inContext: a Pervasive and Collaborative Working Environment for Emerging Team Forms*

Hong-Linh Truong[1], Schahram Dustdar[1], Dino Baggio[5], Stephane Corlosquet[3], Christoph Dorn[1], Giovanni Giuliani[6],Robert Gombotz[1], Yi Hong[7], Pete Kendal[8], Christian Melchiorre[2], Sarit Moretzky[9], Sebastien Peray[4], Axel Polleres[3], Stephan Reiff-Marganiec[7], Daniel Schall[1], Simona Stringa[2], Marcel Tilly[4], HongQing Yu[7]

[1]Distributed Systems Group, Vienna University of Technology
{truong, dustdar, dorn, gombotz, schall}@infosys.tuwien.ac.at

[2]Softeco Sismat SpA, Italy
{christian.melchiorre,simona.stringa}@softeco.it

[3]DERI Galway, Ireland
{stephane.corlosquet,axel.polleres}@deri.org

[4]European Microsoft Innovation Center, Germany
{Marcel.Tilly, speray}@microsoft.com

[5]Electrolux Home Products, Italy
dino.baggio@electrolux.it

[6]HP European Innovation Center, Italy
Giuliani@hp.com

[7]University of Leicester, UK
{yh37,srm13,hqy1}@le.ac.uk

[8]West Midlands LGA, UK
p.kendal@wmlga.gov.uk

[9]Innovation Lab,Comverse, Israel
Sarit.Moretzky@comverse.com

Abstract

Participants in current team collaborations belong to different organizations, work on multiple objectives at the same time, and frequently change locations. They use different devices and infrastructures in collaboration processes that can last from a few hours to several years. All these factors pose new challenges to the development of collaborative working environments (CWEs). Existing CWEs are unable to support emerging teams because diverse collaboration services are not well integrated or adapting to the team context. We present the inContext *approach to providing a novel pervasive CWE infrastructure for emerging team forms.* inContext *aggregates disparate collaboration services using Web services and Semantic Web technologies and provides a platform that captures diverse dynamic aspects of team collaborations. By utilizing runtime and historical context and interaction information, adaptation techniques can be deployed to cope with the changes of emerging teams.*

*This research is partially supported by the EU STREP project inContext (FP6-034718). We thank all members of the inContext consortium for their contribution on the development of the inContext environment.

1 Introduction

Traditionally, collaborative working environments (CWEs) provide a set of collaboration tools and services, such as email, document sharing, project management, etc., to assist people working collaboratively [16]. However, in these systems, collaboration tools and services are not integrated into a unified manner and cannot cope with changing collaboration contexts inherent in new teams forms. Typically, the user has to manually select the required tools/services and invoke them. The context and interaction of the collaboration have not been incorporated into such services. Therefore, the services cannot adapt according to team context and interaction, and such existing systems remain incapable to support emerging teams in highly dynamic environments.

However, the way people collaborate has been changed substantially due to the availability of new technologies. The recent advancements in mobile devices and network technologies have fostered a multi-objective and nomadic working style as well as ad-hoc collaboration. People within a team may work on different objectives and projects at the same time. Team members move from place to place during their collaboration. They use a variety of devices

and rely on diverse types of existing infrastructure. This leads to many new emerging team forms, such as *nimble* (short-lived collaboration to solve emerging problems), *virtual*(spanning different geographical contexts and having diverse professional members), and *nomadic* (collaboration with mobility capabilities) [21]. Thus, current CWEs should be capable of supporting the collaboration of such emerging team forms. However, there are many challenges in the development of CWEs suitable for emerging team forms. We recognize many issues:

- How can diverse collaboration tools and services built with different technologies be integrated so that they can be used in a unified manner?
- How can collaboration services adapt to the collaboration context of emerging team forms?
- How can human interventions in CWEs be reduced?

The new CWEs have to take into account the following aspects: highly dynamic and loosely-coupled infrastructures supporting different emerging team forms such as nimble, virtual, and nomadic. To leverage existing collaboration services for newly emerging teams, context and interaction should be utilized by those collaboration services. To this end, we have to integrate diverse services belonging to different organization and support context/interaction awareness. In this paper, we present the *inContext* environment, which introduces novel techniques and software for supporting adaptive, context aware collaboration within emerging team forms. The approach that the *inContext* project [12] follows is to utilize runtime and historical context and interaction information to adapt services for emerging team forms in real time. This paper presents an overview of the *inContext* environment and its main technical components and demonstrates a real-world example. The salient contributions of this paper are: (i) an advanced SOA-based collaborative working environment for emerging team forms, and (ii) context and interaction based techniques for adaptation in collaborative environments

The rest of this paper is organized as follows: Section 2 outlines the related work. Section 3 discusses the *inContext* approach. The architecture of the *inContext* environment is presented in Section 4. Section 5 presents the *inContext* context model. Interaction mining is discussed in Section 6, followed by service management in Section 7. Section 8 presents our experiments in a real world scenario to illustrate the *inContext* achievements. Section 9 summarizes the paper and outlines the future work.

2 Related Work

The research focus of the *inContext* project centers around how to exploit and combine novel techniques in the fields of context modeling and reasoning, service management, interaction mining, and service-oriented architecture technologies to develop novel CWEs for emerging team forms. Those research fields are already well-established, but their applications in CWEs are not well understood.

Basic collaboration services, such as document sharing (e.g., BSCW [2]), co-office (CoWord and CoPowerPoint) [3], calendars, instant messaging, etc., are not enough for emerging team collaboration. However, they are basic elements of which some are wrapped and integrated into the *inContext* environment.

The ECOSPACE project [4] aims at developing a CWE for eProfessionals. Similar to *inContext* , ECOSPACE also integrates various types of collaboration services. However, ECOSPACE focuses on collaboration services and tools integration for eProfessionals. ECOSPACE is mainly for individuals and it is based on a user-centric approach. *inContext* concentrates on team aspect (team-centric approach) by addressing context and interaction based technologies for emerging teams.The Kimura system [22] monitors user's interaction during the collaboration by integrating and providing various types of context information. However, Kimura is targeted to office environment and does not address issues posed by emerging team forms.

Existing context-aware middleware and applications provide and exploit various types of contextual information about location, time, user activities, user's preferences, profiles of users, devices and networks, etc., [18, 14, 20]. However, those models do not address the rich set of context information associated with collaborations. They mostly focus on user-related context and device capabilities which are utilized in the *inContext* context model. The *inContext* provides a combined model describing a rich source of information for advanced adaptation in collaboration

3 Approach

To support emerging team forms, we have to deal with several issues. First, we have to consider that team members stem from various organizations, they are using a wide range of collaboration services, some of which are publicly and freely available, while others are commercial products. How can we integrate these services? Second, teams, their activities and operating environments are pervasive and highly dynamic - how do we know their context? Third, interactions in emerging team forms are complex, so how do we measure and quantify metrics and patterns associated with interactions for service and team adaptation?

To answer the first question, our approach is to utilize SOA principle, especially Web services technologies, to integrate different types of collaboration services. With the SOA approach, collaboration services are loosely coupled, aggregated from different providers and inclusive of pub-

lic and free services. Collaboration services can be easily composed and adapted according to different needs of different teams. Furthermore, new collaboration services can be easily added into the system.

To answer the second question, we have to explicitly model context associated with emerging teams in detail. Such context is related not only to members, but also their activities and operating environments. Existing context can then be inferred and enriched to provide high-level information about activities and teams.

To answer the third question, we will rely on interaction mining, a technique that can be used to understand how interactions are performed in emerging team forms. We develop an in-depth analysis of human interactions and patterns. Based on that, interactions can be observed and meaningful patterns from observed interactions can be obtained and utilized.

4 Overview of the *inContext* Pervasive and Collaborative Working Environment

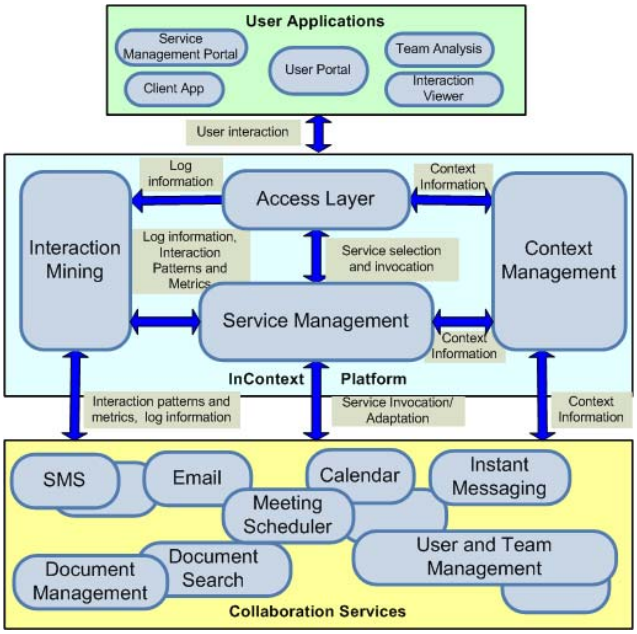

Figure 1. inContext architectural overview

Figure 1 depicts the *inContext* environment which basically comprises three main parts: *Collaboration Services*, *inContext Platform* and *User Applications*. *Collaboration Services* include services that are normally required in team collaboration. Such collaboration services are for document sharing (e.g., `Document Management` and `Document Search`), communication (e.g., `SMS` (Short Message Service), `Instant Messaging` (IM), `Email`, etc.), team and project management (e.g., `User and Team Management` and `Activity Management`). Those services could be specific to particular projects, but many are generic services which can be reconfigured to fit into particular purposes.

The *inContext platform* is the central part of the *inContext* project, where all aspects are brought together. This part includes novel services that support advanced, dynamic collaboration of emerging teams based on context and interaction model. The *Access Layer* acts as an intermediate, receiving requests from the client side and invoking services. The *Interaction Mining* is used to extract and analyze interactions inherent within team collaborations. The *Context Management* manages context associated with human, services, teams and activities. It supports reasoning mechanisms to both infer new context information and to enrich existing context information. The *Service Management* is responsible for selecting the right services, ranking and invoking the services according to requests from *Access Layer*. All the above-mentioned components can be deployed in and operate in a distributed manner.

The architecture of the *inContext* environment shown in Figure 1 is a reference implementation of the so-called *Pervasive Collaboration Service Architecture* (PCSA) that we have developed in the *inContext* project. By introducing new core services that support context- and interaction-based collaboration, the *inContext platform* is able to integrate various existing collaboration services to establish a network of PCSAs deployed in multiple organizations.

5 Context Management

Context information plays an important roles in adapting services suitable for emerging team forms. Unlike existing context-awareness systems in HCI (Human Computer Interaction) or location-based services which utilize a limited context information related to devices, user preferences, user presence and location, the context associated with human collaboration is much more complex. Context of emerging teams is typically associated with human (such as *person, organization, skill*, etc.), services (such as *SMS* and *Document Management*), location (e.g., *site* and *address*), teams (e.g., *membership role* and *department*), activities (e.g., *project* and *communication*), and interactions among human and services. Therefore, to describe the context model for *inContext* , we have not only to utilize many existing concepts and but also to develop new ones suitable for emerging team forms in a flexible manner.

Our approach in *inContext* is that we rely on ontology, namely RDF Schema [9] and OWL [6], to model context information. To obtain a flexible and widely usable context model, we reuse and extend existing ontologies, which are already being used on the

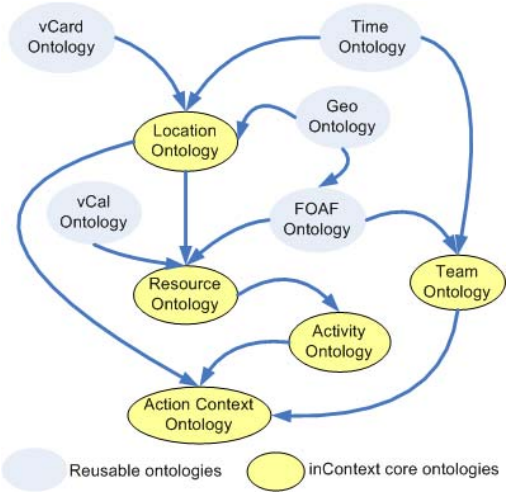

Figure 2. Structure of the *inContext* context model

Web. Figure 2 depicts the hierarchy of existing and *inContext* ontologies[1]. We partially reuse concepts in FOAF [5] (e.g., `Person, Organization, Group, Document` and `Project`) for modeling persons, organizations and their relations, vCard [10] (e.g., `Address`) for modeling addresses, Basic Geo [1] (e.g., `latitude` and `longitude`) for modeling geo-spatial context, vCal [8] (e.g., `VEVENT`) for modeling events, ResumeRDF [11] (e.g., `Skill`) for modeling skills and expertise of team members, and the Time ontology [13] (e.g., `Interval` and `Instant`) for modeling temporal context. These ontologies cover large parts of what is needed for describing user profiles, location information, time information, etc. In addition to those reusable ontologies, we develop five new core ontologies:

- *Location*: describes various fine-rained types of location information, including mobility, because Basic Geo and vCard ontologies are expressive enough to model relocation.
- *Activity*: describes the basic nature of activities and how they are related to users, resources, artifacts as well as other activities.
- *Team*: extends FOAF concepts to describe teams in more detail.
- *Resource*: describes usual input for an activity such as documents, services, and devices.
- *Action*: models the highly dynamic context that is subject to permanent changes.

[1]Current version of our context ontologies can be viewed at http://tyche.mcscw3.le.ac.uk:8080/RDF/showRDF.jsp

Based on the context model, we have developed a set of software sensors that capture relevant context information. The context information is captured and stored whenever context is changed. Context information is collected from various sources and is not necessarily stored at any central place. As shown in Figure 3, the *Context Management* subsystem does not store context information into a central repository. Instead, context information is stored into and retrieved from distributed services. A core model is held in a dedicated store within the *Context Management*, and from that model, different types of context information are linked by using RDF [7] instance data representing the current context. We emphasize that by adhering to RDF and OWL, the meta data and ontology language standard formats promoted by the World Wide Web consortium, our model is flexible enough to integrate with other ontologies published on the Web for CWE domain and applications.

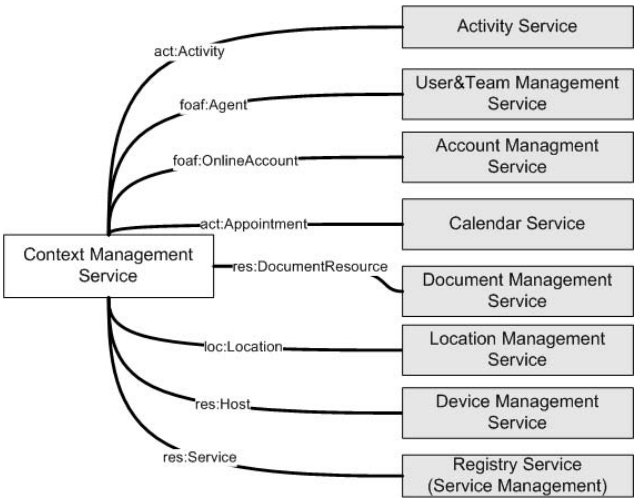

Figure 3. Sources of context information

Moreover, by using ontologies, context information can be inferred based on rules in order to provide value-added information about the context associated with people, teams, services and activities. Our context reasoning techniques are built on a SPARQL++ engine developed on top of the dlvhex system [17] which processes ontological context data collected in the *Context Management*. For example, let's assume we want to setup a team of civil engineers on demand for work at a particular site. To find suitable engineers, the following SPARQL query can be used.

```
PREFIX team:<http://www.in-context.eu/team.owl#>
SELECT ?engineer
WHERE{
  ?engineer :hasProfile ?profile.
  ?profile :hasSkill ?skill.
  ?skill :name ?sname.
  ?engineer :locatedAt :''Genoa sea port''
FILTER regex(?sname,"civil engineer","i") }
```

Any services and clients can invoke the *Context Management* component to query context information by specifying a so called *context requirements description* (CRD) which consists of (i) a SPARQL query for extracting the relevant context from the *Context Management* component and (ii) an XSLT or XSPARQL [15] transformation which translates the extracted RDF context data to an XML format consumable by the services and clients. Furthermore, context reasoning techniques can be used to aggregate context information from external sources, and evaluate and query rules defined over context information.

6 Interaction Mining

Quantitative information associated with interactions can be used to enrich context information as well as be used as inputs for the service selection and ranking. The *Interaction Mining* is used to detect metrics and patterns associated with interactions. Because in emerging team collaboration many activities are defined on demand without any pre-defined processes, interactions are detected from services and activities events based on complex event processing techniques. Various types of interactions associated with human and services are inherent within collaborative environments. In *inContext* we consider three kinds of interactions

- *Service-to-service*: the interaction between two services, e.g., a service might call another service.
- *Human-to-service*: the interaction between a human and a service, e.g., how services are selected and used by a team.
- *Human-to-human*: the interaction between human and human, e.g., how a team member interacts with another one in order to perform activities.

For each type of interactions, interaction mining is applied at multiple levels such as individual (humans or services), group (a team or a set of services), and the collaboration (all available services and/or teams). In order to provide metrics and patterns associated with interactions, we have collected log information of collaboration services and performed the mining online. Table 1 presents an example of metrics and patterns associated with interactions that can be detected and provided by the *Interaction Mining*. Metrics and patterns will be determined by corresponding plugins which process pattern specifications. Using aggregation techniques, higher level metrics and patterns, such as a network of service interactions, can be determined from lower level ones, such as number of interactions and callees. The mining results produced by the *Interaction Mining* are described in XML and can be subscribed or queried by any clients.

7 Service Management

Collaboration services readily available in the PCSA can complement each other or compete against each other. For example, two providers can provide two services with a similar functionality or they can provide services that together exhibit a larger function. Each particular collaboration instance might require different kinds of services, depending on the context. Adaptation is centered around how to use context and interaction information as well as service information to select suitable service instances for the collaboration. The *Service Management* implements the selection of the most suitable service based on four sources of information: context information, interaction metrics and patterns, user preferences, and service meta-information.

While context and interaction information can be obtained from the corresponding components, the service meta information has to be managed by the *Context Management*. In doing so, we have to integrate different kinds of meta-information associated with services. We developed a service meta information model used to relate different types of information associated with services, based on which service selection is performed. The model, defines a service category to indicate the type of services, such as `SMS` and `DocumentSharing`. Operations offered by services are mapped into one or more categories. For each service operation, a set of criteria represents the meta-information. A criteria is represented as a quadruple `(name, type, value, weight)`, indicating the name of the criteria, the data type, value of the criteria, and weighted factor, respectively. A value of a criterion can also be a SPARQL query, like the presence of a person. For example, the system is providing a category named `sendNotification` which can be associated with the following criteria:

name	type	value	weight
availability	String	(online, busy, offline)	0.25
communication type	String	(IM, EMail, phone)	0.25
message priority	Double	0..3	0.5

In this example values representing potential values of user's context. Basically these are expressed and stored by using a SPARQL query so that we are able to find the most-relevant service during runtime. For this example, let's assume that there are three operations (`sendIMMessage, sendMail, sendSMS`) maybe from different services associated with this `sendNotification` category providing specific meta data. By adding more criteria it is possible to express complex selection rules. In this example the selection rules look as follows:

```
IF availability is online or (availability is busy
and message priority <= 1) THEN
        sendIMMessage
IF availability is busy THEN
```

Interaction/Level	Individual	Group	Collaboration
Service-to-service	Number of invocations, number of unavailability, number of failures, number of consumers	Usage distribution, usage mode (isolated or composite) patterns, service interactions network	Usage distribution, usage mode (isolated or composite) patterns
Human-to-service	Number of service invocations, usage mode (isolated or composite) patterns	Usage distribution, constant/-durable/limited duration usage patterns	Usage distribution, constant/-durable/limited duration usage patterns
Human-to-human	Number of callers/callees, number of interactions, number of assigned activities	Team size, total interactions, average number of callers/callees, interaction networks	Broker, proxy, master/slave, co-authoring patterns, interaction networks

Table 1. Examples of interaction metrics and patterns

```
        sendMail
IF availability is offline THEN
   sendSMS
```

The actual ranking process involves multiple steps: First the *Service Management* picks up the right service categories by using a simple keyword matching algorithm. Next based on service meta-information, the services are ranked. Then, the best service is selected based on its rank. The reasoning step is performed by sending requests to the *Context Management*. For ranking services, we have developed an modified LSP (Logic Scoring of Preference) algorithm [19]. Interaction metrics are more complicated, as they consider how well services work with each other; they are typically used in workflow scenarios. However, details are beyond the scope of this paper.

8 Application Examples

Currently, we have achieved the first prototype of the *inContext* system. The *inContext* platform and various collaboration services are deployed in different sites in Aachen, Genoa, Leicester, Milan and Vienna. A system like *inContext* can be used for many purposes. In this section, we particularly illustrate how context and interaction information can be used to solve the "meeting scheduling problem".

8.1 The Meeting Scheduling Problem

Our illustrating example is the meeting scheduling application. During team collaboration, planning a meeting is a task that is frequently required. At the first glance, this application looks simple: just retrieving calendars from team members then performing the scheduling based on the availability date of team members. However, the meeting scheduling is much more complex due to several constraints and requirements as team members are working on highly dynamic environments and on the move. We identified three main steps in a meeting scheduling:

- selecting suitable time and participants: the context of team members and team work will be utilized in order to determine suitable time and participants.
- preparing documents: the context of activities will be needed in order to prepare document templates.
- sending notification/changes: the context of team members and the information about existing communication services will be needed.

The above-mentioned steps can be fully automatically solved by the *inContext* environment by utilizing context reasoning, rules, and service selection. For example, for each step, we defined some policies for the meeting scheduling. The following policies illustrate some necessary rules for the meeting scheduling scenario from a real-world use case introduced by Electrolux:

- *Meeting priority & attendance*: the following rules are used to specify meeting priorities and attendance requirements:

```
IF meeting priority = High THEN
    Attendance type = Physical
    Travel for meeting = True
    Proxy participation = At the same level
    Attendance Quorum = All
ELSE IF meeting priority = Medium THEN
    Attendance type = Any (Physical | Phone | Video)
    Organizer attendance = Physical
    Travel for meeting = False
    Proxy participation = At the same level or
                          one level below
    Attendance Quorum = At least 1 for each L2 type
    (i.e. 1 EL, 1 MEC, 1 LAB)
ELSE IF meeting priority = Low THEN
    Attendance type = Any (Physical | Phone| Video)
    Organizer attendance = Any
    Travel for meeting = False
    Proxy participation = At the same level
                          or one level below
    Attendance Quorum = At least 50% of invited
ENDIF
```

- *Notifications of the planned meeting or when the meeting is changed*: the selection rules in Section 7 can be used to send the notification when a meeting is planned.

The meeting scheduling has many more rules but we just illustrate above-mentioned rules in the next section.

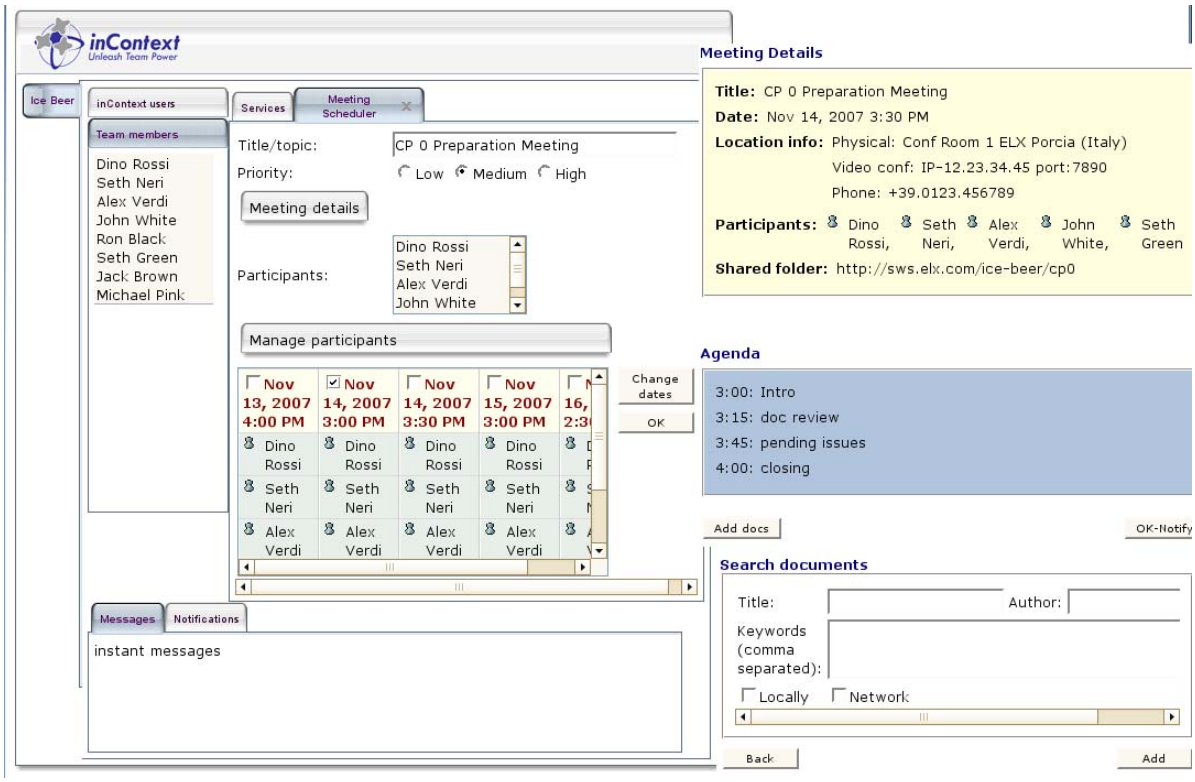

Figure 4. Steps in scheduding a meeting

8.2 Context- and Interaction-based meeting scheduling

Figure 4 depicts the user interface for scheduling a meeting. From the user point of view, it is relatively simple to plan a meeting. The user can select the topic of the meeting, and search and add participants manually or specify expertise or role based on that participants can be selected. The *inContext* environment will automatically recommend available date for the meeting by checking context related to the availability of participants. When the user agrees on the date, *inContext* will create necessary document template for the meeting as well as reserve resources for the meeting based on the availability of participants (e.g., suggesting a location for face-to-face meeting). Finally, notifications will automatically be sent to the participants based on their presence status. However, from system perspective, many complex issues and human interventions have been reduced by utilizing context and interaction information.

First, for example, consider the case in which the meeting priority is `LOW`. In this case, a timeslot is valid where at least half of the invited participants are available. The following query is used by *inContext* in order to find possible time slots for the meeting.

```
PREFIX iCal: <http://www.w3.org/2002/12/cal/ical#>
SELECT ?T
WHERE {<m1> :possibleTimeSlot ?T ; :priority "low".
   ?T time:hasBeginning ?TB; time:hasend ?TE.
FILTER( COUNT{?P : { <m1> :invited ?P }} >=
   2 * COUNT{?P :
   { <m1> :invited ?P .
   ?P :hasCalendar ?C .
GRAPH ?C { ?E a iCal:Vevent;
  ical:dtstart ?B
  ical:dtstart ?E. }
FILTER( ( ?B >= ?TB && ?B <= ?TE )
  || ( ?E >= ?TB && ?E <= ?TE ) )
}
```

Second, consider how the *inContext* finds relevant documents for the meeting. Depending on the purpose of the meeting, document templates can be retrieved and put into a dedicated directory for the meeting

```
PREFIX res: <http://www.in-context.eu/resource.owl#>
PREFIX act: <http://www.in-context.eu/activity.owl#>
PREFIX
  rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT ?resoure ?meeting
{
   ?meeting rdf:type act:Activity.
   ?meeting :shortname "review meeting"^^xsd:string.
   ?meeting :usesResources ?resource.
   ?resource rdf:type res:DocumentRepository.
}
```

Furthermore, `DocumentSearch` service can be invoked

to search for existing documents available in personal and network directories. The found documents can be then associated with the meeting.

Third, consider how the system uses the correct communication to send the notification. A participant Rossi might not be online at the time the notification should be sent, so *inContext* must use context information to determine which type of communication should be used. The following reasoning is used to check the status of Rossi before sending a notification.

```
PREFIX ctx: <http://www.in-context.eu/context.owl#>
SELECT ?x ?y
WHERE{
 ?a ctx:connectedBy ?x .
 ?x ctx:hasOnlineStatus ?y .
 ?y ctx:status ?z .
}
```

Assume that it turns out that user Rossi is currently not online with any Instant Messaging service and we must notify him via SMS. Having the notification sent via SMS, the *Service Management* can even perform a service ranking and select the cheapest SMS provider based on existing service meta-information and interaction metrics.

9 Conclusion and Future Work

In this paper, we described the *inContext* pervasive and collaborative working environment. Motivated by the lack of suitable CWEs for emerging team forms, the *inContext* project has introduced novel techniques to integrate existing collaboration services and context and interaction-based collaboration. Based on context and interaction, advanced features can be supported, making *inContext* suitable for different collaboration purposes, ranging from mobile to nomadic to ad-hoc. In this paper, we presented the main components that make *inContext* unique, as well as an illustration of a real-world example.

Nevertheless, there is room for improvements. One aspect is to investigate how interaction patterns can be used in team adaptation. As current users/teams management is performed by a centralized service, how *inContext* connects different users/teams management services belonging to different organizations, to create or utilize a virtualization of users/teams management systems, will be investigated.

References

[1] Basic Geo (WGS84 lat/long) Vocabulary, http://www.w3.org/2003/01/geo/.

[2] BSCW (Basic Support for Collaborative Work), http://www.bscw.de/english/index.html.

[3] CoOffice Suite, http://cooffice.ntu.edu.sg.

[4] ECOSPACE: eProfessionals Collaboration Space, http://www.ip-ecospace.org.

[5] FOAF Vocabulary Specification 0.91,http://xmlns.com/foaf/spec/.

[6] OWL - Web Ontology Language Reference. http://www.w3.org/TR/owl-ref/.

[7] RDF - Resource Description Framework. http://www.w3.org/RDF.

[8] RDF Calendar Workspace, http://www.w3.org/2002/12/cal.

[9] *RDF Vocabulary Description Language 1.0: RDF Schema, http://www.w3.org/TR/rdf-schema/.*

[10] Representing vCard Objects in RDF/XML, http://www.w3.org/TR/vcard-rdf.

[11] ResumeRDF Ontology Specification, http://rdfs.org/resume-rdf/.

[12] The inContext project,http://www.in-context.eu.

[13] Time Ontology in OWL,http://www.w3.org/TR/owl-time/.

[14] G. D. Abowd, A. K. Dey, P. J. Brown, N. Davies, M. Smith, and P. Steggles. Towards a better understanding of context and context-awareness. In *HUC '99: Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing*, pages 304–307, London, UK, 1999. Springer-Verlag.

[15] W. Akhtar, J. Kopecky, T. Krennwallner, and A. Polleres. 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.

[16] G. Bafoutsou and G. Ment. Review and functional classification of collaborative systems. *International Journal of Information Management*, 22(4):281–305, August 2002.

[17] A. Polleres and R. Schindlauer. dlvhex-sparql: A SPARQL-compliant query engine based on dlvhex. In *2nd International Workshop on Applications of Logic Programming to the Web, Semantic Web and Semantic Web Services (ALPSWS2007)*, volume 287 of *CEUR Workshop Proceedings*, pages 3–12, Porto, Portugal, Sept. 2007. CEUR-WS.org.

[18] M. Raento, A. Oulasvirta, R. Petit, and H. Toivonen. Contextphone: A prototyping platform for context-aware mobile applications. *IEEE Pervasive Computing*, 4(2):51–59, 2005.

[19] S. Reiff-Marganiec, H. Yu, and M. Tilly. Service selection based on non-functional properties. In *NFPSLASOC 2007*, LNCS. Springer, 2007. (in press).

[20] M. Solarski, L. Strick, K. Motonaga, C. Noda, and W. Kellerer. Flexible middleware support for future mobile services and their context-aware adaptation. In F. A. Aagesen, C. Anutariya, and V. Wuwongse, editors, *INTELLCOMM*, volume 3283 of *Lecture Notes in Computer Science*, pages 281–292. Springer, 2004.

[21] T. van Do, I. Jørstad, and S. Dustdar. *Handbook of Research on Mobile Multimedia*, chapter Mobile Multimedia Collaborative Services, pages 414–429. Idea Group Publishing, 2006.

[22] S. Voida, E. D. Mynatt, B. MacIntyre, and G. M. Corso. Integrating virtual and physical context to support knowledge workers. *IEEE Pervasive Computing*, 1(3):73–79, 2002.