A Service-Oriented Approach to Document-Centric Situational Collaboration Processes

Nelly Schuster\textsuperscript{1}, Christian Zirpins\textsuperscript{1}, Stefan Tai\textsuperscript{1}, Steve Battle\textsuperscript{2}, Nils Heuer\textsuperscript{2}
\textsuperscript{1}FZI Forschungszentrum Informatik, Karlsruhe,\textsuperscript{2}Hewlett Packard
\texttt{nschust|czirpins|tai@fzi.de, battle|nils.heuer@hp.com}

Abstract

Agile networked and service-centric organizations increasingly rely on organizational processes of an open collaborative nature that are not well supported by standard information technology. Enterprise documents and service computing are orthogonal technologies that provide promising features to amend this situation, as regards human interaction, as well as situational control structures, but their paradigms cannot easily be integrated. In this paper we illustrate open document-driven collaborations and discuss their requirements. Furthermore we introduce a service-oriented approach to integrating enterprise documents within SOA landscapes leveraging service mashups for ad hoc situational document collaboration.

1. Introduction

In the context of globalized markets, service-dominated economies and networked enterprises, organizations are increasingly required to be responsive to planned changes and unplanned incidents, to innovate in the face of these situations and in collaborations with clients and partners. To meet these requirements, organizations need to look beyond optimization of recurring business processes and focus on open collaboration processes that are situational, weakly structured and highly interactive human-centered collaborations within or across organizational boundaries. For example, the IT infrastructure Library (ITIL) \cite{8} prescribes a number of key processes for IT service providers like incident and problem management that require situational ad hoc collaborations between employees of the providers, their clients and possibly external experts. Because such open collaborative processes are competitive, e.g. with respect to the quality of IT services, it is a key objective to foster and support them by organizational as well as technical means.

From a practical perspective, organizational collaboration is closely related to enterprise documents. Generally, documents provide a means of communicating information in a purpose-optimized (structured, annotated, graphically appealing, legally binding) form of representation in order to share and clarify individual points-of-view with targeted recipients. In the enterprise, documents are often used as an organizational instrument. Sophisticated enterprise documents encapsulate business rules and processes and provide the means for their regulation and enforcement in a highly efficient manner e.g. utilizing customer relationship, and workflow management, systems. Thus, it is a natural step to leverage document models and technologies to support situational document collaboration.

In terms of technical support for situational document collaboration, various technologies enable organizational processes and human collaboration. Technologies like BPM have significantly improved the efficiency of organizations as regards optimizing and automating highly structured and recurring core business processes \cite{15} but largely, they fail to provide effective support for open collaborative processes of an ad hoc and fluid nature. To foster more general forms of collaboration, CSCW technologies have improved the effectiveness of organizations as regards information sharing and communication \cite{11} but they do not explicitly drive open collaborative processes in an efficient manner. Orthogonal, service-oriented computing technology provides an integrative platform and architectural framework for composing more flexible process-driven business information systems \cite{6} and for mashing up situational end-user applications \cite{16}.

Service computing provides a promising basis to support open collaborative processes, but common service-oriented architecture (SOA) \cite{6} misses an explicit document abstraction at the business level. Because service computing leverages XML to represent data in the context of service invocations, it is principally possible to map existing XML-based enterprise document models onto SOAP RPC parameters or REST style HTTP messages. However, these documents are not present as an explicit abstraction at the
architectural level and thus cannot be easily integrated with business logic or instrumented for organizational purposes. This is a severe problem, because it results in the disconnection of two major paradigms of ICT-enabled organizational instruments and obstructs the utilization of service computing technology for situational document collaboration. Accordingly, we aim to devise service-oriented document environments that allow for coordinated collaborative creation and evolution of composite enterprise documents within the surrounding SOA landscape.

More specifically, we are developing a service-oriented approach for composite enterprise documents supporting ad hoc collaboration and weakly structured innovative (situational) processes within and across enterprises. As a technical foundation, we propose to integrate electronic documents with SOA software services and processes. We are developing a lightweight document service bus that represents document fragments as stateless software services and introduce a simple composition model to coordinate document content, editing and publishing services. Our document service model provides the basis of an intuitive mashup language for document-centered situational collaboration. Collaborative mashups regulate and enforce flexible and dynamic interaction patterns for document communication supporting organizational teams.

In this paper, we intend to contribute to the practical understanding and conceptual portfolio of organizational support for collaborative documents and document service mashups. Following, we will describe the context and concepts of situational document collaboration from a practical and illustrative perspective. In section two we describe application domains for situational document collaboration and discuss their requirements. Section three gives an overview of our approach to service-oriented documents and collaborative mashups. This is further illustrated in section four by a concrete use case from IT service management. Finally, we discuss related work in section five and conclude with a summary in section six.

2. Domain analysis for situational document collaboration

Situational collaborative processes involving document creation and evolution can be found in various parts of an organization. We identified several domains and analyzed their characteristics with respect to typical documents, processes and types of collaboration. We are especially interested to see how documents evolve during human collaboration and whether processes are explicitly modeled beforehand or are reactions to certain situations or events. Furthermore, we tried to partition documents in order to analyze the reusability of the parts in other documents and examined which data sources were used, e.g., the Web, databases or humans. The ability to automatically generate or update document parts and automate certain parts of the collaborative process was also an important aspect.

We outline the results of three application domains: IT infrastructure; software documentation; and scientific documents. These were chosen because they are relevant in various organizations and promise a mixture of structured and dynamic document processes, involving collaborative activities on semi-structured reusable documents.

IT infrastructure and service management processes should be kept flexible and highly-responsive to ad hoc requests, in order that business processes can adapt faster to requirements and customer demands. In this domain we looked at documents and processes in the IT Infrastructure Library, ITIL [8]. ITIL provides guidelines for typical enterprise documents, e.g., problem reports, configuration specifications, as well as best practices for various types of IT management processes where participants with different roles use, create or evolve documents collaboratively. These processes are not always clear-cut, but must be flexible and reactive. Processes react to incidents, e.g., a customer problem, and thus require ad hoc refinements in documents, for instance, the creation of a problem report that is further refined when a problem solution was found. In addition to human beings, who contribute to a document, data comes from different sources, e.g. from databases, the Web, or system measurements. ITIL provides a holistic view on relevant IT service management (ITSM) processes with individual perspectives that depend on each other and thus need to be integrated to leverage their full potential.

Similar to ITSM, software is an essential driver of business. In case of failure it is important to immediately gather relevant information, e.g. from software documentation, in order to rapidly react to the problem. Therefore, our second domain is software documentation, describing different perspectives of a software product created and evolved during different phases of the software lifecycle; for instance, design, usage or problems during operation. This domain is characterized by a diversity of best practices, for instance formal methods, architectural styles, or model-based approaches, and by a high reusability of knowledge, e.g., in the form of templates or design patterns, which can be found in books, the Web or similar project documentation. Processes in this domain vary from structured, e.g., bug fixing, to open and creative, e.g., architectural design, but often involve situational collaboration of participants with certain roles, e.g., architect, developer, user, who are interested in and continuously
evolve different parts of the whole documentation. Parts might be reused in several documents addressing different stakeholders, e.g., the architectural overview can be reused in documentation for architects, customers and developers. Logical links between document parts, e.g., the design should realize all specified requirements, allow for automated checks or code generation. Besides humans, sources for documentation are manifold, e.g., books, discussion boards or sites on the Web, other project documentation, or test results.

Research departments drive innovation in organizations. Not only product innovation but also scientific contributions like publications or project proposals are important assets in this domain. Scientific documents often follow a certain structure and have typical parts like an abstract or a list of citations. Project proposals, for instance, are large documents where responsibilities for different parts are distributed over several participants employed at different organizations. Usually, there are no modeled workflows: During the evolution of such documents, discussions, upcoming ideas and gathered information are integrated in an ad hoc manner into the document. Documents evolve dynamically until the deadline of a call. A coordinator has to consolidate all parts of the document, ensuring their logical flow. In this domain, often the document itself is the reason for collaboration processes.

To summarize, documents in the application domains are partially structured and often composed of reusable parts. Templates and best practices help in the initial creation and guidance through documentation processes. Documents and their contents dynamically and incrementally evolve over time through collaborative activities that include ad hoc and unexpected changes, e.g., as new information come in.

In order to support situational document collaboration, authoring environments need to be flexible, usable and transparent. Extended and extensible editing capabilities are required, which expose a graphical representation of document parts, support different types of data sources and formats, and allow for automation, e.g., through automatic update or generation of document parts. In addition, an underlying document model should be extensible and generic with respect to document semantics and layout, and it should support reusable and composable building blocks. In order to support agile processes, documents or document parts should be able to participate in processes and be capable of exposing and reacting to events. Collaboration features, e.g., discussion boards, and a role system support the exchange and maturing of knowledge. Furthermore, a system supporting situational collaboration should be interoperable with respect to integration of external content into documents as well as preparation of documents for use in external IT systems.

3. A service-oriented approach to document collaboration

In order to support the discussed domains and requirements for situational document collaboration, we are developing an approach that integrates enterprise documents within SOA and leverages resulting synergies for the case of open collaborative processes.

Fundamentally, we propose a document service abstraction in order to promote enterprise documents as first class citizens of SOA. The idea is to represent enterprise documents as compositions of stateless software services. The representation of a document part can be the result of the invocation of a document service function. These functions might have very different natures in the sense that they either provide document content, editing capabilities or publication means. With respect to the SOA paradigm, document services might be described, published, discovered, selected and accessed, which leads to tight integration within an existing SOA landscape. Beyond that, document services can be composed with respect to complex document content, as well as document lifecycle processes. In particular, composite document services may incorporate various types of value added services to facilitate document features like real-time updates, collaborative editing or coordinated routing.

Building on service-oriented documents, we propose to leverage software service mashups to realize a class of composite document service for open collaborative processes. We argue that unlike common service orchestration technologies, mashups provide the simplicity and ergonomics to govern document interaction in the course of open collaborative processes. The idea is to refine generic Web-based content mashups as a way to aggregate document services and add the means to express fundamental interaction patterns for coordination purposes. We expect members of organizational teams to use interaction patterns and document services as building blocks of mashups that can be altered and evolved in the course of collaboration. We believe that document service mashups provide an appropriate instrument to realize situational document collaboration.

In our ongoing work, we continue to explore the conceptual foundations of a holistic document service model as well as the practical implementation of a service-oriented document collaboration platform, both of which are outlined in the following sections.

3.1. Document Service Model

Our document service model describes the foundational concepts of document service components and their composition within document service mashups.
In terms of modeling document service components, a fundamental aspect is the reflection of existing enterprise document models. Here, we have considered a number of mostly XML-based models and specific formats ranging from XHTML to DDF [9]. As regards criteria like document structure, reuse, generality, maturity and simplicity we finally decided to use the DITA model [13] for our initial experiments. Another facet is the software service model, where we focus on an “as-simple-as-possible” enabling technology for a generic class of software services. For our initial experiments, we are using RESTful Web Services [12] based on WSDL and HTTP. On top, we add specific components and relationships for composite enterprise document services. Here we define a variety of document service types for provision, transformation and publication of document parts building on human-interaction, automated mechanisms, and external Web sources. Furthermore, we specify an asynchronous, event-based communication model to communicate between document services.

In terms of document service mashups, we focus on modeling document-driven collaboration on the one hand and definition of a respective mashup language on the other. We are approaching document-driven collaboration by informally specifying a basic catalog of document interaction patterns for common control structures in situational document collaboration. For example, the two-man rule (or four-eye principle) is a control principle that leads to a high level of security in critical situations because all transactions or decisions need approval from two authorized people. A number of such patterns exist in various interdisciplinary fields like business administration, security or politics. In our model, document interaction patterns are not only a means of understanding the control mechanisms of document collaboration but are also intended as reusable design patterns for ad hoc development of document service mashups. On a more concrete level, we are translating the control structures of document collaboration into elements of a domain specific document services mashup language. At the current stage, we have identified the main principles of this language with respect to the general dimensions of mashup approaches [16]. In particular, we are combining principles of layout-based and event-based composition in order to combine document characteristics with collaboration control. This allows for declarative specification of partial control to represent the weakly structured nature of open collaboration processes. Furthermore, we do not distinguish between design time schemas and runtime instances. Document service mashups represent unique collaborations that are interactively created and evolved in parallel to their semi-automatic control.

3.2. Document collaboration platform

Corresponding to the main parts of the document service model, we are implementing document service components and document service mashups by means of a document service bus and mashup infrastructure, both of which contribute to an integrated platform for situational document collaboration. Figure 1 sketches the technical architecture from 30.000 feet.

![Figure 1. Document collaboration architecture](image)

An essential part of this architecture is a service-oriented bus infrastructure that enables document service interaction and composition. To implement the bus, we are developing a lightweight software framework for document services that provides a RESTful API for document message queuing on top of JMS (similar to RestMS for AMQP [5]). We are also developing a set of fundamental Web services to provide bus functionalities like registration of document services and providers, and configuration of message routing.

The document service mashup infrastructure is designed as a value added document service on top of the document service bus. We plan to reuse the mashup engine based on the refinement of an existing implementation, whereby we are currently comparing open source offerings against our own system Swashup [10]. While we don’t need all capabilities of common engines to aggregate the full diversity of Web content, we do need to add control structures to represent and enforce document interaction patterns. The idea is to implement interaction patterns by means of simple value
added document publication services that can be plugged into document service mashups and control communication of components on the bus. Collaborative mashups then consist of simple scripts that define compositions of document content, editing and publishing services provided by different members of an organizational team. To improve end user experience, we are building a Web-based front-end for the mashup document service. A promising direction, here, is to integrate mashup services within Wiki tools – e.g. as an extension to Semantic MediaWiki [7]. The following section describes a scenario in the ITIL domain in which our approach could be applied.

4. Usage Scenario for the service-oriented document collaboration approach

Pity the poor IT staff who work at the service desk; the front-line of IT fire-fighting, otherwise known as Incident Management. They are charged with understanding and recording an 'incident' as the user sees it, and then figuring out the root cause buried deep within the IT system. In ITIL, the steps involved in incident management include incident recording & classification, investigation & analysis, resolution & closure, as well as a host of secondary activities like tracking and monitoring. The IT staff needs to create an incident record – a process-aware and semi-structured document – detailing the lifecycle and ultimate resolution of the incident. Like an online form it must capture mandatory attributes, detailing, for example, customer information. On the other hand, the investigation & analysis phase is far more ad hoc, requiring true problem solving and intelligence. These sections of the document are far less prescriptive and more open-ended. The document itself has a clear lifecycle, and the kinds of editing that are possible at each stage of its lifecycle will differ. Once the incident is resolved the document will be locked-down; perhaps leaving only residual powers to insert addenda. There is no hidden workflow state lurking in a back-end database or the BPM system – the state of the process is equivalent to the state of the document.

Consider this document as a mashup of different kinds of content, combining intelligence from different systems in order to characterize the problem, its root cause, possible workarounds, and ultimate resolution. The ad hoc structure of the document evinces a logical argument that connects its different parts. The internal structure of document mashups is explicit and enabled by a set of services that can be assembled in endlessly innovative ways.

Existing Configuration Management Systems (CMS), including the HP Universal CMDB [4], are good at storing system configuration data, but the general UI, search functionality, and usability of these products are not tailored towards casual use. While ‘hard’ data about Configuration Items (CIs) is formally managed in a Change Process or updated through dynamic discovery, the conventional CMS is not really appropriate for managing ‘soft’ or anecdotal information about CIs. The first line of services we utilize is the ability to place the traditional CMS in a Web 2.0 wrapper. One option is HP’s experimental CMDWiki that provides an alternative UI to ITIL configuration data. This is achieved by integrating the HP UCMDB with Semantic MediaWiki [7] which allows relationships between CIs to be exposed as typed metadata. With the configuration data now in a browsable and document-centric format, we can begin to explore how to include this content within a document mashup.

The service desk worker must deep dive into the IT configuration to identify the root cause of the problem. Each configuration item has its own page on which we can see its type; its properties and relationships with other CIs. To elevate this ordinary browsing experience to an authoring process, we need to build a summary of relevant information; something akin to a site-map, but one that captures the logical ad hoc structure of the problem under investigation. Extensions to Semantic MediaWiki support query interfaces that make it ideal for inclusion in value-added services including the kind of site-summary described above.

Our worker may also draw on additional services that will help them to explain the cause of the problem. The language of the business user is managed in a separate wiki (built on IkeWiki [14]) that supports the collaborative creation and management of a machine processable ITIL ontology.

5. Related work

The Liquid Publications community researches alternative ways for the creation, dissemination, evaluation and maintenance of scientific publications by leveraging methods from agile software development and collaborative evolution of knowledge in Web 2.0 [2]. Similar to our approach, liquid publications are evolutionary, collaborative, reusable knowledge objects that can be composed. However, there is not yet a notion of a system, which could support scientific publications. Also, we support best practice situational collaboration but do not change the notion of a document. Thus, liquid publications are complementary to our research.

Google docs [3] is an online real-time document management tool with which users can create and edit text documents, spreadsheets and presentations collaboratively with office-like functionalities like format-
ting or spell checking. In spreadsheets users can add little visualization applications called gadgets that work with data in the cells of the spreadsheet. Users can register for notification of changes in a document. However, dynamic changes to a part with dependencies on events in other parts of the document are not possible. Furthermore, processes, whether situational or workflow-like, are not supported.

With the IBM Mashup Center [1] users are enabled to create graphical situational mashups through widgets that represent data from different sources, e.g., Web feeds or enterprise data. Widgets can be wired together by specifying which parameter of one widget should be input to another widget. For instance, if a user selects a customer in a customer list widget, the address of the selected customer is shown in a maps widget. IBM Mashup Center provides an extensible catalog of widgets, however, there is no notion of user-provided content. Its focus is on creating situational applications and information representations but not on collaborative documentation processes.

CuteFlow (www.cuteflow.org) is a document circulation and workflow system. Based on routing lists documents aggregated from input fields are e-mailed to recipients who can edit the input fields in their e-mail client. The system manages these circulations until the document is completed. CuteFlow, does not support collaboration on the same part of the document.

6. Summary and outlook

In this paper, we have introduced a novel technique to support open, situational collaboration within and across organizations.

In order to foster understanding and extend the scientific debate, we have introduced and analyzed a variety of application domains and discussed a number of concrete situations of situational document collaboration in the practical context of IT service management that originate from our genuine experience.

Additionally, we have outlined our approach to support open collaborative processes by means of document service mashups and sketched our ongoing activities to materialize it. While we have successfully realized first experimental structures, we are now in the process of elaborating our document service model and extending our document collaboration platform towards a stable prototype that we can evaluate in close collaboration with our industry partners.

7. Acknowledgements

We would like to thank Hewlett Packard for supporting our work in the Innovation Research Program.

8. References