A Document-Centric Approach to Open Collaboration Processes

Nelly Schuster, Christian Zirpins, and Stefan Tai

Karlsruhe Institute of Technology (KIT),
76131 Karlsruhe, Germany
{firstname.lastname}@kit.edu

Abstract. Individuals collaborate with each other in groups in order to solve complex tasks or to jointly develop new solutions. The Web offers them a huge amount of data, information and services to be used as input to their collaboration. In our research we examine such creative and emergent open collaboration processes and their relationship to documents as a communication and collaboration vehicle. Our goal is to better support coordination and integration in such collaborations. From a technological perspective, we aim to support open collaborations by putting a service-oriented document model and middleware in the center of a collaboration. As a first step, we provide a concrete design of a document collaboration system representing documents as rule-based mashups of RESTful services provided by humans, enterprise systems or the Web. As a second step, we aim to analyze communication of resources in open collaboration processes. We strive to use this analysis data in order to a) provide end-users of our collaboration system with a visual method in order to help them understanding the evolution of their document collaboration ecosystem and processes and to b) allow detecting potential inconsistencies in rule-based process definitions which can be caused by the open ecosystem of resources. In this paper we give a short overview of the current state of our research and outline future research directions.

Keywords: document mashups, open collaboration processes, document engineering, rule-based mashup.

1 Research Context

Creative collaborative design of software systems or research activities in teams are examples of so called emergent knowledge processes [1]. These processes denote the engineering of documents involving different individuals, activities and resources, which might be changed in an ad-hoc way. These open collaboration processes require lightweight coordination support. While structured or semi-structured processes can be modeled in advance, open collaboration processes emerge during runtime often in unforeseen ways. For instance, at the beginning of such a collaborative process it is not clear, which resource is affected or needed at which time and who participates when in the collaboration. Thus, methods and tools for open collaborations need to be able to adapt to changing participants and resources. In addition, there need to be mechanisms to capture and specify dependencies between activities.

Besides coordination support, open collaboration processes require content consolidation support. The Web offers a steadily growing amount of distributed resources providing content or functionality, e.g. linked open data or translation services. These content or functionality services can be used, reused and exchanged in open collaborations. In order to allow easy integration of these sources into open collaboration, new kinds of tools are required which are able to cope with this evolving ecosystem of autonomous and heterogeneous Web resources.

Various approaches exist to support flexible design and execution of collaborative processes [2, 3]. However, they do not focus on processes targeted at collaboratively engineering documents. On the other hand, Web-based collaboration technologies like online collaborative writing tools have improved the effectiveness as regards information consolidation and communication, but they do not explicitly drive the coordination of collaborations and information integration.

New kinds of software and architectures can be built from compositions of Web resources [4]. For example, mashups of services allow end-users to build composite applications for situational purposes [5]. We claim that this new kind of end-user-driven compositions based on service-oriented computing technology can be leveraged to support open collaboration processes.

In the following Section 2, we describe the general goals and planned contributions of this research. In Section 3, we outline the state of our current approach for supporting open document collaborations and describe two ideas for future work. Related work for our approach and the planned future work is described in Section 4. A short conclusion is given in Section 5.

2 Goals and Contributions

As a first goal, we aim to integrate the mashup approach with document models in order to support open collaboration processes from a technological perspective. We represent documents as *document mashups* of human- or software-provided content delivery, transformation or publication services. Coordination of activities is supported by user-defined rules which are part of the mashup. This should enable end-users to coordinate their document collaborations and to more easily integrate Web resources as contributions to their collaborations. As a concrete result, we provide a model of document mashups, and the design and implementation of a middleware and collaboration system.

Second, we aim to analyze communication of services in open collaboration processes. The emerging nature of the processes and the ecosystem of autonomous services make it hard for coordinators of a collaboration to keep track of the collaboration and to detect and resolve potential inconsistencies. Thus, we strive to collect data exposed by services (e.g. changes of underlying resources), mashups (e.g. service compositions) and rules (e.g. execution state) and analyze and prepare this data in order to a) provide collaboration coordinators with a visual method helping them to understand the evolution of their collaborations and used services and to b) allow detecting potential inconsistencies in the specified rules which can be caused by the autonomous nature of the services. The idea is to analyze the data in real time to be able to support the coordinator with the information needed in a certain situation. We intend to integrate these features into our document mashup tool.

3 Approach

In the following sections, we describe our document mashup approach. In addition, we describe our ideas for supporting end-user-engineering of our document mashups based on analyzing data of open collaborations.

3.1 Model and Tool for Open Collaboration Process Support

We developed a document mashup model that defines documents as artifact-centric collaboration processes [6].

Fig. 1 shows the logical collaboration structure of a document mashup supporting the authoring of a scientific publication. The example mashup includes five structural nodes, e.g. for the abstract and the main part of the publication. Nodes are coordinated by human coordinators who can select activities which should be performed on the node or its subtree.

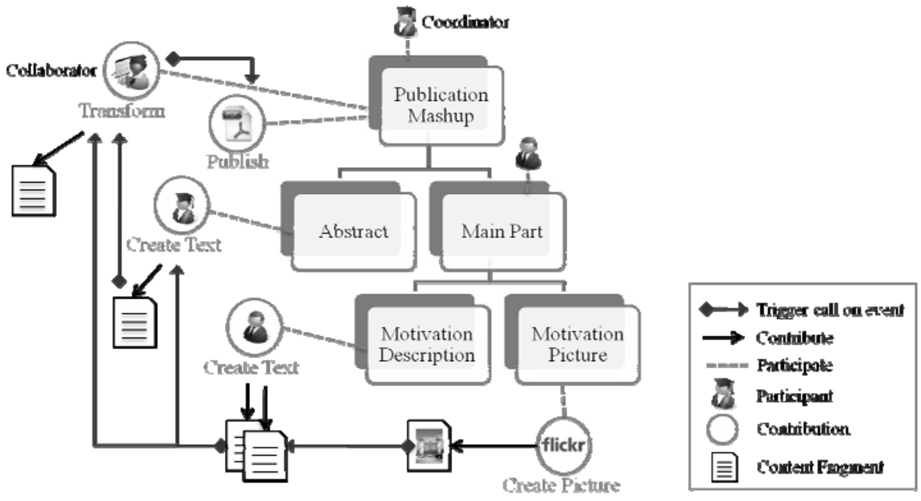

Fig. 1. Document mashup example: scientific publication

Activities of a collaboration are mapped to content creation, transformation and publication services. Examples are writing the "abstract" of a paper, proofreading or publishing the mashup to a Weblog. We assume that all services follow a uniform create-retrieve-update-delete interface. Services can be provided by humans, Web services or enterprise systems. Results of the activities are integrated into the graph structure and represent contributions to the collaboration.

Each service might expose events indicating creation, updates or deletion of its underlying content fragments. On the one hand, the document mashup is updated accordingly. On the other hand, the events are consumed and reacted upon by interaction rules which are part of the mashup and regulate and control activities. For instance in Fig. 1, the update of a picture provided by flickr triggers an update request to a human who writes a text about the picture.

During collaboration the mashup graph structure as well as included services and interaction rules can be adapted by the coordinators as required.

Based on the document mashup model, we have designed and implemented a prototype infrastructure for event-based interaction between document mashups and services which follow the REST architectural style. Furthermore, we have implemented a Web-based collaboration environment for end-users leveraging the infrastructure which supports collaborative mashup authoring and service provisioning use cases.

3.2 Monitoring and Analyzing Open Collaborations

In open collaboration processes, involved or available providers, resources and activities are changeable and heterogeneous in their nature. In addition, the collaboration itself evolves over time. The development of next-level collaboration systems thus requires making such collaborations and the evolution of the resources understandable. We aim to monitor and prepare data exposed by services as well as mashups in order to be able to analyze their behavior.

Precisely, we want to monitor creation, update and deletion events emitted by services when underlying resources change. Furthermore, we response times of certain services can be measured. We aim to use this information for the analysis of service lifecycles. When observing events of services of a specific mashup, the collaboration flow could be reconstructed. It might also be possible to detect implicit dependencies, e.g. if the update of a service always is followed by the update of another service although no rule is specified. Furthermore, influences of a service could be analyzed, e.g. if changes to it have a high impact on the whole mashup it might be a central part of the mashup.

Observing the rule execution in a document mashup could be used to find blocking rules, e.g. rules which wait for a second event and thus block the execution of other services. Furthermore, potential inconsistencies could be analyzed, for instance, if a service call was triggered by a rule but not yet answered.

When observing these behaviors for a series of mashups, frequent compositions of services, service types or providers or typical collaboration flows could be detected.

We believe that this knowledge about the evolution of services and mashups can be used to enhance collaboration platforms in various ways, from end-user support to system architecture adaption. Following, we present two ideas how the data analysis can be leveraged for better end-user support.

3.3 Data Visualization

Representing information in a visual form often is used to make complex information better understandable. Thus, we propose to develop visualizations of selected service and mashup data for end-users in order to support them in their open collaborations. This might be advantageous in various use cases.

As mashup coordinators need to select services to add to their mashup, they might be interested in the lifecycle or evolution of a service in order to make the decision about including the service. Thus, a visualization of the evolution of a service might accompany the textual or semantic description. For instance, dots on a timeline indicating updates of the underlying resource could show how the dynamicity of a service developed over time.

In an advanced scenario, a list of services could be shown to the coordinator which seemed to be useful in similar situation or collaboration context, e.g. they were frequently used with a service which the coordinator just added to the mashup.

In order to support traceability and understandability of the collaboration, browseable visualizations could show the flow of the collaboration. A playback mechanism could be applied to show which contributions were made to the mashup at which point in time. In order to show the evolution of a mashup without presenting the contents the history flow visualization [7] could be adopted. This visualization shows how parts of a document are developed by various authors over time. This or other kinds of visualizations could also help in detecting blocking or boosting activities or showing influences of a certain event on the collaboration.

3.4 Rule Recommendations

Dependencies between services in document mashups are specified by the mashup coordinator through event-condition-action (ECA) rules. For instance, the rule engine retrieves an update event from a picture service and triggers as action a request to a text service. Input to such rules are events representing service requests and responses, resource updates, deletions and creations. Rules can be specified based on existing resources. As resources might change over time or might disappear, rules need to be adapted to this changing environment.

Large and long running mashups potentially produce large rule bases. These rule-bases need to be kept up to date and consistent by the mashup coordinator. Thus, we aim to use the service and mashup data in order to find potential inconsistencies, deadlocks or useful or needless rules. Based on the findings the system could make recommendations or indications to the mashup coordinator. In order to support rule recommendations, we first need to determine critical events and the possible inconsistencies they could cause.

An example for an inconsistent rule is, if a rule is waiting for an update event of a deleted service. This might also cause a deadlock. Contrariwise, if the action of a rule calls a deleted service, it is inconsistent. If the system detects such rules, it either removes them automatically or it notifies the mashup coordinator who then can change or delete the rules.

Rules could be recommended to the mashup coordinator if implicit dependencies between services are detected or certain rules are very common in similar mashups.

3.5 Planned Evaluation

We intend to evaluate our document mashup approach through end-user experiments in different domains, namely collaborative authoring of student project reports and project proposals. The overall goals are to evaluate the applicability of the general document mashup approach in these scenarios as well as the usability of our collaboration tool. The experiments will also be used to monitor service and mashup data for our intended extensions to the approach. In a second end-user case study experiment, we will evaluate the usefulness of these extensions.

4 Related Work

Our work adopts concepts of different fields of research like service mashups, open document collaboration, as well as software evolution analysis and visualization. We partially use approaches from these fields as entry points or additions to our work.

Mashup technologies aim at enabling end-users to compose situational applications from Web-based content and services. Several mashup research tools and products exist which offer a graphical user interface. These tools range from very specific tools which are restricted in the operations and formats they support to general purpose mashup tools. For instance, Yahoo! Pipes (http://pipes.yahoo.com) is a powerful tool to compose and filter data feeds. Another example is the IBM Mashup Center (http://www-01.ibm.com/software/info/mashup-center/) that lets users compose widgets referencing various types of services and data. Widgets can be composed both presentation-wise and through simple rules which route data between them. However, we did not find a mashup approach facilitating (document) collaboration of humans while allowing the integration of contents and services from the Web.

As regards *open document collaboration*, various research prototypes and products exist. Several Web 2.0 applications like TypeWith.me (http://typewith.me/) or Google Docs (http://docs.google.com/) allow collaborative authoring of rich text documents. However, these tools do not support coordination mechanisms or the integration of content or services from the Web. Another related technology is Google Wave (http://wave.google.com/). A wave is a collaboration of participants based on XML documents consisting of wavelets. This is similar to the composition of document services in a mashup. Waves might include automated robots that are comparable to our services which are not provided by humans. However, Google Wave focuses more on communication than on collaborative evolution of a document. Also, there is no way to define interaction rules based on events and to re-use wavelets in other waves.

Related work to our planned extension to visualize service and mashup data is described [8], where the evolution of data on programmableWeb.com is analyzed and visualized. The programmableWeb.com API gives access to a large database of Web API and mashup descriptions. However, this work focuses on the general evolution of the mashup ecosystem, but not on the evolution of specific mashups or services.

Furthermore, related work can be found in the field of *analysis and visualization* of software or software evolution. There exists work based on mining of different kinds of repositories, like bug databases or versioning systems [9]. There is also work on the evolution characteristics of Web services [10]. However, we did not find any approach which has collaboration processes in mind when analyzing software evolution.

With respect to analyzing processes, related work exists in the field of structured or semi-structured processes as well as the mining of processes for patterns [11], even for ad-hoc processes [12]. However, none of these approaches focus on documents as communication and collaboration vehicle.

5 Conclusion

Our work provides a unique and novel approach to support open document-centric collaboration processes. In particular, it firstly provides a model, infrastructure and collaboration tool for document mashups to support teams in collaboratively authoring documents like scientific publications, EU-proposals or software documentation. The solution allows coordination of collaborations through representing collaboration activities as autonomous services which can be interconnected through rules reacting on events of these services. Furthermore, content and functionality services from various sources, i.e. humans, software systems or the Web, can be integrated into the evolving document.

Secondly, in future work we aim to capture and analyze data of such collaborations. This potentially allows visualizing service lifecycles or collaborations in order to help end-users understanding the evolution of services and collaborations and thus helping them to choose the services, steps or compositions they need at hand. Furthermore, coordination rules in a mashup could be checked for inconsistencies. Based on detected inconsistencies, the system could recommend rule base adaptions to the mashup coordinator. Using real time data emitted by services and mashups could improve the user experience of collaboration systems and enable new application areas involving more challenging collaboration scenarios or less experienced users.

References

1. Markus, M., Majchrzak, A., Gasser, L.: A Design Theory for Systems that Support Emergent Knowledge Processes. MIS Quarterly 26(3), 179–212 (2002)
2. Ceri, S., Daniel, F., Matera, M., Raffio, A.: Providing Flexible Process Support to Project-Centered Learning. IEEE Trans. on Knowl. and Data Eng. 21(6), 894–909 (2009)
3. Dustdar, S.: Caramba – A Process-Aware Collaboration System Supporting Ad hoc and Collaborative Processes in Virtual Teams. Distrib. Parallel Databases 15(1), 45–66 (2004)
4. Shaw, M.: Architectural requirements for computing with coalitions of resources. In: Proceedings of the First Working International Federation for Information Processing Conference on Software Architecture (1999)
5. Yu, J., Benatallah, B., Casati, F., Daniel, F.: Understanding Mashup Development. IEEE Internet Computing 12(5), 44–52 (2008)
6. Schuster, N., Zirpins, C., Tai, S., Battle, S., Heuer, N.: A Service-Oriented Approach to Document-Centric Situational Collaboration Processes. In: Reddy, S. (ed.) WETICE, pp. 221–226. IEEE Computer Society, Los Alamitos (2009)
7. Viégas, F.B., Wattenberg, M., Dave, K.: Studying cooperation and conflict between authors with history flow visualizations. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI 2004, pp. 575–582. ACM, New York (2004)
8. Yu, S., Woodard, C.: Innovation in the Programmable Web: Characterizing the Mashup Ecosystem. In: Service-Oriented Computing – ICSOC 2008 Workshops, pp. 136–147 (2009)
9. Voinea, L., Telea, A.: Visual data mining and analysis of software repositories. Computers & Graphics 31(3), 410–428 (2007)
10. Treiber, M., Truong, H.-L., Dustdar, S.: On Analyzing Evolutionary Changes of Web Services. In: Service-Oriented Computing – ICSOC 2008 Workshops, pp. 284–297 (2009)
11. van der Aalst, W.M.P., van Dongen, B.F., Herbst, J., Maruster, L., Schimm, G., Weijters, A.J.M.M.: Workflow mining: A survey of issues and approaches. Data & Knowledge Engineering 47(2), 237–267 (2003)
12. Truong, H.L., Dustdar, S.: Online Interaction Analysis Framework for Ad-Hoc Collaborative Processes in SOA-Based Environments. Transactions on Petri Nets and Other Models of Concurrency 2, 260–277 (2009)