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

Hong-Linh Truong[1], Schahram Dustdar[1], Dino Baggio[5], Christoph Dorn[1], Giovanni Giuliani[6], Robert Gombotz[1], Yi Hong[7], Peter 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
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

The way people collaborate has been changed substantially. Team members are now belonging to different organizations, working on multiple objectives at the same time, frequently changing their locations, using different devices and infrastructures in their collaboration processes lasting from few hours to several years. This poses many challenges to the development of collaborative working environments (CWEs). Existing CWEs are unable to support emerging teams because in those CWEs, diverse collaboration services are not well integrated and adapted suitable to the team context. This paper presents the **inContext** *approach to providing a novel pervasive and collaborative working environment for emerging team forms.* **inContext** *aggregates disparate collaboration services using Web services technologies and provides a platform that is capable of capturing diverse contexts and interactions inherent in team collaborations. By utilizing runtime and historical context and interaction information, various adaptation techniques can be achieved to cope with the changes in collaborations.*

*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 performing collaborative work [15]. However, in those systems, collaboration tools and services are not integrated into a unified manner, thus they do not cope with the change of collaboration contexts inherent in new teams forms. Typically, the user has to manually select adequate tools/services and invoke them. The context and interaction of the collaboration have not been taken 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 are now working on different objectives and projects at the same time. Team members are moving from places to places during their collaborations. They are using a variety of devices and relying 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) [20]. 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 observed many issues:

- How can diverse collaboration tools and services built with different technologies be integrated so that the user can use them 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?

Adequate 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 to support context/interaction awareness. In this paper, we present the **inContext** environment, which aims at introducing 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 on the fly. This paper presents an overview of the **inContext** environment, discusses its main technical components and presents an illustrative real-world example. The salient contributions of this paper comprise

- an advanced SOA-based collaborative working environment for emerging team forms
- 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 is 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 a novel pervasive collaborative working environment 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 [21] 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, 19]. 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

Recently, there is a growing interest in exploiting autonomic computing techniques [16] to achieve self-management properties. Self-management techniques have made significant progress, but they have not been considered in contemporary CWEs. The **inContext** addresses the service adaptation in CWEs by applying context reasoning and service ranking techniques to automate the selection of teams, activities and services, thus substantially reducing user interventions.

3 Approach

To support emerging team forms, we have to deal with several issues. First, we have to consider that team mem-

bers 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 those 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. 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, including public 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 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 collaborations of teams. The *Context Management* manages context associated with human, services, teams and activities. It supports reasoning mechanisms to infer new context information and can enrich existing context information. The *Service Management* is responsible for selecting the right services, ranking the services 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.

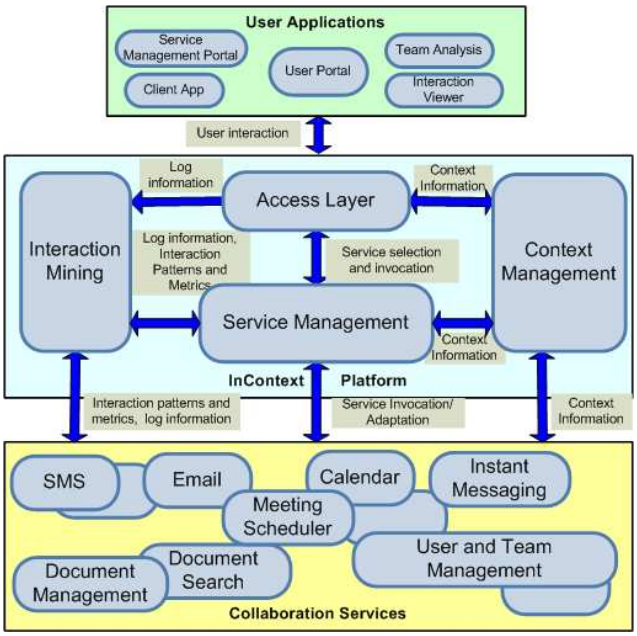

Figure 1. inContext architectural overview

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 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 will be 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.

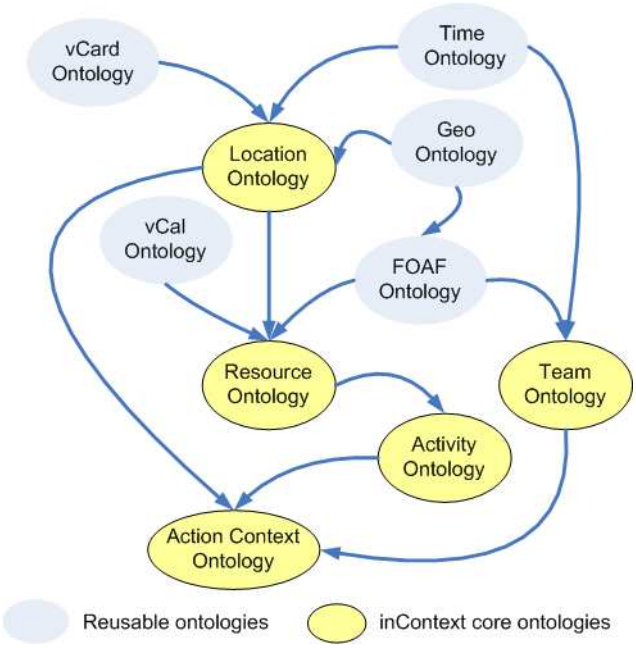

Figure 2. Structure of the inContext context model

Our approach in **inContext** is that we rely on ontology to model context. Namely, we adopt 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 Web. Figure 2 depicts the hierarchy of existing and **inContext** ontologies. 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 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 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

Based on the context model which is made up by the described network of ontologies, we have developed a set of software sensors that capture relevant context information. The context information is captured and stored whenever context is changed accroding to the sensors. Context information is collected from various sources and is not necessarily stored at any centralized 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 stored 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. By using RDF and OWL, our model is flexible enough to integrate with other ontologies published on the Web for CWE domain and applications.

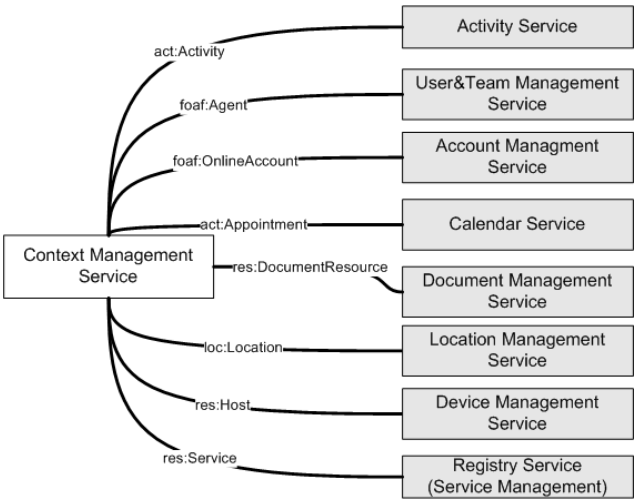

Figure 3. Sources of context information

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* to query context information. specifying a so called *context requirements description* (CRD) which consists of (i) a SPARQL query for extracting the relevant context from the *Context Management* and (ii) an XSLT which translates the extracted context data to an XML forma 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

Understanding interaction among team members and services sheds the light on characteristics of team members, for example, the role of a team member, which type of communications a team member prefers, and the performance of a service. Quantitative information associated with interactions can then be used to enrich context information as well as be used as inputs for the service selection and ranking.

Because in emerging team collaboration many activities are defined on demand without any pre-defined processes, interactions are detected from log information based on correlation techniques. Various types of interactions associated with human and services are inherent within collaborative environments. We categorize three kinds of interactions

- *Service-to-service interaction*: the interaction between two services, e.g., a service might call another service
- *Human-to-service interaction*: the interaction between a human and a service, e.g., how services are selected and used by a team.
- *Human-to-human interaction*: 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 interaction, interaction mining is applied at multiple levels such as individual (human or service), group (a team or a set of services), and the whole system (all services and/or teams). In order to provide metrics associated with interactions, we have collected log information of collaboration services and performed the mining. Table 1 presents an example of metrics associated with interactions that can be detected and provided by the *Interaction Mining*. Using aggregation techniques, higher level metrics can be determined from lower level ones.

The amount of information provided by the Interaction Mining is vast and the information ranges from low-level, such as historical metrics associated with a service, to high-level, such as detected patterns associated with a team. To provide such information to *Context Management* and *Service Management* as well as other clients, the *Interaction Mining* provides APIs and languages for accessing the information through Web services. We are currently working on a query language that allows the client to specify *concepts* in **inContext** ontologies and *duration* for which the *Interaction Mining* should provide mining information associated with the concepts.

7 Service Management

In the PCSA there are many collaboration services readily available. Services can complement or compete each other, for example two providers can provide the two services with the same function. However, each particular collaboration instance might require different kinds of services, depending on the context. The key of adaptation is centered around how to use context and interaction information and service information to select suitable service instances for the collaboration. The *Service Management* is not only for managing collaboration services but also for selecting the right service based on the context. To this end, three sources of information are used by *Service Management*: context information, interaction information, and service meta-information.

While context and interaction information can be obtained from 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 that service selection is performed. In this model, we first define a service category to indicate the type of services, such as `SMS` and `DocumentSharing`. Then, operations offered by services are mapped into one or more categories. For each service operation, a set of criteria will be used to represent the meta-information about service operation. A criteria is represented as a quadruple `(name, type,`

Interaction/Level	Individual	Group	The whole system
Service-to-service	Number of invocations, number of unavailability, number of failures, number of consumers	Usage distribution, usage mode (isolated or composite) patterns	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, number of callee, number of interactions, number of assigned activities	Team size, total interactions, average number of callers, average number of callees	Broker, proxy, master/slave, co-authoring patterns

Table 1. Examples of interaction metrics and patterns

`value, weight)`, indicating the name of the criteria, the data type, value of the criteria, and weighted factor, respectively. For example, an `SMS` service provides an operation named `sendSMS` which can be associated with the following criteria:

name	type	value	weight
cost	double	1.3 EUR	0.25
reliability	double	1.0	0.75

Based on context information, interaction information, and service meta-information, the *Service Management* performs the selection and ranking of services. This involves multiple-steps. First, using context reasoning, the *Service Management* picks up the right service categories. Next based on service meta-information and interaction metrics, the services are ranked. Then, the best service is selected based on its rank. The reasoning step is performed by sending request to the *Context Management*. For ranking services, we have developed an modified LSP (Logic Scoring of Preference) algorithm.

8 Experiments

Currently, we have achieved the first prototype of the **inContext** system. Various collaboration services are provided such as calendar, Email, instant messaging, etc. The **inContext** platform and collaboration services are deployed in various sites in Aachen, Genoa, Leicester, Milan and Vienna. A system like **inContext** can be used for many purposes. In this session, 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 **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
```

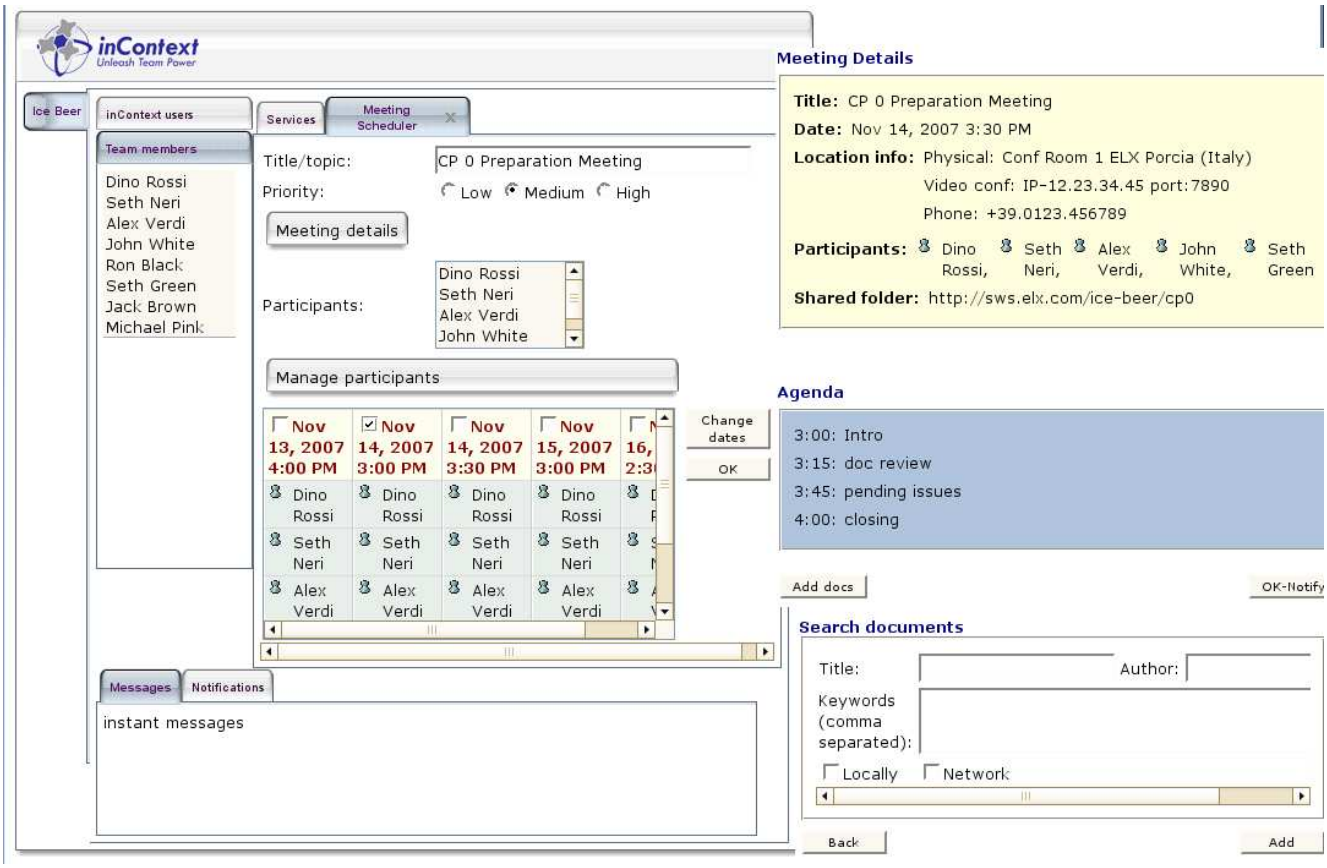

Figure 4. Steps in scheduding a meeting

```
                                or one level below
    Attendance Quorum = At least 50% of invited
ENDIF
```

- *Notifications of the planned meeting or when the meeting is changed*: the following rule can be used to send the notification when a meeting is planned.

```
Always send MAIL with Full Details
IF present on Instant Messaging (IM) THEN
      send summary as IM message
ELSE
      send summary using SMS
ENDIF
```

Of course, the meeting scheduling has many more rules. However, we just illustrate rules that we will present in the next section.

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., if some are available in face to face - the system will suggest a physical room or if some are available using video conf - the system will suggest IP and port of the video conf application). Finally, notifications will automatically be sent to the participants based on their presence status. However, from system point of view, 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 and added into the dedicated directory.

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. Therefore, **inContext** has to 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, nomadic to ad-hoc ones. In this paper, we presented the main components that make **inContext** unique as well as a real-world example.

Still there is space for improvements. One aspect is to investigate how interaction patterns can be used in team adaptation. Currently 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] G. Bafoutsou and G. Ment. Review and functional classification of collaborative systems. *International Journal of Information Management*, 22(4):281–305, August 2002.

[16] J. O. Kephart and D. M. Chess. The vision of autonomic computing. *Computer*, 36(1):41–50, 2003.

[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 (ALP-SWS2007), 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] 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.

[20] 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.

[21] 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.