inContext: On Coupling and Sharing Context for Collaborative Teams

Hong-Linh Truong[1], Christoph Dorn[1], Giovanni Casella[2], Axel Polleres[3], Stephan Reiff-Marganiec[4], Schahram Dustdar[1]

[1]*Vienna University of Technology, Austria, {truong,dorn,dustdar}@infosys.tuwien.ac.at*

[2]*Softeco Sismat SpA, Italy, giovanni.casella @softeco.it*

[3]*DERI, National University of Ireland, Galway, Ireland, axel.polleres@deri.org*

[4]*University of Leicester, UK, srm13@le.ac.uk*

Abstract

Present team members have difficulties in keeping the relations between their various, concurrent activities due to the lack of suitable tools supporting context coupling and sharing. Furthermore, collaboration services are hardly aware of related context of team members and their activities. Such awareness is required to adapt to the dynamics of collaborative teams. In this paper, we discuss the context coupling techniques provided by the inContext project. Utilizing the concept of activity-based context and Web services techniques, we can couple individual and team contexts at runtime, thus improving the context-awareness and adaptation of collaboration services such as email, shared calendars, instant messaging and document management.

Keywords

Context, context coupling, interaction, teamwork, collaborative working environment

1 Introduction

In today's collaborative working environments (CWEs), team members utilize various tools and services, such as document repositories, shared workspaces, calendars, email, video conferencing platforms, and instant messaging, to perform their work. As team members normally work on multiple tasks in different projects, managing complex relations between artefacts, assignments, and resources in an efficient way is a challenge. Utilizing context associated with such relations can substantially improve the awareness of collaboration services involved in teamwork. However, in the absence of a framework that supports a full cycle of managing structure of context and correlating and managing context from the design time to the runtime, dependencies and implicit links among team members, activities and collaboration services remain concealed, thus hampering team and process awareness. To solve the above-mentioned problem, contexts associated with people's activities should be coupled at both design time and runtime. Context coupling techniques help to simplify collaboration by making the dependencies and links between humans and services explicit and taking them both into account in the overall context model. They connect people, services, and activities across time and organization boundaries and enable reuse of context information across these dimensions, thus bringing significant enhancements in awareness of collaborators and teams.

The inContext project[1] introduces novel interaction and context-based techniques for collaborative teams, with the focus on emerging team forms such as nomadic, mobile and virtual teams. The inContext system [Truong et al., 2008] comprises of many different types

[1] http://www.in-context.eu

of collaboration services loosely coupled through the Internet using a service-oriented architecture (SOA). To deal with the dynamics of collaborative teams, inContext introduces diverse types of contexts and manages them. To increase team and process awareness, inContext considers context coupling techniques to be of paramount importance. Though, many systems support context awareness for teams, coupling contexts associated with people and their activities in SOA environments has not yet been well addressed. Since current CWEs heavily rely on Web services technologies, the context coupling techniques that inContext presents are targeted to SOA-based CWEs.

This paper discusses the inContext solution to context coupling. We present motivating scenarios in which context coupling could improve the collaborative work in Section 2. We discuss techniques for sharing and coupling context that connect services, human, and their activities by utilizing the concept of activity-centric collaboration in Section 3. Examples illustrating the benefit of context coupling techniques are given in Section 4. We outline the related in Section 5 and Section 6 concludes the paper.

2 Motivation

Team members are always engaged in joint activities. Therefore, any activity, even assigned to an individual, involves other members. In our supporting environments, people working on joint activities belong to different organizations, locate in different places and have different working times. They can move during their work and provide the input/result from different locations. Furthermore, teams may be established in an ad-hoc manner and a team member usually participates in different activities at the same time. Without coupling context, collaboration services are unaware of the connection between people and activities, thus they are unable to adapt to teamwork's needs.

Consider the following scenarios as motivating examples taken from needs of industrial partners in the inContext project. Mr. John, a busy e-worker involved in many projects, spends his days coordinating teams, assigning tasks, collaborating and communicating with other team members, preparing events and writing documents. His address book includes many contacts and he uses several collaboration services to complete his tasks.

In the first scenario, Mr. John must organize a training course to introduce a new technology in his company. The course includes ten lessons, for each lesson a particular topic is covered and two or more teachers are required. Mr. John must repeat the following steps to organize each lesson:

1. *Look at the calendars of the required teachers to find a suitable date for the lesson*
2. *Discuss with the teachers to fix the program of the lesson in detail*
3. *Update the program of the course*
4. *Inform students of the new lesson*

To complete his task Mr. John can use several services: a calendar service, a video-conferencing tool, an e-learning supporting service and an email service. Anyway, even with these services the task of Mr. John's is not easy. For each lesson, Mr. John must query the calendar service to retrieve the agendas of the required teachers and check them to find a suitable date for the lesson. Next he must find the teacher's addresses in his address book and arrange a video-conference with them. When the lesson details have been defined Mr. John must update the program of the course and the calendars of the teachers. Finally he must invoke the email service to create a mail to inform students about the new lesson. The problem is that Mr. John needs several services to complete his task but each service is not aware of Mr. John context. *Context coupling mechanisms will enable Mr. John to use services in a more useful, efficient and engaging way.*

In the second scenario, Mr. John is involved in the "Grenoble city library" project and he must find some suitable locations where to put advertisements for the new library. He can use a map service, an instant messaging tool to interact with Mr. Brown (an employee living in Grenoble) and a shared folder where he can collect all the documents regarding the project. Mr. John has all the services that he needs to complete his task but he spends a lot of time doing unnecessary things. Before using each service, he must provide a set of information to initialise it for his task. In particular, to use the map tool he must centre the map on the Grenoble city. Before to communicate with Mr. Brown he must find his address and initialise the instant messaging tool to interact with him. Finally before to use the shared folder service he must create a suitable folder tree to contain all the "Grenoble city library" project documents and he must find all the email addresses of the project members to send them an email with the shared folder address. Again, the problem is that Mr. John needs several services to complete his task but each service is not aware of Mr. John context (current activity, involved partners, etc.). *Here context coupling mechanisms will enable the services to utilize Mr. John context to eliminate unnecessary things.*

3 Coupling and Sharing Context in inContext

To address context coupling we identify four core capabilities depicted in Figure 1**.** First, we need to describe and manage the *structure of activities* that is so far only available in the user's mind. These activities serve as underlying information for *the creation of correlations* between activities (and thus associated context) and services. On the service side we need to perform *the extraction of these correlations* to retrieve the corresponding *context*. The first capability is realized at design-time context coupling whereas the last three capabilities are achieved through runtime context coupling.

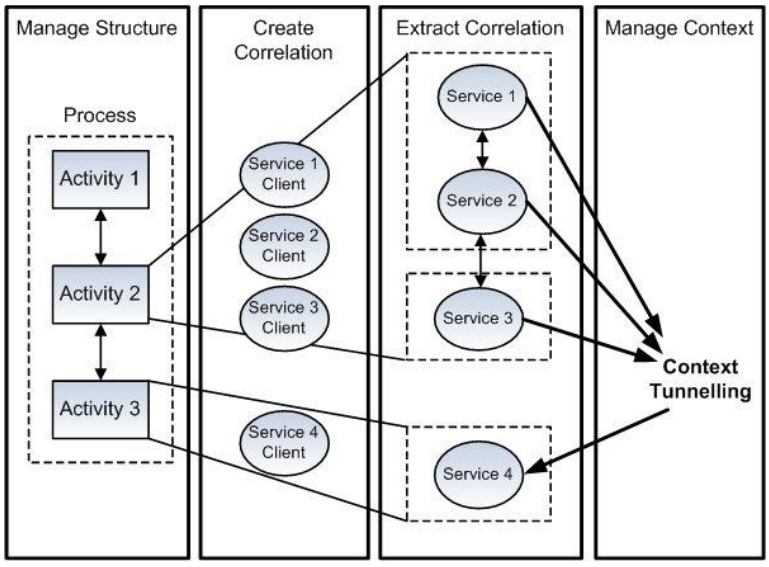

Figure 1: Required core capabilities

3.1 Design-time Context Coupling

To support the "*manage structure*" capability, we rely on the context models for individual (team member), team and activities. Based on the models and using inContext tools, the user can create his/her activities and specify relations among various entities involved in the activities.

Figure 2 shows the association between three types of context abstraction. **Individual context** includes most of the traditional context types such as the current location, available devices, communication channels and channel status, but also more CWE related information such as membership in various teams, current activities (within the different teams), available resources, skills, assignment of tasks, or co-located members (from different teams). **Team context** includes information about interactions, projects, organizations, and locations that are associated with members of a team. Modelling team context is a challenge, especially due to the emergence of new team forms requiring different modelling and reasoning to capture the dynamicity of today's teams. In inContext, team context is structured according to the Team Characteristics Meta Model (TCMM) [Dorn et al. 2007, inContext D1.1]. To couple individual context and team context, we define the **activity context**. Individual context alone does not allow a complete picture. In contrast, team context is too coarse grained and high-level while lacking the awareness of users participating in multiple teams simultaneously. Activity context is most suitable to cover and describe the work settings at various levels of detail: up from the project structure down to the user's current working directory on a device and their use of specific services.

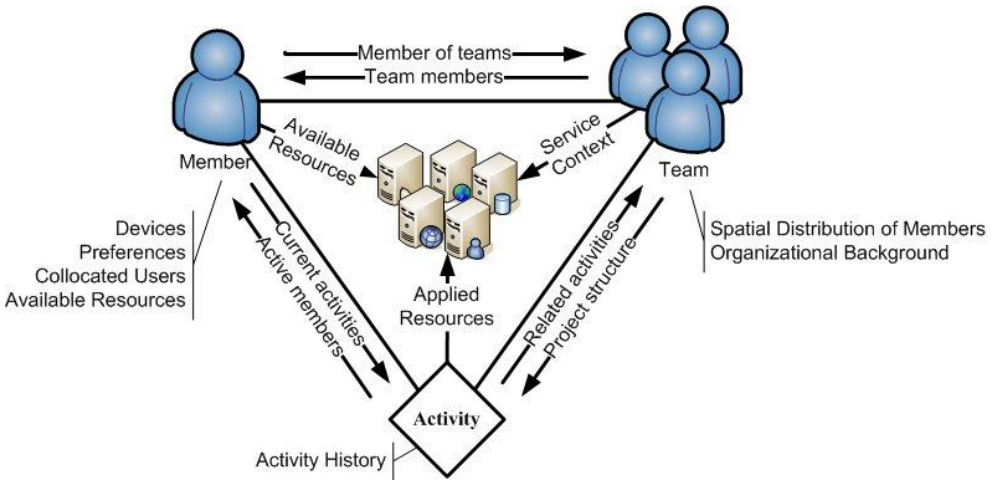

Figure 2: Association between Individual (Member), Team and Activity context

All information relevant to the individual, team and activity context is stored in inContext's context model. Our context model is implemented using the Resource Description Framweork (RDF)[2], RDF Schema[3] and OWL[4]. The ontology-based approach allows for flexibility and extensibility of our current context model, for instance by inclusion of domain-specific data or reuse of Web data already available in common RDF formats.

3.2 Runtime Context Coupling

The three capabilities, namely *"Create Correlation"*, *"Extract Correlation"*, and *"Manage Context"*, are achieved through runtime context coupling. During runtime, context is used in two ways: first of all to identify the best collaboration service for the user's current activity [Reiff-Marganiec et al. 2007] and secondly to provide the used services with the right context. The latter uses runtime context coupling techniques and is the focus of this section. Services require context information as input to perform their respective task and traditionally all data required is passed to the service. Our runtime context coupling techniques do not require the transfer of context information; only the context correlation

[2] http://www.w3.org/TR/REC-rdf-syntax/

[3] http://www.w3.org/TR/rdf-schema/

[4] http://www.w3.org/TR/owl-guide/

information, such as identifiers for the current involved user and activity. This information is used later on to obtain more detailed context information from our framework where and when required.

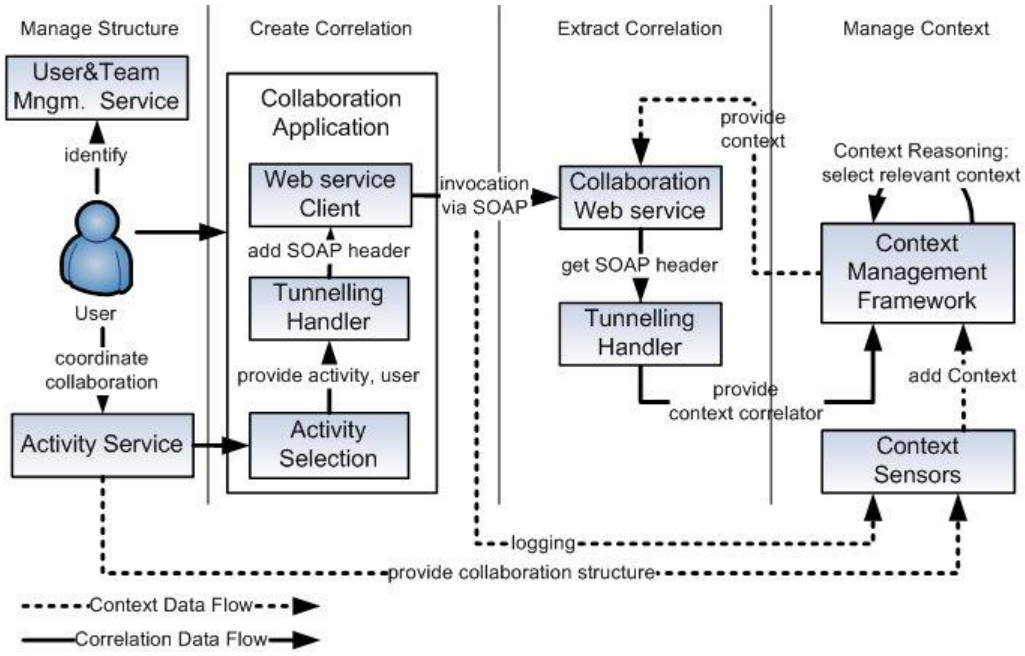

Figure 3: Runtime context coupling

Figure 3 depicts how runtime context coupling is implemented in inContext. The main components of the architecture are:

- **Activity Service:** provides references to users' activities (i.e. ActivityURI).
- **User & Team Management Service**: provides user identifiers (i.e. UserURI) to link invocations to users (thus is part of the correlation creation).
- **Web service Client**: captures the context correlation and passes it to the service by using *Tunnelling Handler*.
- **Collaboration Web service**: receives the correlation by means of the *Tunnelling Handler*.
- **Context Management Framework**: includes the *Context Store* which stores context data and *Context Tunnelling Extension* which enables correlation extraction.
- **Tunnelling Handler**: software component that enables the transfer of correlation information from any *Web service Client* to *Collaboration Web service.*

When a client invokes a collaboration web service, the *Tunnelling Handler* component will send to the web service URIs locating context information. The service then uses the URIs to invoke the *Context Tunnelling Extension* of the *Context Management Framework*. The contextual information (*Context*) is sent back to the service so that the service can utilize the information. The ActivityURI and UserURI [inContext D4.1] are the key information here, as they allow correlating the context of two or more services. For example, consider that we want to pass the context information related to activity **`act1`** and user **`Rossi`**. The ActivityURI and UserURI are **`http://www.in-context.eu/pcsa#act1`** and **`http://www.in-context.eu/pcsa#Rossi.E54`**, respectively, and they will be passed to another service. The context information itself is stored in the *Context Store* of the *Context Management Framework*. To pass this context to another service, the information is added to SOAP calls for any context-aware service by means of a special SOAP header extension. Listing 1 presents a simplified example.

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope
      …
      <soapenv:Header>
            <ns1:ctxtunnelling
      soapenv:actor="http://schemas.xmlsoap.org/soap/actor/next"
            soapenv:mustUnderstand="0" xmlns:ns1="www.in-context.eu">
            <ns1:Activity>
                  http://www.in-context.eu/pcsa#act1
            </ns1:Activity>
            <ns1:User>
                  http://www.in-context.eu/pcsa#Rossi.E54
            </ns1:User>
            </ns1:ctxtunnelling>
      </soapenv:Header>
      <soapenv:Body>
      ...
      </soapenv:Body>
</soapenv:Envelope>
```

Listing 1: Simplified example of SOAP header message including context coupling information.

Each service can then access context information, obtain additional context information or update the context from the *Context Management Framework* as follows. The context model is based on RDF, so we could – slightly simplifying – describe inContext's *Context Store* as a simple RDF store with added OWL and RDFS inferencing capabilities (to infer additional, implicit context information from ontological knowledge). This *Context Store* is queried and updated by the standard query language for RDF - SPARQL[5] and a recent extension for SPARQL to facilitate updates called SPARUL (SPARQL update language)[6].

It is worth to note that typical services nowadays available do not rely on RDF, but rather send "custom" XML messages wrapped in SOAP envelopes back and forth. In order to extract RDF content to update the context information in the *Context Store* from this message payload on the one hand, or enrich SOAP messages by context information in RDF on the other hand, we need to translate between customized XML formats and RDF. To this end, inContext developed its own query and update language based on XQuery, SPARQL and SPARUL. This novel language is called XSPARQL [Akhtar,et al. 2008] and allows integrating and enriching arbitrary XML message formats from context-aware services or even from legacy services with inContext's RDF-based context model.

4 Illustrating Examples

We will now consider the motivating examples presented in Section 2. Rather than providing details of XML and RDF data being passed around and queried, we stay at the more abstract (and readable level) and provide insight to the context coupling concepts.

In the scenario of the training course organization, first of all Mr. John adds the lesson with its teachers to the course program using the e-learning tool. When he invokes the calendar service it directly returns his and the teacher's calendars. The calendar service in fact exploits Mr. John context to understand that he and the teachers are involved in a joint activity. The same contextual information is used when Mr. John invokes the video conferencing tool to establish a video-conference with the involved teachers. If Mr. John selects a date on all the

[5] http://www.w3.org/TR/rdf-sparql-query/

[6] http://jena.hpl.hp.com/~afs/SPARQL-Update.html

teacher calendars, then his context is enriched with this "interesting date" and the e-learning service exploits this contextual information to assign a date to the new lesson. Mr. John only needs to confirm this date. In the same way the calendar service, exploiting the information that Mr. John has just defined a new activity (lesson) involving some teachers creates a new event to store in the calendars of the teachers. Finally the mail service, exploiting Mr. John contextual information (Mr. John has just created an activity involving the teachers and the students), automatically creates a mail template containing the addresses of the students to inform them about the new lesson. Thanks to context coupling each independent service does not work in isolation but is able to exploit Mr. John context (activity, partners, dates, location) to offer a customized and more useful set of functionalities.

In the second scenario, when Mr. John starts to work on the "Grenoble city library" his context is enriched with the information that he is involved in an activity with other users and also with the information that Grenoble is a relevant location for Mr. John's task. Exploiting this information the map is automatically centred on the city of Grenoble and his address book highlights the addresses of the others involved users. When Mr. John selects the address of Mr. Brown on his address book and then invokes the instant messaging service it initialises itself to enable Mr. John to quickly contact Mr. Brown. Finally when Mr. John invokes the shared folder service a suitable folder tree is automatically created using Mr. John activity information and other project members are automatically notified when Mr. John adds a new document.

5 Related work

The task-centric approach for self-adaptation in [Garlan et al. 2004] does not consider collaboration context based on joint activities. [Yang et al. 2006] focuses on the user's context and but does not consider the overall collaboration context. A context-aware resource recommendation system [Ning et al. 2007] uses ontologies to describe tools used in collaborations. In contrast, our context coupling approach considers interactions arising in collaborations as source for selecting the most relevant context.

The concept of activities itself is not new. Dustdar [Dustdar 2004] placed activities at the core of his Caramba system, a process-aware collaboration system supporting ad-hoc and collaborative processes in virtual teams, while IBM supported this idea in the Unified Activity Management (UAM) research effort [Cozzi et al. 2006, Moran 2005]. Both approaches use activities as a structural element. In particular, IBM's UAM is targeted more at enabling team-awareness, while Caramba enables process awareness. Both approaches use activities as a static structural element, while inContext considers the dynamic runtime status of activities as an important and meaningful element to set up context. Neither Caramba nor UAM manage runtime context coupling.

The most significant differences between our work and the related work presented above are twofold. First, we enhance applicability of context that is not purely user-centric as we enable the coupling of context from different collaboration services. Second, we ensure the complete decoupling of collaboration services, reflecting the dynamic behaviour of SOA-based CWEs. This allows simple and rapid integration of additional collaboration tools in SOA-based environments without having to rely on a central tool such as in [UAM].

To transfer reference of context between different collaboration services, we rely on techniques to manipulate SOAP header extension that are widely used in practice. Our implementation applies to inContext collaboration services based on SOAP. The OCA-WG (Open Collaborative Architecture Working Group[7]) aims at defining a reference architecture

[7] http://www.oca-wg.org

for collaboration services. Though no implementation of OCA-WG model exists now, we believe that our context coupling techniques can be applied to OCA-WG implementation when collaboration services are SOAP-based Web services.

6 Conclusion

In this paper, we have presented context coupling techniques implemented in the inContext project. Context coupling is an import issue to improve context-awareness and adaptation abilities of collaboration services. Our design-time and runtime context coupling techniques allow linking the relations among shared activities, team member's involvement and associated roles, and collaboration services in a SOA-based CWE, thus helping us to establish activity- and process-awareness.

Acknowledgement

This work has been partly funded by the European Commission through IST Project *inContext: Interaction and Context-based Technologies for Collaborative Teams* (http://www.in-context.eu). We thank all inContext members for their contribution on the discussion and the implementation of the context coupling.

References

Akhtar, W., Kopecky, J., Krennwallner, T., Polleres, A. : XSPARQL: Traveling between the XML and RDF worlds - and avoiding the XSLT pilgrimage. In Proceedings of the 5th European Semantic Web Conference (ESWC2008), Tenerife, Spain, June 2008. Springer.

Cozzi, A., Farrell, S., Lau, T., Smith, B., Drews, C., Lin, J., Stachel, B., and Moran, T. P. : Activity Management as a Web Service, IBM Systems Journal 45, No. 4, 695–712. 2006.

Dorn, C., Schall, D., Gombotz, R., Dustdar, S.: A View-Based Analysis of Distributed and Mobile Teams. WETICE 2007: 198-203. 2007

Dustdar, S. Caramba - A Process-Aware Collaboration System Supporting Ad Hoc and Collaborative Processes in Virtual Teams. Distributed and Parallel Databases, 15:1, 45-66, Kluwer Academic Publishers, Special Issue on Teamware Technologies. January 2004.

Garlan, D., Poladian, V., Schmerl, B., and Sousa, J. P.: Task-based self-adaptation. In Proceedings of the 1st ACM SIGSOFT Workshop on Self-Managed Systems (Newport Beach, California, October 31 - November 01, 2004). D. Garlan, J. Kramer, and A. Wolf, Eds. WOSS '04. ACM, New York, NY, 54-57. 2004

inContext D1.1: Analysis of Emergent Team Configurations and their Interaction Patterns - Methods and Mining Algorithms, inContext deliverable, http://www.in-context.eu/

inContext D4.1: Principles and Mechanism for "Context Tunnelling", inContext project deliverable. http://www.in-context.eu/

Moran T.P.: Unified Activity Management: Explicitly Representing Activity in Work-Support Systems. Proceedings of the European Conference on Computer-Supported Cooperative Work (ECSCW 2005), Workshop on Activity: From Theoretical to a Computational Construct. 2005.

Ning, K., Gong, R., Decker, S., Chen, Y., O'Sullivan, D.: A Context-Aware Resource Recommendation System for Business Collaboration. In Proceedings of the 9th IEEE Conference on E-Commerce Technology (CEC' 07) and the 4th IEEE Conference on Enterprise Computing, E-Commerce and E-Services (EEE ' 07) , IEEE Computer Society Press. 2007

Reiff-Marganiec, S., Yu, H.Q., Tilly, M.: Service Selection based on Non-Functional Properties, in Proceeding of NFPSLASOC 2007. (LNCS forthcoming).

Truong, H.-L., Dustdar, S., Baggio, D., Corlosquet, S., Dorn, C., Giuliani, G., Gombotz, R., Hong, Y., Kendal, P., Melchiorre, C., Moretzky, S., Peray, S., Polleres, A., Reiff-Marganiec, S., Schall, D., Stringa, S., Tilly, M., Yu. H.Q.: inContext: a Pervasive and Collaborative Working Environment for Emerging Team Forms, The 2008 International Symposium on Applications and the Internet (SAINT2008), IEEE Computer Society. 2008.

Yang, Y., Williams, M.H., Pooley, R., Dewar, R.: Context-Aware Personalization in Pervasive Communications. ICEBE 2006: 663-669. 2006