2013 IEEE International Conference on Business Informatics

Real-Time Collaboration and Experience Reuse For Cloud-Based Workflow Management Systems

Sebastian Görg*, Ralph Bergmann* and Sarah Gessinger*
*Department of Business Information Systems II
University of Trier, 54286 Trier, Germany
Email: see http://www.wi2.uni-trier.de

Mirjam Minor†
†Wirtschaftsinformatik
Goethe University, 60325 Frankfurt a.M., Germany
Email: minor@informatik.uni-frankfurt.de

Abstract—**In this paper we explore the concept of a real-time workflow collaboration platform. The work presents how a cloud-based Workflow Management System (WfMS) combines the technologic features which are offered by the cloud computing paradigm with a developed resource model for collaboration and reuse of experiential knowledge in workflows. It is based on a prototypical generic software system for integrated process and knowledge management and addresses the concept for collaborative workflow modeling. The concept for the reuse of workflows combines the ability of the resource model to share workflows with Case-Based Reasoning, a specific field of Artificial Intelligence. In particular, a sample workflow of a process in the financial industry is discussed. By enabling collaborative workflow modeling and by providing expert knowledge to large group of users, we aim at the improvement of the quality of workflows. Consequently, workload can be reduced, thus facilitating the work of all process stakeholders.**

I. Introduction

During the last years the cloud computing paradigm emerged. Cloud computing uses virtualization and modern web technologies to dynamically provide scalable, network-centric, abstracted IT infrastructures, platforms, and applications as on-demand services [1]. It allows a shift from local IT-systems and information to the internet-cloud. In the cloud, service provider can offer web-based applications which replace local distributed and heterogeneous systems. As high network bandwidths become available, the usage of cloud-based applications delivers the same speed and response times as local applications. Compared to traditional IT-infrastructures cloud computing has attractive features [2]: it leads to lower maintenance costs, more resources can be shared, they are easier to scale and more reliable.

A cloud-based Workflow Management System (WfMS) provides a web-based platform for the modeling and execution of workflows. Because of the almost ubiquitous availability of the internet, new chances for workflow modeling emerge, which stronger involve all process stakeholders, right from the beginning of the process design. As a consequence collaborative modeling and reuse of workflows becomes feasible.

This work describes how a cloud-based WfMS combines the technologic features that are offered by the cloud computing paradigm with a developed resource model [3] for collaboration and reuse of workflows. The concept for collaborative workflow modeling supports the integration of experts and a real-time display of structural changes. The concept for the reuse of workflows combines the potential of the resource model to share workflows with a specific field of Artificial Intelligence research: Case-Based Reasoning (CBR).

An example for the benefit of collaborative workflow modeling in cloud-based WfMS is the opportunity in gaining better workflows. Many enterprises especially small and medium-sized enterprises have only a few employees with the skills to design and maintain business processes and related workflows. This is a big issue when enterprises grow and need to shift from pure operational business to a more strategic alignment. Workflow modeling is a tedious task which growths with the size of the enterprise. Workflow modelers must have expert knowledge on the business of the enterprise in order to model the workflow. Usually, the workflow modeler cannot have such complete in-depth knowledge of an expert. Therefore the workflow modeler can benefit from a cloud-based WfMS, which supports collaborative modeling. The workflow modeler can invite an expert by granting access to him/her to the affected workflow. Thereby, the expert can use the workflow modeling user interface with an arbitrary browser without installing additional software. The workflow modeler and the expert can then collaboratively model the workflow.

Another example for the benefit of cloud-based WfMS is the possibility to reuse best-practice workflows. In particular, administrative workflows are very similar throughout different departments, e.g. the deployment of software-updates or escalation workflows. If the knowledge of such internal workflows is easy to obtain and to discover, there is a chance for synergy because the departments are more informed about the internal workflows of other departments and thus could benefit from the shared, collaboratively gained knowledge.

This work is not specific for a particular domain, but can be customized to a certain application domain. The presented resource model was developed based on the experience in different application domains for workflows and their according requirements: office/administrative workflows [4], chip design [5], scientific workflows [6], cooking workflows [7], construction workflows and personal/social workflows [8].

The idea of a cloud-based WfMS is not a new one. Especially research communities are currently supported by various systems which are specialized on scientific workflows. Scientific workflows capture tacit knowledge on laboratory experiments and express it as valuable know-how, which can easily be reused. The WINGS system [9] assists users in the design of scientific workflows by validating semantic constraints during the modeling phase. It supports its community in sharing and reuse of their workflows. The ClowdFlows system [10] is built as a service-oriented architecture that allows integrating third-party services in workflows at runtime. As a social aspect,

978-0-7695-5072-5/13 $26.00 © 2013 IEEE
DOI 10.1109/CBI.2013.63

every registered user may share and reuse workflows of other users by copying a public workflow in her/his private repository. MyExperiment [11] is a social networking environment for sharing research objects. It offers a collection of scientific workflows which consists most notably of Taverna[1] workflows, which are executable in the Taverna workflow workbench. For the community of MyExperiment the workflow execution engine does not matter but the knowledge in the workflows and the used services. For this reasons it offers a market place where users can share, reuse, and repurpose scientific workflows to avoid unnecessary reinvention. However, these systems are customized for a certain community and their needs and do not regard collaborative workflow modeling.

The remainder of this paper is organized as follows. In the next section, the CAKE framework and its integrated cloud-based WfMS is presented, which is the foundation for this paper. Then, the resource model is regarded, which allows a cloud-based WfMS to reuse any workflow specific resource, thus facilitating collaboration [3]. In the next section, it is described how the resource model and the system architecture enable collaborative workflow modeling. After that, the approach of process-oriented case-based reasoning is explained and how it reuses collectively gathered procedural knowledge in order to support all workflow stakeholders, during the modeling of a workflow. Finally, we draw some conclusions and present an outlook to the next steps of this work.

II. The CAKE System

The Collaborative Agile Knowledge Engine (short CAKE)[2] is a prototypical generic software system for integrated process and knowledge management. CAKE integrates selected research results on agile workflows[3], case-based reasoning (see section V), and web technologies into a common platform that can be configured to different application domains and needs. Agile workflow technology [13], [14] means that a workflow can be modeled and changed on demand, even running workflows can be paused and adapted to new requirements.

A. Workflows in CAKE

Workflows in CAKE consist of tasks which are linked by a control flow, and data objects which are linked by a data flow [15]. In general, the control flow consists of a sequence of tasks and routing constructs. There are several routing constructs, and of these the parallel execution of sequences (AND-Split and AND-Join), the exclusive choice of sequences (XOR-Split and XOR-Join) or the multi-choice (OR-Split and OR-Join) are often supported by WfMS. Tasks can be executed by humans or automatically by an invoked application. The data flow consists of interlinked data objects. Data objects can be linked amongst others between tasks, between tasks and routing constructs, or between tasks and sub-workflows. During workflow enactment, tasks can be executed by services (e.g. web services or specific implementations of automated tasks) or certain activities may require human workflow participants (e.g. experts).

In Figure 1, a very simple workflow in Cake Flow Cloud Notation (CFCN) [16] is illustrated. CFCN is a workflow modeling language which is derived from UML activity diagrams. The assumption is, that it will receive more acceptance by the users of the CAKE system. This assumption is based on current Human Computer Interaction (HCI) research where 'enchantment' is an important aspect and beauty is part of enchantment in using UIs [17]. Furthermore many HCI researchers state that the visual appeal influences factors as (perceived) reliability, usability, information quality, trustworthiness and usefulness [18].

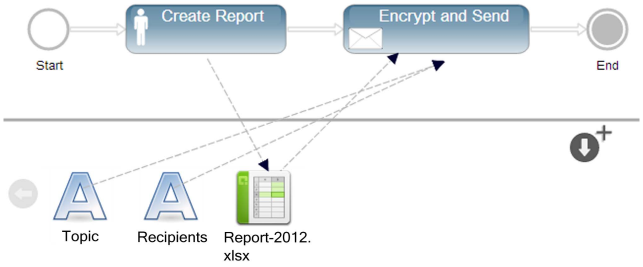

Fig. 1. A simple workflow in the CAKE system.

In this simple example in Figure 1 two subsequent tasks are executed. The first task is delegated to an employee. The employee will create a report and upload it to the workflow. When the data object 'Report-2012.xlsx' becomes available, the execution of the preconfigured automated 'Encrypt and Send' task is triggered. This automated task will send the report to a list of specified recipients.

B. Architecture

In Figure 2, the overall CAKE system architecture is illustrated. It is a cloud platform which provides all services for the modeling and execution of workflows. The client side consists of web-based applications which offer these services to users. The server side consists of a storage layer which handles persistency and access control, an interface layer for the communication with the web applications and two central engines, which are briefly described in the following.

The workflow engine is used for the enactment of workflows. Its internal architecture is strongly tied to the reference architecture of the Workflow Management Coalition (WfMC) [19]. Thus, the workflow engine provides interfaces for modeling and execution of workflows (Agile Workflow Execution), for invoking applications (Service Connector), and an interface for the delegation of tasks (Worklist Manager). All these interfaces are encapsulated in the CAKE server API. On the basis of Figure 1 these interfaces can be explained. The insertion of the two tasks and the data objects uses the interface of the agile workflow execution. In doing so, the workflow engine creates an executable representation of the workflow, which is instantly written in the storage layer with all access information. Because the first task ("create report") is to be executed by a human, the workflow engine prepares all necessary information for the task delegation. This includes the task description as well as all provided data items and data items that shall be produced during the task execution. When the workflow is executed and the task becomes active, these prepared information are passed to the

[1] http://www.taverna.org.uk/

[2] For a detailed description visit http://cake.wi2.uni-trier.de

[3] For more information on agile workflow technology please refer to [12].

worklist manager interface, hence the according user interface can visualize these information to the delegate user. The second task (”Encrypt and Send”) is automatically executed by the service connector interface when the task becomes active during the runtime of the workflow. This interface provides a set of available implementations. An implementation can be generic like the usage of an exposed web service URL as well as a customized one like the communication with REST services. New implementations require a specific XML description which can be understood by the service connector interface. In contrast to web services (which already have a description language: WSDL), customized implementations must describe their required input parameters and their return values.

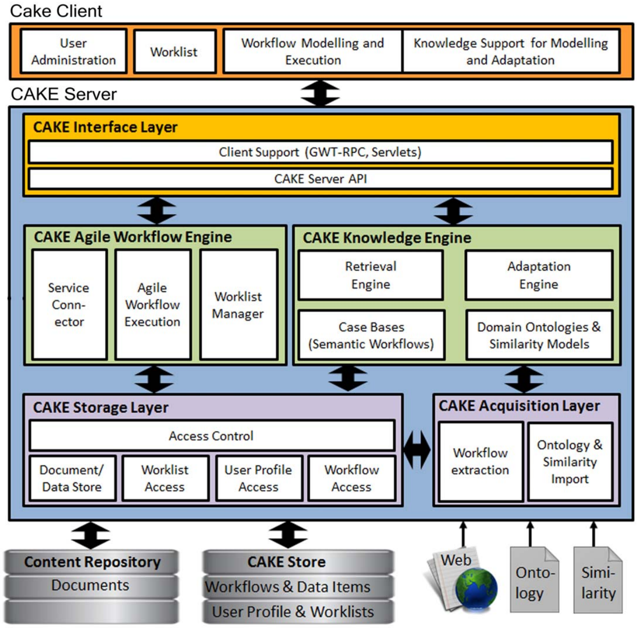

Fig. 2. The CAKE system architecture.

The knowledge engine supports process-oriented case-based reasoning [20], [21], [22], [23] and aims at intelligent methods for reusing experiential knowledge (see section V). It supports similarity-based retrieval of workflows based on semantic descriptions as well as the adaptation of workflows [24]. The aim is to support unexperienced workflow modelers with gathered and structured knowledge in a database. This experiential knowledge can then be discovered and reused in order to build or to improve a workflow under construction. The reuse of procedural knowledge is closely linked to the possibility to share and spread workflows.

The interface layer provides communication means which encapsulate the functions of the two engines and it enables the development of complex web applications such as a workflow modeling and execution environment. It provides a real-time push mechanism, which propagates all changes on workflows to the affected clients. Thus, if several users concurrently model a single workflow, each of them immediately observes the structural changes to the workflow done by all others.

III. The Resource Model

The stage of development of the CAKE system towards a cloud-based platform made it necessary to elaborate an open resource model. By open we mean that every person can register for the system and gets a set of predefined access rights on resources (documents, workflows, etc.). The model enables all registered persons to share their resources and to discover new ones. A detailed description of the resource model would go beyond the scope of this paper and the interested reader is referred to [3]. In the following the foundations for collaborative workflow modeling and reuse are described according to the resource model.

The resource model is illustrated in the UML class diagram of Figure 3. Everything in the CAKE system is considered as a **Resource**. A resource has exactly one owner. **Participants** are resources as well, and hence we need to avoid that an **Object**, e.g. a task or a data object, gets access rights to a participant. The subclasses of participant are **Account** and **Participant Group**, which is a group of an arbitrary set of accounts. Accounts have a unique identifier.

For the retrieval of resources - especially workflows - the enrichment with semantic information is a means to simplify and improve the discovery of resources. The **Semantic Meta Data** can be realized via **Ontologies** which should be specified in a public standard format such as OWL to enable inference mechanisms.

The left four main types of resources are the **Adaptation Case** (see section V-B), **Data**, **Workflow** and **Task**. Due to the nature of workflow management systems, tasks are connected with participants by a specified **Execution**. In the reference model of the WfMC it is the role of the Worklist which controls the **Human Execution** and task delegation. The **Assignment** differentiates between allowed and assigned participants. The modeler may define several allowed participants to execute a task, but only one participant can be assigned in the end. A participant group can be a set of colleagues or experts, who fulfil a predefined role in an organization. Besides the human execution, we also support automated tasks, which can be executed without user interaction. For these tasks, the **Service Execution** is responsible. Automated tasks which use the service execution are custom software components. The automatic sending of an e-mail or the retrieval of data from a database are examples for this.

A concrete workflow consists of a control flow with tasks and data objects linked with tasks. The definitions of prototypes and instances are a generalization of the common workflow-schema/instance separation in WfMS to fit to every concrete workflow resource of the resource model.

- A **Prototype** is a template which cannot be executed. We have prototypes for workflow, task, and data items. We distinguish between **system** prototypes and **derived** prototypes. System prototypes are reusable entities. The usage of a system prototype leads to a derived prototype. A derived prototype is configurable and editable.
- An **Instance** is a copy of a derived prototype. The instantiation creates an executable copy of a derived workflow prototype. Only instances can be executed. Hence, we have instances for workflow, task and data items.

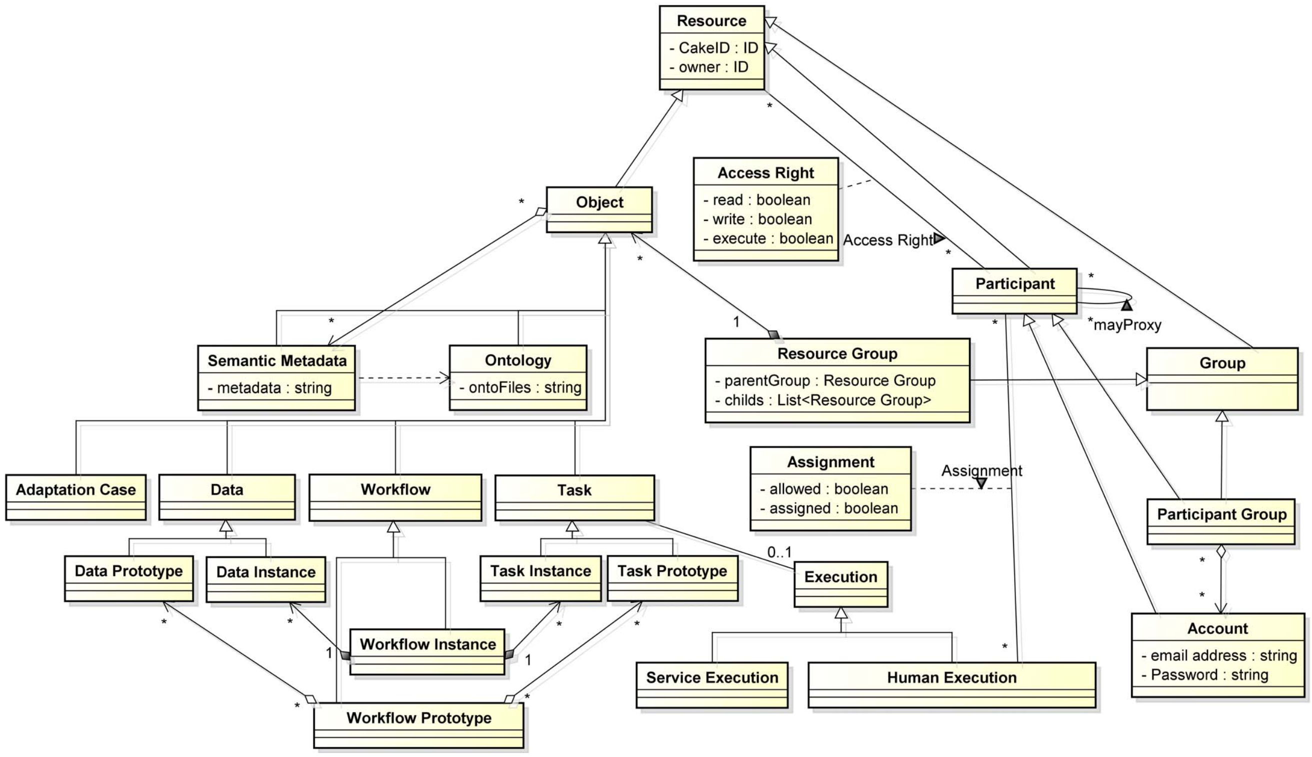

Fig. 3. The resource model.

In the middle part of the diagram the **Resource Group** is shown. There are two reasons why it is different from the participant group. The first reason is the durability of the group members. If a resource group is deleted, all of its members should also vanish, because they are no longer needed. A group of participants needs another behavior. The deletion of a participant group may never entail the deletion of its group members, i.e. the accounts. The second reason is the need to give a hierarchical structure to resources. In contrast to the participant group, the resource group should be able to form tree structures. Objects might be grouped by participants in order to organize and share personal libraries of resources. For instance, a new secretary shall see all administrative workflows for a certain division of a company. Then, the secretary only needs the read right to the folder (represented as resource group) where these workflows are.

IV. Enabling Collaboration

To enable collaborative work was one of the goals during the development of the resource model. This section covers the topics of ownership and access rights as well as their role in the resource model. Then, collaborative workflow modeling in the CAKE system is described and an example is given.

A. Access Rights

Access rights are defined between participants and resources. Thus, a set of people (participant group) or a single person (account) can get access rights to an object resource. This also means that every person can control his/her visibility to the public by restricting read rights about himself to a certain person or a group of persons. Access rights regulate three kinds of access: read, write, and execute. The access to a resource is managed by the access rights, but the access rights can only be assigned by the owner. The resource model only regards positive access rights. It is not possible to explicitly deny a right. Hence, inconsistent access rights due to the membership of participants in different participant groups cannot occur. The access rights that a single account possesses, is a union of all access rights for this account and all participant groups in which the account has a membership. On certain resources, not all access rights are applicable. For instance the execute right is only applicable to workflow instances, but not to individual task or data items.

B. Ownership

Access rights can only be assigned or removed by the resource owner. Ownership can only be taken by a single account. The concept of access rights could also allow to transfer the ownership of a resource to another participant or group of participants, but this will not be supported for the following reasons.
Workflows are always group members of a resource group. The resource group facilitates the organization of resources for the users and thereby gives an outer structure to a set of resources. It can build tree structures, which represent folders, projects, or e.g. shared resources. It must be ensured that an outer resource structure has always a responsible person who controls the access rights. Ownership is passed top-down, so no inconsistent access rights can occur in the outer structure. The same also counts for all resources within the outer structure.

If there is a workflow on which several people collaboratively work, it is not intended that a user can lose the ability to read, write, or execute his/her formerly owned workflow, because someone else has taken the ownership. In such a case, the owner of the outer structure in which the workflow lies would differ from the workflow owner and this would cause an inconsistency. Also non-executable resource objects, such as data and documents, must meet this requirement.

If the ownership of a resource could be passed to a participant group, then it would be possible that an empty group is the owner of an already running workflow instance, which would also lead to the consequence that no one is responsible for the execution of the workflow and its monitoring. Therefore, only accounts may have an ownership.

C. Collaborative Workflow Modeling

After the explanation of ownership and its connection with access rights, collaborative workflow modeling is portrayed in this section. To allow collaborative workflow modeling is not only a matter of giving write access to a workflow for another person, but also a technical one. First some technical issues are stated and then the importance for access control by the resource model.

If a set of users has a write right to a workflow, it can happen that they overwrite their changes without knowing it or that changes become obsolete, because they were made on objects which no longer exist. A simple solution for this problem is an exclusive lock. Every time a user wants to edit a workflow she/he asks the system for an exclusive lock. If the lock is not hold by another user, the system will give him/her exclusive access to the workflow. Although this idea seems very straight-forward, it strongly depends on the willingness of the users to unlock a workflow. It will fail, if the lock-holding user forgets to unlock the workflow and even worse if she/he falls ill or goes on vacation. Another solution for this problem is backed by the CAKE system architecture itself. For collaborative workflow modeling in CAKE an exclusive lock-mechanism is not mandatory, because of two reasons:

- Every operation on a workflow and its elements is persistent. An operation is only performed completely or not. Concurrent operations are synchronized in the agile workflow engine. If an operation is not applicable anymore, because of prior changes on the workflow, it is omitted.
- The workflow engine core only creates push events if an operation on a workflow or its elements has been written persistently. The interface layer of CAKE broadcasts this event to all registered users, who currently regard the concerned workflow in the web-based interface. The user will only see changes in the user interface, if a push event has been received.

The CAKE system currently integrates both methods. Nevertheless, the second one is an optimistic strategy, because theoretical a person could perform a change on a workflow in the time frame between the persistent write of an operation by another user and the receiving of the push event. But for the operation of CAKE as a cloud-based WfMS the second approach seems more advantageous.

The resource model is the other part which facilitates collaborative workflow modeling. Its access control mechanism allows collaboration on different levels. Not only workflows can be edited collaboratively but also all structures which are based on the resource group. So an owner can give write access to a folder (represented by a resource group) to different participants. Then, each of these participants must also get the write right to all workflows which are organized below the concerned structure. If such a write-released resource is modified by inserting a new resource, the access rights must also be assigned to the owner of the parent structure and all participants which have write rights to the parent structure.

Figure 4 shows how access rights are set if a workspace is accessible by two users and a new workflow is created. User A is the workspace owner and so she/he is the owner of all resources of this workspace. In this scenario the workspace owner has granted User B read and write access to the resource ’Folder 1’. Thus, User B has read and write access to all resources within this folder, i.e. in this example ’workflow prototype 1’. User B then decides to create a new workflow (’workflow prototype 2’) in ’Folder 1’ because she/he wants to collaboratively model this workflow with User A. User B may do it, because she/he has write access to ’Folder 1’. However, User B is of course not the owner of the new workflow prototype but she/he obtains the inherited rights of ’Folder 1’ to the new workflow. Then, both users can collaboratively model ’workflow prototype 2’. For collaborative workflow modeling access must be granted to an entire workflow.

Resource	User A Read	User A Write	User B Read	User B Write	Owner
Workspace	X	X			User A
Folder 1	X	X	X	X	User A
WF prototype 1	X	X	X	X	User A
WF prototype 2	X	X	X	X	User A

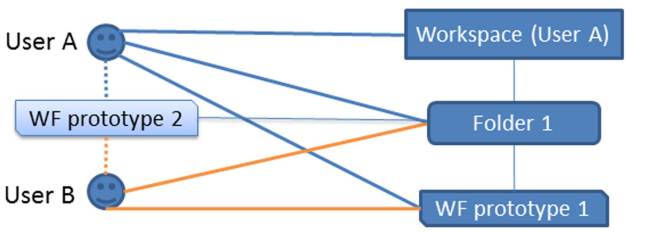

Fig. 4. A collaboration example.

It is not advisable to grant access to single tasks or data objects within a workflow for modeling. An access to fragments of a workflow would pull these elements out of their context and their linkage to the rest of the workflow. For further details on this topic please refer to [3]. Additionally, an operational cloud-based WfMS would need much time to check the access to every resource or even more granular for every attribute of a resource.

Similar to collaborative workflow modeling, the resource model allows the sharing of resources. Sharing of resources is the reuse of resources by other participants. The next section is about the reuse of procedural knowledge, which has been gathered collaboratively in an organization or in a community.

V. Experience Reuse

Workflow modeling usually requires significant skills and experience in the respective domain and in modeling principles in general. Additionally, when performed using some modeling environment, skills in using this environment and knowledge about the available services that can be integrated into a workflow is important. As far as the quality is concerned, a well-designed workflow is hard to produce by a person with no or little experience in workflow modeling. To address this problem, we propose a process-oriented case-based reasoning approach sketched below.

A. Process-Oriented Case-Based Reasoning

Human problem solving in many fields is based on extensive experience. The computer supported reuse of experience requires representing experience in appropriate data structures and Case-Based Reasoning (CBR) provides appropriate means for representing experience [25]. The assumption is that many problems that we face in the present are similar to problems in the past. Therefore, solutions of past problems could help to solve present problems. CBR stores the experience in cases. A case records a problem situation together with the experiences gained during a problem-solving episode, which mostly includes a solution.

In process-oriented CBR the case structure aims at solving problems for workflow modeling and to support workflow modelers to react on environmental changes which influence the runtime of the workflow. Process-oriented CBR systems are able to support the creation and adaptation of workflows [26]. In the next section, the adaptation case representation in the CAKE system is described.

B. Adaptation Cases

A possible representation for cases in process-oriented CBR is the adaptation case [24], which we use in the CAKE system. An adaptation is the representation of a previous adaptation episode of a workflow. It is structured into a problem part and a solution part:

1) The problem part consists of
   a) a semantic description of the change request (what causes the change) and
   b) the original workflow prior to the adaptation.
2) The solution part contains
   a) the adapted workflow and
   b) the description of the adaptation steps (added and deleted workflow elements) that have been executed to transform the original workflow into the adapted workflow.

The change request (1a) is a semantic description of the cause of the request. This part of the problem description makes use of traditional case representation approaches [25], e.g. a structural representation by meta data or a textual representation. Workflows (1b and 2a) are represented as described before in section II-A. The representation of the adaptation steps (2b) deserves some special attention. Similar to STRIPS operators [27], the adaptation steps are described by an add and a delete list. Each list contains a set of chains of workflow elements. A chain encapsulates a maximum set of connected workflow elements with one entry point and one exit point. Further, each chain records a pair of anchors. A pre anchor is the workflow element (in the original workflow) after which workflow elements from the chain have been added or deleted. A post anchor is the workflow element from the original workflow following the last element of the chain. Hence, the pre anchor describes the position after which the adaptation starts and the post anchor describes the first position in the original workflow that is not anymore affected by the adaptation described in the chain. These anchors are further used during the reuse of the workflow adaptation to identify similar points in new workflows at which the proposed adaptation can be applied. Please note that the overall description of the adaptation steps can be automatically computed [6] from the original workflow and the adapted workflow or alternatively it can be captured as part of a modeling tool that is used for manual workflow adaptation.

Figure 5 illustrates a sample for the collection of an adaptation case in a real-time collaboration scenario. The considered workflow in CFCN describes a software update process in the financial industry. A bank (A) needs to update their transaction system. Enterprise (B) is an ICT service provider, who is responsible for the maintenance of the IT-systems of bank (A). Normally, all relevant system components are updated by enterprise (B) and after that enterprise (B) informs bank (A) that the update is complete. Afterwards, bank (A) needs to test the enrolled update for compliance. If the update is compliant, bank (A) informs enterprise (B) that the release was successful. In this scenario an exception occurred during the test of the enrolled update. Bank (A) discovered that the transaction system also transfers data to the German Federal Financial Supervisory Authority (BaFin) and that the BaFin requires certificates for the communication. Therefore, bank (A) pauses the execution of the workflow and gives enterprise (B) write access to the update process. Enterprise (B) models a new task for the generation of a valid certificate. The workflow is now adapted and bank (A) can continue its execution.

This process of exception management, from the start of the exception until the collaborative adaptation of the expert (ICT-enterprise (B)), can be captured in an adaptation case. It contains a textual change request (”We need a certificate in order to transfer data to the Financial Supervisory Authority.”) and the workflow before and after the adaptation. The add and delete lists with the containing chains can be computed automatically [6]. In this case, there is only one chain in the add list, containing the task ”Generate certificate for authentication” and its pre- and post-anchor of the workflow before the adaptation. The next time, an other update process fails because of missing certificates, this adaptation case could be reused in the organization of bank (A), because it contains the knowledge how to create a ’generate certificate’ task and who was responsible the last time to generate the certificate.

This is only a very simple sample of an adaptation case. In [6] adaptation episodes for scientific workflows were investigated. For scientific workflows we found connected chains of add and delete lists which crossed several workflow hierarchies.

The resource model allows participants to access repositories of adaptation cases by giving them read access to the cases. This way, every participant has a dynamic repository which comprehends the gathered knowledge of his/her department or

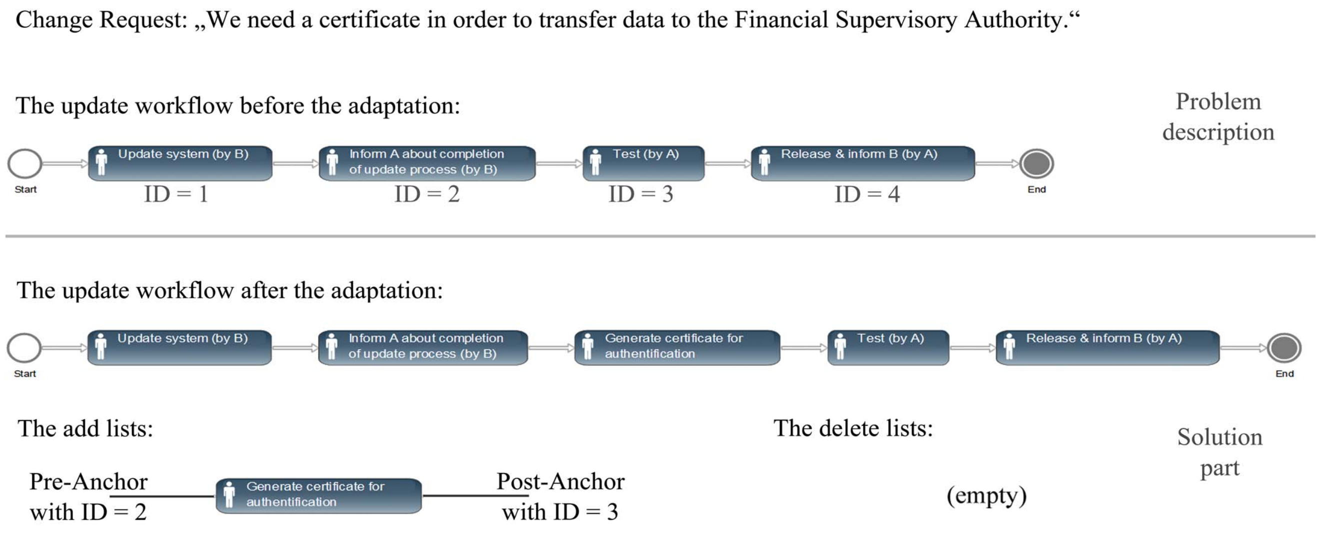
Change Request: „We need a certificate in order to transfer data to the Financial Supervisory Authority.“
The update workflow before the adaptation:
Problem description
Start
Update system (by B)
ID = 1
Inform A about completion of update process (by B)
ID = 2
Test (by A)
ID = 3
Release & inform B (by A)
ID = 4
End
The update workflow after the adaptation:
Start
Update system (by B)
Inform A about completion of update process (by B)
Generate certificate for authentification
Test (by A)
Release & inform B (by A)
End
The add lists:
Pre-Anchor with ID = 2
Generate certificate for authentification
Post-Anchor with ID = 3
The delete lists:
(empty)
Solution part

Fig. 5. A sample workflow adaptation case 'Update software on transaction system.'.

organization.

C. Workflow Modelling and Adaptation

For the support of workflow modeling we differentiate between manual and automatic adaptation support. In the manual scenario users may start describing their problem by creating a brief sketch of the workflow (e.g. a sequence of 2 or 3 tasks) they are interested in, thereby specifying already some structure of the intended solution. The retrieval component will then propose workflows that contain a sub-workflow similar to what the user has specified. This method is similar to the retrieval via workflow execution traces [28] but our retrieval methods focus on the workflow structure. If the user is satisfied with the found workflow, she/he can transfer it to the adaptive workflow management system and prepare it for execution. However, if necessary, the user can also manually adapt the current workflow by editing it via the workflow editor. For this manual adaptation scenario, adaptation cases are not mandatory. It would be sufficient for a person to have a workflow repository. The workflow repository for a person consists of all workflows to which a person has access via the resource model and the similarity-based retrieval of the knowledge engine (see section II-B) searches for similar (sub-)workflows. The success of workflow sharing and providing access via similarity-based retrieval, relies on the willingness of the users to collaboratively create repositories of workflows, which continuously evolve. However, similar ideas have already proven successful for scientific workflows (cf. myExperiment [29], Wings [30]), i.e. addressing a community of professional researchers.

As a step ahead of such a manual adaptation, we also envision an automatic adaptation of workflows supported by workflow adaptation cases. When it should become necessary in the course of workflow execution, the user can even modify an already running workflow. The agile CAKE workflow engine therefore provides a breakpoint mechanism [24]. It allows suspending certain areas of a workflow from execution to enable their modification without loss of consistency. This feature is important if a person, who is responsible for the execution of a workflow needs to adapt the workflow to environmental changes. In such a scenario the concerned person can articulate the environmental change or the problem as a change request. The structure of the current workflow and the change request build a query, which can be sent to the CAKE system. If an appropriate case, with a high similarity to the query, exist, the system can automatically suspend relevant areas of the workflow by the breakpoint mechanism. Finally, the according add and delete lists of the found adaptation case can be applied on the current workflow.

VI. Conclusion

In the course of this paper, we have described the cloud-based WfMS CAKE and its collaborative workflow modeling approach as well as the modeling support by experience reuse. These two features could be a means to continuously increase the quality of the process design in an organization. On the one hand, experts in a department can support process design, because they can easily participate in the modeling process. On the other hand, trainees or employees, who are new to workflow modeling, can ask the system to deliver recommendations in form of workflow templates or adaptation steps, based on a concrete change request.

Due to the extent of this paper, some issues of the resource model and the access control mechanism cannot be discussed: The modeling and access of hierarchical workflows and the reuse of workflow instances requires a version management for the produced data during the runtime. Also, the approach of case-based reasoning could only be sketched without the details, e.g. how anchor positions of the past are mapped to a new problem/workflow. Of course, the reuse of experiential knowledge is directly depended on the amount of gathered knowledge and the willingness of its users to spread their knowledge. The quality of these automatic adaptations has to be assessed in an evaluation. So far, we proved the feasibility of the concept for the application of adaptation cases in the office domain [24].

First evaluations of workflow reuse are planned. Similarity measures for workflows have already been developed and assessed [31] and methods to identify matching positions during the application of add and delete lists already exist [32], [24]. Besides that, the user interfaces for the integration of the resource model are under development.

Acknowledgment

This work is part of the WEDA project (Web-based design, reuse, documentation and execution of agile Workflows). WEDA is funded by Stiftung Rheinland-Pfalz für Innovation, grant no. 974.

References

[1] C. Baun, *Cloud computing: Web-based dynamic IT services*. Springerverlag Berlin Heidelberg, 2011.

[2] X. Liu, D. Yuan, G. Zhang, W. Li, D. Cao, Q. He, J. Chen, and Y. Yang, *The Design of Cloud Workflow Systems*, ser. SpringerBriefs in Computer Science. Springer, 2011.

[3] S. Görg, R. Bergmann, S. Gessinger, and M. Minor, "A resource model for cloud-based workflow management systems enabling access control, collaboration and reuse," in *Proceedings of the 3rd International Conference on Cloud Computing and Services Science*, 2013, p. to be printed.

[4] M. Minor, D. Schmalen, S. Kempin, and S. Kempin, "Demonstration of the agile workflow management system cake ii based on long-term office workflows." in *BPM (Demos)*, 2009.

[5] M. Minor, D. Schmalen, and A. Koldehoff, "Fallstudie zum einsatz agiler, prozessorientierter methoden in der chipindustrie," in *Business Services: Konzepte, Technologien, Anwendungen, 9. Internationale Tagung Wirtschaftsinformatik, 25. - 27. Februar 2009, Wien*, H. R. Hansen, D. Karagiannis, and H.-G. Fill, Eds., vol. 1. Oesterreichische Computer Gesellschaft, 2009, pp. 193 – 201.

[6] M. Minor and S. Görg, "Acquiring adaptation cases for scientific workflows," in *Case-Based Reasoning. Research and Development, 19th International Conference on Case-Based Reasoning, ICCBR 2011*, ser. LNCS, vol. 6880. Springer, 2011, pp. 166–180.

[7] K. Walter, M. Minor, and R. Bergmann, "Workflow extraction from cooking recipes," in *Proceedings of the ICCBR 2011 Workshops*, B. Diaz-Agudo and A. Cordier, Eds., 2011, pp. 207–216.

[8] S. Görg, R. Bergmann, M. Minor, S. Gessinger, and S. Islam, "Collecting, reusing and executing private workflows on social network platforms," in *WWW'12 Workshop Proceedings*, 2012.

[9] Y. Gil, V. Ratnakar, J. Kim, P. A. Gonzlez-Calero, P. T. Groth, J. Moody, and E. Deelman, "Wings: Intelligent workflow-based design of computational experiments." 2011, pp. 62–72.

[10] J. Kranjc, V. Podpecan, and N. Lavrac, "Clowdflows: A cloud based scientific workflow platform." in *ECML/PKDD (2)*, 2012, pp. 816–819.

[11] C. A. Goble, J. Bhagat, S. Aleksejevs, D. Cruickshank, D. T. Michaelides, D. R. Newman, M. Borkum, S. Bechhofer, M. Roos, P. Li, and D. D. Roure, "myexperiment: a repository and social network for the sharing of bioinformatics workflows." 2010, pp. 677–682.

[12] M. Minor, A. Tartakovski, D. Schmalen, and R. Bergmann, "Agile workflow technology and case-based change reuse for long-term processes," *International Journal of Intelligent Information Technologies*, vol. 4, no. 1, pp. 80–98, 2008.

[13] B. Weber and W. Wild, "An agile approach to workflow management," *Proceedings of Modellierung 2004*, pp. 187–201, 2004.

[14] M. Reichert and P. Dadam, "Adept flex supporting dynamic changes of workflows without losing control," *Journal of Intelligent Information Systems*, vol. 10, no. 2, pp. 93–129, 1998.

[15] W. van Der Aalst, A. Ter Hofstede, B. Kiepuszewski, and A. Barros, "Workflow patterns," *Distributed and parallel databases*, vol. 14, no. 1, pp. 5–51, 2003.

[16] M. Minor, R. Bergmann, and S. Görg, "Adaptive workflow management in the cloud - towards a novel platform as a service," in *Proceedings of the ICCBR 2011 Workshops*, 2011, pp. 131—138.

[17] P. Wright, J. Wallace, and J. McCarthy, "Aesthetics and experience-centered design," *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 15, no. 4, p. 18, 2008.

[18] G. Lindgaard, C. Dudek, D. Sen, L. Sumegi, and P. Noonan, "An exploration of relations between visual appeal, trustworthiness and perceived usability of homepages," *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 18, no. 1, p. 1, 2011.

[19] W. M. Coalition, "Terminology and glossary," http://www.wfmc.org/standards/docs/TC-1011_term_glossary_v3.pdf, 1999, [Online; accessed 14-Nov-2012].

[20] R. Bergmann, A. Freßmann, K. Maximini, R. Maximini, and T. Sauer, "Case-based support for collaborative business," in *Advances in Case-Based Reasoning, 8th European Conference, ECCBR 2006, Fethiye, Turkey, September 4-7, 2006, Proceedings*, ser. LNCS, T. Roth-Berghofer, M. H. Göker, and H. A. Güvenir, Eds., vol. 4106. Springer, 2006, pp. 519–533.

[21] R. Bergmann and Y. Gil, "Retrieval of semantic workfows with knowledge intensive similarity measures," in *Case-Based Reasoning. Research and Development, 19th International Conference on Case-Based Reasoning, ICCBR 2011*, ser. LNCS, vol. 6880. Springer, 2011, pp. 17—31.

[22] T. Madhusudan, J. Zhao, and B. Marshall, "A case-based reasoning framework for workflow model management," *Data and Knowledge Engineering*, vol. 50, no. 1, pp. 87 – 115, 2004, advances in business process management.

[23] E. Chinthaka, J. Ekanayake, D. Leake, and B. Plale, "Cbr based workflow composition assistant," in *Services - I, 2009 World Conference on*, july 2009, pp. 352 –355.

[24] M. Minor, R. Bergmann, S. Görg, and K. Walter, "Towards case-based adaptation of workflows," in *Case-Based Reasoning. Research and Development, 18th International Conference on Case-Based Reasoning, ICCBR 2010, Alessandria, Italy, July 19-22, 2010. Proceedings*, ser. LNAI 6176, S. Montani and I. Bichindaritz, Eds. Springer, 2010, pp. 421–435.

[25] R. Bergmann, *Experience Management: Foundations, Development Methodology, and Internet-Based Applications*, ser. Lecture Notes in Computer Science. Springer, 2002, vol. 2432.

[26] M. Minor, S. Montani, and J. A. Recio-Garca, "Process-oriented case-based reasoning." in *Proceedings of the ICCBR 2011 Workshops*, 2011, pp. 75–138.

[27] R. Fikes and N. J. Nilsson, "Strips: A new approach to the application of theorem proving to problem solving," *Artif. Intell.*, vol. 2, no. 3/4, pp. 189–208, 1971.

[28] D. B. Leake, J. Kendall-Morwick, and J. Kendall-Morwick, "Towards case-based support for e-science workflow generation by mining provenance." in *ECCBR*, 2008, pp. 269–283.

[29] D. De Roure, C. Goble, and R. Stevens, "The design and realisation of the myexperiment virtual research environment for social sharing of workflows," *Future Generation Computer Systems*, vol. 25, no. 5, pp. 561–567, 2009.

[30] Y. Gil, V. Ratnakar, J. Kim, J. Moody, E. Deelman, P. A. González-Calero, and P. Groth, "Wings: Intelligent workflow-based design of computational experiments," *Intelligent Systems, IEEE*, vol. 26, no. 1, pp. 62–72, 2011.

[31] R. Bergmann and Y. Gil, "Similarity assessment and efficient retrieval of semantic workfows," *Information Systems, Special Issue on Processoriented Case-Based Reasoning*, 2012.

[32] M. Minor, S. Islam, and P. Schumacher, "Confidence in workflow adaptation," in *ICCBR*, ser. Lecture Notes in Computer Science, B. Díaz-Agudo and I. Watson, Eds., vol. 7466. Springer, 2012, pp. 255–268.