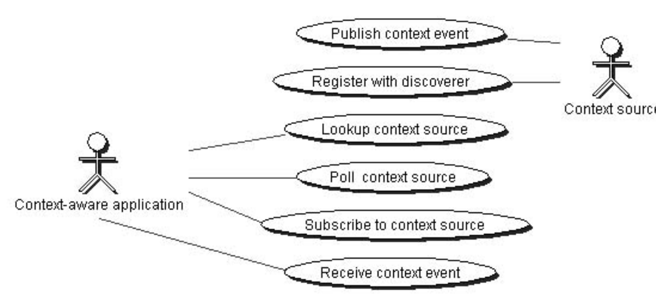

Figure 2. Use case diagram for context framework and context-aware application

Figure 2 shows the actors and use cases involved in using a context information framework. The context source provides the context information. The context-aware application uses the services of the other actors.

The list below summarises the requirements for context information representation, interpretation and retrieval:

CR1 Provide a discoverer service for identifying available context sources.

CR2 Provide access to context sources using abstractions/mediators to remove low-level input handling from the workflow system (separation of concerns). Sensor abstractions can also be used to provide limited support for context fusion, aggregation and interpretation.

CR3 Support both polling and publish/subscribe mechanisms for context retrieval.

3.2 CONTEXT-AWARE FUNCTIONALITY IN A WORKFLOW SYSTEM

This section will present and discuss requirements related to the integration and use of context information in a context-aware workflow system. Context information can be used both in the workflow enactment service and in the workflow client.

3.2.1 Context-Aware Workflow Enactment Service

A context-aware workflow enactment service should have the ability to use context information in the processing of *workflow transitions*. This is possible using the current standards. However, the modelling of context source lookup, polling/subscription, and the conversion of context information from a context information representation to a workflow representation, must be added to the existing workflow planning environment.

Another important functionality is to make the workflow system able to respond to contextual changes. Contextual events can be used to initiate processes, and thus be considered as *preconditions*. Since activities may influence the environment and the environment changes over time, contextual events may also be used to terminate the processes, and thus be considered as *post-conditions*. Context events may also invalidate previous satisfied conditions for the current process path. In an invariant scenario where a certain contextual state is to be kept to perform an action, it is necessary for the enactment engine to revalidate the current process path or choose another. The events must further be directed to the correct processes, initiating execution of appropriate responses.

Contextual changes may also open new process paths, which earlier in a process enactment had been discarded. A workflow enactment service should therefore be capable of finding other ways to finish a process in a certain contextual state, if the current process path does not lead to the process goal.

The pervasive environment may not always be in a defined state, and the context support system has therefore problems to report the actual state of the environment. This situation may require the workflow system to decide the contextual state with more elaborate means or the situation may require human intervention. A workflow enactment service must therefore be able to handle such exceptions and provide solutions for managing these situations.

Application of activity theory [9, 21] including situated planning [2] introduce important considerations of the behaviour of a context-aware workflow system. Likewise, situated actions [17] is important to consider since the environment will always be used to adapt the actions and operations. In addition to the pre-planned workflow enactment already present in the WfMC specification, unplanned workflow process enactment should also be supported. Unplanned workflow is more challenging since it is at the moment not part of any specification. The ad hoc construction of workflow processes based on the environmental state/context is regarded as necessary functionality for a context-aware workflow enactment service. Such functionality can be based on specified rules for context state evaluation by linking certain state(s) with one or more template processes. Another possibility is to make the pervasive environment capable of returning processes using e.g. augmented artefacts [16]. The process goals or the preferred environmental state can e.g. be communicated to the environment from the workflow enactment service. The environment can then adjust to ascertain the specified state. The pervasive environment may also provide process goals based on its own perception of the current state. The new process should become part of the overall process definition to allow the context-aware workflow system to work as a "technology of accountability" [2].

Figure 3 illustrates the requirements for a context-aware workflow enactment service.

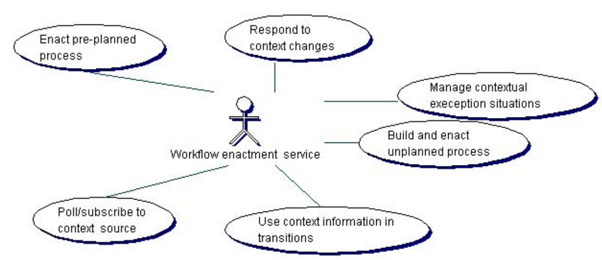

Figure 3. Use case diagram for a context-aware workflow enactment service

The list below summarises the functional requirements related to the use of context information in a workflow enactment service:

WR1 Context information should be used in the evaluation of workflow transitions.

WR2 The system should be able to define processes using both abstract and concrete context sources.

WR3 The system should be able to define how to access context information and convert the information to a workflow representation.

WR4 Responsiveness to contextual changes or events:

WR4.1 Ad hoc start of processes and activities.

WR4.2 Context as post conditions for processes.

WR4.3 Correct handling of invariant conditions.

WR5 Support revalidation of selected process paths if the current path does not lead to the process goal.

WR6 Exception handling of undefined contextual states.

WR7 Support both pre-planned and unplanned process enactment, by providing rule-based process definition by e.g. learning.

3.2.2 Context-Aware Workflow Client

The workflow clients have a relatively simple function according to the WfMC Interface 1 [19] specification. They receive activity definitions with related data, update the data, and return the activity with data and state back to the workflow enactment service. Supporting the relationship between workflow systems and activity theory means that we also have to enhance the functionality of the workflow client. When a workflow client receives an activity specifying work to be done at a remote location, activity theory argues that the planning and enactment are best done in situ. A workflow client should therefore benefit being able to itself enact a process definition. The workflow enactment service should of course be capable of processing this for the workflow client, but there are situations in mobile scenarios where communication is not possible or undesirable. It is impossible for a workflow planner to in advance know all possible contextual conditions an activity may be performed in. Rules must be provided together with the activity definition enabling the workflow client to interpret the environmental context. The mobile users may also want to have local control to adapt to the environment.

Activities to be performed in situ to solve an overall process can either be defined in advance if known, or be provided by the environment itself. A workflow client must thus be able to perform process enactment, access the context sources, and interpret the contextual state based on specified rules. The client must also be able to specify a process path based on activities received from the environment, or from the local or central workflow enactment service.

Activity coordination is another important aspect of workflow process enactment. It is possible to have several concurrent processes running in a workflow system. Sometimes these processes directly affect each other. In existing workflow systems, the coordination is handled by the workflow enactment service. In a process scenario using situated planning, coordination can be problematic. Individual situated activities performed by one workflow participant may function as a precondition for (situated) actions to be performed by another workflow participant. In this scenario, the workflow clients would work as context sources for each other. The rules for every activity to be performed in situ must therefore specify relations to other activities also performed in situ. The use of context information from another workflow client would not be different from the use of any other context source.

Workflow clients are better suited to monitor the environmental state to decide if the post-conditions for the current activities have been satisfied. This applies to both normal activities and situated activities. Workflow clients should therefore support enactment based on context monitoring.

Figure 4 shows the use cases for the workflow client and the participant using the client:

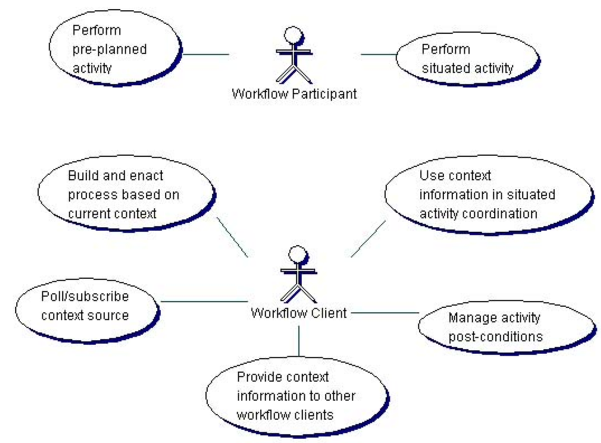

Figure 4. Use case diagram for a context-aware workflow client

The list below summarises the functional requirements related to workflow client-based context-awareness:

WCR1 Perform situated planning based on current contextual conditions.

WCR2 Perform situated activity coordination between workflow participants.

WCR3 Context as post-condition for activities.

3.3 MOBILITY REQUIREMENTS

Generally, a workflow system supporting mobility should try to fulfil the following requirements. These requirements are related to both the underlying technology as well as the workflow system itself:

MR1 Support for physical mobility and network mobility.

MR2 Support for unreliable communications.

MR3 Support for disconnected operations and asynchronous communications.

MR4 Activity locking with support for reassignment of activities.

MR5 Flexible task assignment support.

MR6 Support for session mobility.

MR7 User able to select the data elements to be transferred to the mobile device.

MR8 Device independence.

4 DESIGN CONSIDERATIONS

The main challenge presented in this paper is how context information can effectively be integrated in a workflow system and utilised in workflow enactment. External components are considered necessary to be able to acquire and use context information according to the WfMC standards. Dey et al. have presented a conceptual framework [7] that contains several abstractions for context information collection and transformation. A widget abstraction communicates directly with context sources such as sensors, and provides the lowest level of abstraction. A widget transforms low-level input from sensors into the context information representation that can be used by the system. This means that context widgets can provide context information to the workflow enactment service. The conceptual framework in [7] also uses a discoverer service that satisfies the requirement for locating distributed context sources. The discoverer service works by allowing context sources to register with attributes describing the context information they provide. The context-aware system can then query the discoverer service to get a handle to the context source. This service is useful when trying to locate context sources. A workflow system should thus also include this kind of service.

Context sources should offer both context source polling and subscription services. The polling approach gives the ability to retrieve context information when it is needed. Applications are, however, not notified of changes to the context information. Publish/subscribe services do not necessarily provide context information when needed, since listeners only receive events when they occur. We consider it necessary to allow subscribers to specify conditions for when to receive contextual events. Workflow clients running on limited mobile devices may easily be overloaded without such conditions. A workflow system should also be able to specify conditions for which abstraction level these events should be received, to avoid unnecessary processing.

A context situation is usually complex with multiple context sources generating information. The sum of this information forms the overall state of the environment, which may require operations to be performed by the workflow enactment service. This means that we need to specify rules for every operation that can be used in an inference engine to discover whether that operation is necessary. The sentient object abstraction [10] becomes appropriate for such a context-handling component. The input to the component is sensor information. With an inference engine and the workflow enactment service as actuator, it fits the definition perfectly.

5 DISCUSSION

The number of possible process paths increases with more context sources and possible states (values) from every source. E.g. with **n** possible context sources where each may enter **M** different states, the process paths splits based on these context sources would in the worst case be M^n. This is the worst case scenario if no grouping of values or priority of context sources is possible. A way to avoid such scenarios is to try to aggregate context values to reduce the number of permutations using, e.g., sensor fusion. This aggregation can both be conducted in the underlying context framework and in the context inference engine connected to the workflow system.

Inference engines mainly deal with values of either true or false. Fuzzy logic is a super set of normal Boolean logic. It is possible to have partial truths in fuzzy logic. This is useful when dealing with exception conditions for context values where the exact state cannot be ascertained. It would be possible to use fuzzy logic in the inference engines of a context-aware workflow system to handle a wider array of context states without involving an exception handler. This should be investigated further.

Activity coordination between several workflow participants should be possible in the workflow enactment service for both pre-planned and unplanned situated processes. The same is true on the workflow client level for situated activities. A workflow process can contain both cooperating and competing workflow participants. A workflow enactment service should provide some level of process coordination by controlling the migration of activities. The coordination can thus be made implicit when using dynamic process definition from situated processes.

The coordination of situated activities for the workflow client is more complex. Each situated activity should specify its relation to other situated activities. The other workflow clients then become context sources for the workflow client running a situated action. We may be able to control how work progresses by using the state of the other workflow clients as context in the local situated process enactment. However, this aspect of coordination requires further study.

It is possible to use context information in workflow transitions without changing any of the WfMC standards. This can be accomplished by using an external component to gather the context information from the environment on behalf of the workflow system and make this information available by activity parameter passing. Simple responsiveness to context changes can be achieved by using the functionality of the WfMC WAPI [19] and an external component. The dynamic behaviour achieved by this approach is limited to the methods defined in the WAPI.

Based on the discussion of situated planning and situated activities, a context information integration component is considered best integrated within a workflow enactment service, since this component needs to make radical changes to processes as they run and build processes based on current contextual conditions. Such a component can be built around the sentient object programming paradigm [10]. The input to the component is sensor information, it contains an inference engine, and the workflow enactment service works as an actuator. This component would effectively form a new interface to the workflow enactment service as shown in Figure 5. This interface is proposed as an enhancement of the reference model described in Section 2.1.

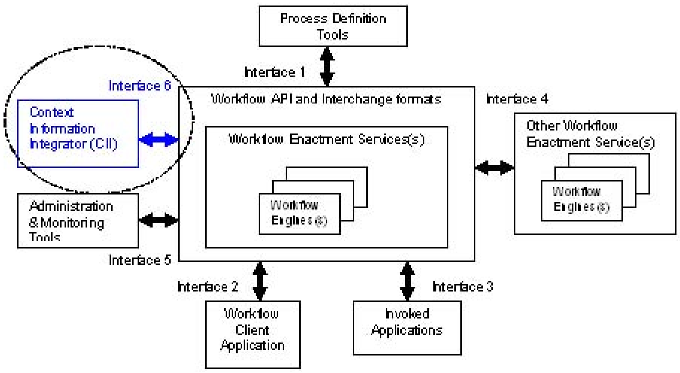

Figure 5. WfMC workflow reference model with new interface

The workflow client needs a local workflow enactment service to satisfy the requirement for situated actions. This means that similarly to the workflow service, the client needs to build situated local processes, based on local contextual conditions. This situated planning should also be based on inference rules.

The handling of context related exception situations is another important aspect of a context-aware workflow system. An exception handler for a context-aware workflow system should be able to handle exceptions generated during process enactment and exceptions based on context. The exception handler can be used to increase the dynamic behaviour of a workflow enactment service with regard to process path selection. The same is true for the local enactment service of the workflow client.

The WfMC defines two types of activities. Activities without calling applications are called manual activities. Automated activities can make use of applications directly. Situated activities fall in between these two definitions. A situated activity is done manually by a human participant, but the way the activity is completed is controlled by automatic means. We will therefore suggest a semi-automated activity, which is defined in between the manual and automated activities. This is illustrated in Figure 6, which is an extended version of the basic terms and relationships defined by WfMC.

Activities consist of work item data and invoked applications. Context sources have a vital role for workflow activities to be performed situated. We have therefore chosen to include context sources as a part of activity instances as illustrated in Figure 6.

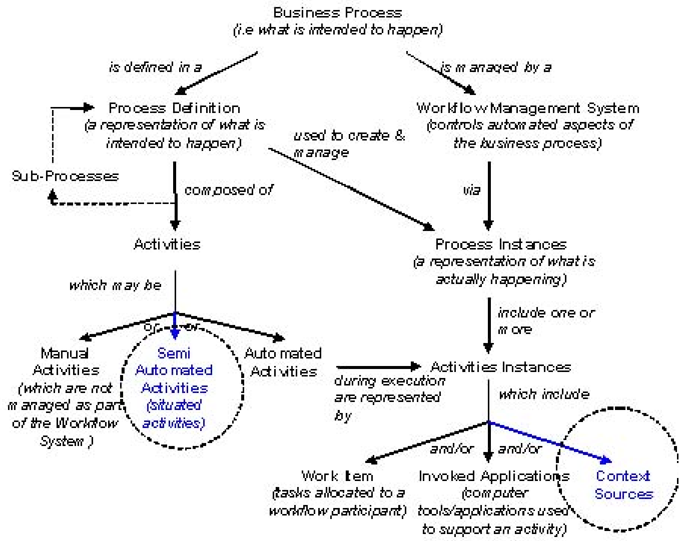

Figure 6. Revised basic terms and relationships

6 RELATED WORK

Some of the existing workflow management tools like Staffware [20], WebSphere MQ Workflow [13], and TeamWare [14] concentrate on project execution and provide little or no support for process modelling and project planning. In particular, a change of a process plan during enactment requires a complete restart of the process in most workflow management tools. However, there are a number of other approaches in the area of process modelling and enactment research, e.g.: *Endeavors* [4] is a support system for dynamic distributed execution of (workflow) processes; *Serendipity* [11] is a process modelling and enactment environment that supports collaborative modelling as well as execution of software processes; and *CAGIS* [22] is a Software Engineering Environment with emphasis on support of cooperative work. In CAGIS, it is possible to have local changes in processes in a decentralised manner.

These approaches all have separate ways to model workflow processes with their respective properties and attributes. They are, however, not targeting context-aware work processes in pervasive computing environments.

In [5], an approach to exception handling of expected exceptions is presented. This approach includes a language for expressing expected exceptions, called Chimera-Exc, and a way of integrating the exception handler with the workflow system, called FORO Active Rule component (FAR). The exception handling mechanism captures exceptions and reacts to them. The exceptions need to access the state of the workflow processes. The state is shared between the workflow enactment engine and the exception handler. Chimera-Exc therefore introduces a schema definition language for managing state information about workflows. The schema is a simple, object-oriented schema, consisting of object classes. Chimera-Exc and the FAR architecture provide flexible mechanisms for handling exceptions arising through workflow enactment. In this approach, it is not necessary to model each exception situation in the workflow process description.

7 CONCLUSION

This paper have presented and discussed requirements and design considerations of a context-aware workflow system to be used to support mobile work in pervasive computing environments. We uses the term *smart* work process to capture adaptive, autonomous work support. The WfMC specifications have been used as a starting point to both to the requirements and the design considerations. Based on the properties of a context-aware workflow system, we propose to extend the WfMC reference model with a separate interface to manage context-awareness. We have further extended the basic workflow terms to also include context information.

We have built some prototypes based on the requirements presented in this paper, and are in progress to try these out on realistic work scenarios. It is, however, still too early to adopt smart work processes in real life. Many research challenges are to be addressed before making context-aware workflow systems ready for practical use.

REFERENCES

[1] Rob Allen, editor. *Workflow: An Introduction*, pages 1–24. WfMC, 2001.

[2] Jakob E. Bardram. Plans as Situated Action: An Activity Theory Approach to Workflow Systems. In Tom Rodden, John Hughes, and Kjeld Schmidt, editors, *5th European Conference on Computer Supported Cooperative Work*, pages 17–32, Lancaster University, UK, 7-11 September 1997. Kluwer Academic Publishers.

[3] Victoria Bellotti and Keith Edwards. Intelligibility and Accountability: Human Considerations in Context-Aware Systems. *Human-Computer Interaction (HCI) Journal. Special Issue: Context-Aware Computing*, 16(2–4):193–212, 2001.

[4] Gregory A. Bolcer and Richard Taylor. Endeavors: A Process System Integration Infrastructure. In *4th International Conference on the Software Process*, pages 76–89. IEEE CS, 1996.

[5] Fabio Casati, Stefano Ceri, Stefano Paraboschi, and Guiseppe Pozzi. Specification and implementation of exceptions in workflow management systems. *ACM Transactions on Database Systems (TODS)*, 24(3):405–451, 1999.

[6] The Workflow Management Coalition. Workflow Management Coalition - Terminology & Glossary. Technical report, The Workflow Management Coalition (WfMC), February 1999. Document Number WFMC-TC-1011.

[7] Anind K. Dey and Gregory D. Abowd. A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications. *Human-Computer Interaction (HCI) Journal. Special Issue: Context-Aware Computing*, 16(2–4):97–166, 2001.

[8] J. Eder and W. Liebhart. The workflow activity model WAMO. In *International Conference on Cooperative Information Systems*, Vienna, Austria, May 1995.

[9] Yrjö Engeström, Reijo Miettinen, and Raija-Leena Punamäki, editors. *Perspectives on Activity Theory*. Learning in doing: Social, Cognitive, and Computational Perspectives. Cambridge University Press, 1999.

[10] Adrian Fitzpatrick, Gregory Biegel, Siobhán Clarke, and Vinny Cahill. Towards a Sentient Object Model. In *Workshop on Engineering Context-Aware Object Oriented Systems and Environments (ECOOSE)*, Seattle, WA, USA, November 2002.

[11] John C. Grundy and John G. Hosking. Serendipity: Integrated Environment Support for Process Modelling, Enactment and Work Coordination. *Automated Software Engineering: Special Issue on Process Technology*, 5(1):27–60, 1998.

[12] D. Hollingsworth. The Workflow Management Coalition – The Workflow Reference Model. Technical report, Workflow Management Coalition, Lighthouse Point, Fla., January 1995. Technical Report WFMC-TC00-1003, version 1.1.

[13] IBM. WebSphere MQ Workflow. http://www-306.ibm.com/software/integration/wmqwf/, 2004.

[14] Teamware Group Oy. Teamware. http://www.teamware.com, 2004.

[15] M. Satyanarayanan. Of Smart Dust and Brilliant Rocks. *IEEE Pervasive Computing*, 2(4):2–4, October–December 2003.

[16] Martin Strohbach, Hans Gellersen, Gerd Kortuem, and Christian Kray. Cooperative Artefacts: Assessing Real World Situations with Embedded Technology. In Nigel Davies, Elizabeth Mynatt, and Itiro Siio, editors, *6th International Conference on Ubiquitous Computing (UbiComp'04)*, pages 250–267, Nottingham, England, September 7-10 2004. Springer-Verlag. LNCS 3205.

[17] Lucy Suchman. *Plans and situated actions. The problem of human-machine communication*. Cambridge University Press, 1987.

[18] Carl-Fredrik Sørensen, Alf Inge Wang, and Reidar Conradi. Support of Smart Work Processes in Context Rich Environments. In John Krogstie, Karlheinz Kautz, and David Allen, editors, *IFIP TC8 Working Conference on Mobile Information Systems (MOBIS'2005)*, pages 15–30, Leeds, UK, December 6-7 2005. Springer, New York, USA. IFIP 191/2005.

[19] The Workflow Management Coalition. Workflow Process Definition Interface – XML Process Definition Language. Technical Report WFMC-TC-1025, The Workflow Management Coalition (WfMC), October 25th 2002. Document Number WFMC-TC-1025.

[20] Tibco. Staffware - Business Process Management. http://www.tibco.com/software/process_management/default.jsp, 2004.

[21] L. S. Vygotskij. *Mind and Society*. Harvard University Press, Cambrigde, MA, USA, 1978.

[22] Alf Inge Wang. *Using a Mobile, Agent-based Environment to support Cooperative Software Processes*. PhD thesis, Norwegian University of Science and Technology, Department of Computer and Information Science, NTNU, Trondheim, Norway, February 5th 2001.

[23] Mark Weiser. The Computer for the 21st Century. *IEEE Pervasive Computing*, 1(1):18–25, January–March 2002. Reprinted from Scientific American, 1991.