An Application Framework for Seamless Synchronous Collaboration Support in Ubiquitous Computing Environments

Seunghyun Han, Niels A. Nijdam, and Nadia Magnenat-Thalmann

MIRALab, University of Geneva, 7, rte de Drize, 1227, Geneva, Switzerland
{han,nijdam,thalmann}@miralab.ch

Abstract. Dynamic and heterogeneous nature of ubiquitous computing environments introduces additional requirements to support synchronous collaboration. Such requirements include support of various interaction types, flexible data couplings, and transparent context adaptation. To meet those requirements, in this paper, we propose the manipulation based application model. In comparison to the presentation semantics split model [5], we introduce the manipulation in between the presentation and shared semantics. A manipulation is a fragment of the semantics, which is dynamically created when a presentation requires personalized interaction to the shared semantics. A manipulation enables transparent context adaptation by migrating its states to a new manipulation of the different presentation to adapt the current context, e.g., user location change. We prototyped the proposed application framework and tested the feasibility of the framework.

Keywords: Synchronous collaboration, ubiquitous computing, application framework, context adaptation.

1 Introduction

With the proliferation of networks and computers, the vision of ubiquitous computing [1][10] is realized in everyday living environments such as a home and an office where is often populated by multiple users simultaneously. Within such an environment, collaboration among a group of users is conducted to accomplish a common task, exploiting a wide range of devices in many different shapes, sizes and computing capabilities while moving around in the environment. There have been several research efforts [2][6] on nomadic users to collaborate with each other. They assume that each participant uses a dedicated device with similar capabilities, e.g., mobile phone, PDA, notebook, and user mobility is handled at the network level e.g., handoff management [8]. However, this does not hold for ubiquitous computing environments anymore. It means that, while participating in collaboration, each participant does not need to stick to one device and can change their devices for better utilization of available devices. For instance, a user prefers to conduct a video conferencing with a large wall-display rather than a small display on PDA as he enters to his living room from outside, adapting to dynamic context changes [3].

F. Alvarez and C. Costa (Eds.): UCMEDIA 2010, LNICST 60, pp. 141–149, 2012.

To successfully support such dynamic adaptation, Computer Supported Collaborative Work (CSCW) technologies are faced with a new challenge: to support dynamic changes of interaction devices without losing the consistency of shared information in the presence of context changes. To meet the new requirement, in this paper, a new application model extending the presentation and semantic split model is proposed. In this model, a collaborative application is separated into five parts: semantics, presentation, manipulation, context adaptation, and session control. The semantics represents the shared data in a collaborative application. The presentation acts as a function for the interactive control between users and the shared semantics. It transforms the shared states into a form that is perceivable by a user. The manipulation provides methods to process shared semantics. Multiple different manipulations can be dynamically coupled to a subset of the shared data in the semantics, so that the data in the shared semantics can be processed in different ways by the coupled manipulations. The main role of the manipulation is to maintain the consistency between the shared semantics during any changes of presentations while adapting to the transition of devices. Two types of sessions are defined, a user session and collaboration session. A user session is defined as 'a group of application components, devices on which application components are resided and their bindings which are used by a user within a given time'. A collaboration session is composed of a set of user sessions in the current environment. The context adaptation triggers changes of a user session based on current context. Context adaptation is different 'user by user' as well as 'environment by environment'. We prototyped the proposed application framework with the runtime support and built several applications based on the framework.

The remainder of this paper is organized as follows. In section 2, we describe the requirements for multi-user collaboration in ubiquitous computing environments. In section 3, we describe the proposed application model. Section 4 describes the implementation of our prototype system and its applications respectively. Conclusion is followed in section 5.

2 Requirement Analysis

One of the key characteristics of ubiquitous computing environments is the dynamic change of context of users and environments. This introduces two key technical requirements to support seamless collaboration among users in ubiquitous computing environments. First, users are not just limited to utilize one dedicated device, e.g., mobile phone, PDA, and notebook, but have more freedom to exploit diverse and different set of devices while participating in a collaborative session. It requires a polymorphic presentation adaptation to overcome the resource heterogeneity of the client devices while providing suitable responsiveness to users. This means that the presentation to a user, which includes view and interface, i.e., frame size, frame rate and user interfaces, has to be dynamically adapted to the current device context such as processing power, memory size, display size, and network condition at runtime. Secondly context changes cannot be fully predefined at the application design stage,

because the context can be different for each user, as well as from one environment to another. It requires the ability to embracing the context changes but still clearly separate the context adaptation from the application semantics. The context adaptation must be transparent to the target user as well to all other participants.

3 The Proposed Approach

A collaborative application is a mean by which users are able to perform a common collaborative task. The proposed model is specifically designed for building collaborative applications; with the goal that it provides application-developers with a suitable abstraction that overcomes the heterogeneity of devices and diversity of contexts in ubiquitous computing environments.

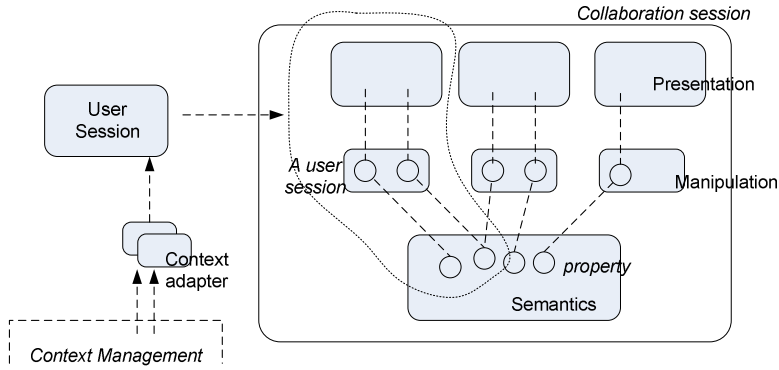


Fig. 1. The proposed application model. It is composed of the five elements: semantics, presentation, manipulation, context adapter, and session.

The proposed application model defines five elements: semantics, manipulation, presentation, session, and context adapter as shown Fig. 1. The semantics, manipulation, and presentation are the application base-level building blocks and are strictly related to the application domain functionality. The session manages the composition of the three base-level components and implements the application meta-level. It stores information about the user session in the current environment. A context adapter monitors any context changes in the environment and changes session composition accordingly.

3.1 Presentation Tier

In ubiquitous computing environments, we cannot simply assume that users share the exactly same view of the shared data for collaboration due to the following reasons. First, devices have different capabilities in terms of display and processing resources in an environment. Secondly, the available devices can be changed due to context changes. Last, users can exploit a diversity of devices to participate into collaboration

based on the users' preferences. We handle the situations in presentation tier. The presentation tier can be seen as a bag of *presentations*. It maintains a set of *presentations* and is responsible for accessing and manipulating the *semantics* through corresponding *manipulation*. Each user has its own presentation tier; within this tier each *presentation* is specific to one device. A *presentation* acts as a function of an interactive control between users and functionality which transforms application data in the shared *semantics* into a perceivable form. This implies that each *presentation* should be developed specifically to fit a device, because each device has different capabilities (e.g. display and processing capacity). The *presentation* receives user-actions as events and responds to these 'actions'. Multiple different *presentations* are selectively coupled with specific data within the shared *semantics* through *manipulation*. Each *presentation* selectively registers its interests to a specific *manipulation*. Whenever a specific state of the shared *semantics* is changed a notification is send back to the *presentation*. For instance, each user can browse through a presentation file independently and view different pages. Whenever a user changes a page, this state change does not need to be distributed to all *presentations* but only reflect to the target *presentation* that currently represents the changed page.

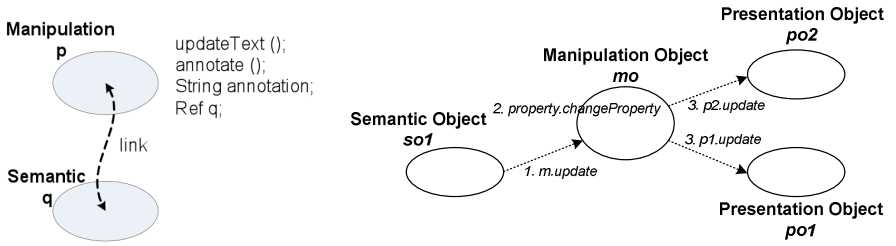
3.2 Semantics Tier

The semantics tier represents the shared data in the collaborative application. It does not contain any application logics or states pertaining to the environment or user specific contexts. If the preferences of a user or presentation-specific characteristics are incorporated into the semantics, an application is specialized to a specific user or a target presentation. Moreover, it is almost impossible to expect that the application developers can anticipate all the different states of presentations.

Unlike the model in distributed MVC model [6][7][9], we explicitly decouple manipulation functionality from the shared data. This allows users to share only data with different functionality and representations, so called 'polymorphic collaboration'. The semantics is responsible for notifying any states changes in the shared data. It can support various forms of interactions by dynamically binding different types of manipulations to the shared semantic. It persistently keeps its states, which allows users to continue a collaborative task by reincarnating the shared *semantics* when users re-initiate the collaborative task.

3.3 Manipulation Tier

The manipulation tier maintains a set of manipulations and is responsible for linking *presentations* and *semantics*. A *manipulation* specifies how the shared *semantics* is manipulated by *presentations*. We define a *manipulation* as follows. Let q be a semantic and p be a manipulation for q. p provides functions for manipulating q. p can have additional attributes and methods. Correspondence between p and q is represented by its *link*, which enables update propagation between p and q. A *link* between p and q could be dynamically mapped if necessary.



(a) (b)

Fig. 2. (a) represents a relationship between semantics and manipulation. (b) represents m:1 relationship between manipulation and two presentations.

Fig. 2(a) shows an example of the relationship between a *manipulation* and a *semantics*. A *semantics q* provides presentation material *q.material* as a shared data. A *manipulation p* can have reference of *q* as its *link*. It provides *p.updateText* method to change *q.material*. A *manipulation p* also defines additional property *p.annotation* and additional method *p.annotate* for personal manipulation. Each *manipulation* can provide different functionality of a shared *semantics*. It implies that multiple different *manipulations* are bound to a shared *semantics* so that the shared *semantics* can be manipulated in different ways. The *manipulation* support *m*:1 relationship (more than one presentation per manipulation) as shown in Fig. 2(b). It is useful when multiple devices for a user share the same functionality of a collaborative application. For example, in case of presentation manager application, a user can review the same slide with a desktop display (*po1*) and a wall-display (*po2*) in his room while other users are collaborating in their own rooms (different environment). A personal annotation made by a user (*mo*) cannot be distributed other participants and it also does not change the shared data (*so1*). In this case, the manipulation notification should not be distributed to other participants, thus reducing the number of interaction events.

3.4 Context Adaptation Tier

Collaborative applications in ubiquitous computing environments are influenced by external changes that affect the composition of the collaborative application at runtime, and therefore collaborative applications require support in order to adapt to these context changes dynamically. It usually alters the application composition according to the current context of the users and environment.

Figure 3 outlines the basic flow of a context adaptation process as an example of location context adapter. Each time a user moves from one place to another, the context manager invokes the notify method of the registered location context adapter. The location adapter obtains a list of available devices from presence manager in the execution infrastructure. Next, it uses the interfaces of the target user's session, in order to change between devices and to migrate states. This is specific to the target environment and may not be used to another environment. A context adapter also can change a specific property of *manipulation*, *presentation*, and *semantics* using the target user's session, if the developer knows specific properties of them.

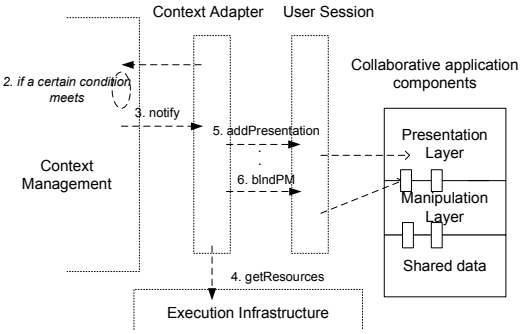


Fig. 3. Dynamic context adaptation process

3.5 Collaboration Session Control Tier

We define two types of sessions, a user session and collaboration session. A user session is defined as 'a group of application components and devices on which the application components are resided, and their bindings that are used by a user within a given time'. Collaboration session is composed of a set of user sessions in the current environment. Collaborative session control such as dynamic user join and leave is managed by collaborative session control and context adaptation is control by a user session.

A user session maintains meta-level information of currently exploiting devices and software components of a user. It is dynamically changed at runtime when a target user changes devices or application components adapting to a context change. It also persistently preserves the session states such as device ID and references of software components when a user leaves the current collaboration session. This persistently preserved session states are reused when the user re-joins to the collaborative session. It reduces user's distractions to manually reconfigure the application components.

4 Implementation

Fig. 4 shows the overall architectures of the runtime environment of the proposed framework.

The session description is environment independent so that it lists the components of a session and their requirements. It contains a list of entries that describe each required component. Each entry contains name-value pairs to specify the component name and type, and the resources required by the component. When users create a collaborative session or a new user joins the session, a list of matching resources are obtained by system services. Context adaptation is user specific so that the each user session manages its own context adapters. The context adaptation logic is different from one environment to another so that the context adaptation types are provided with a separate description. A Collaboration Session Manager (CSM) is generic to all kinds of collaborative applications and only one exists in an environment. It is a kind

of abstract application factory. It receives an abstract session description. It interacts with system services to create environment specific applications. Its key roles are to create application component instances, bind them together, and pass the result, which contain meta-information of the user session, to the Session Manager. A Node Manager mirrors the CSM, but is limited to only one node. When a device is entered into an environment the Node Manager is registered to Presence Manager so that the device participates in the environment. It is a daemon-like process that is always running on each individual node and responsible for spawning an Object Reconfiguration Manager (ORM). It creates an ORM whenever lifecycle management requests are invoked. It also instantiates the Migration Manager when migration of an application object which currently runs on the node is requested. The Object Reconfiguration Manager is responsible for managing the lifecycle of objects, which includes create, destroy, suspend, resume, save and restore states of objects. To create a presentation object, it first checks if the target object is preserved in the State Repository. If it exists, it restores the object; otherwise, it creates a new presentation object and its corresponding manipulation objects by downloading executables of the target object. Then, it registers presentation and manipulation objects to the Node Manager.

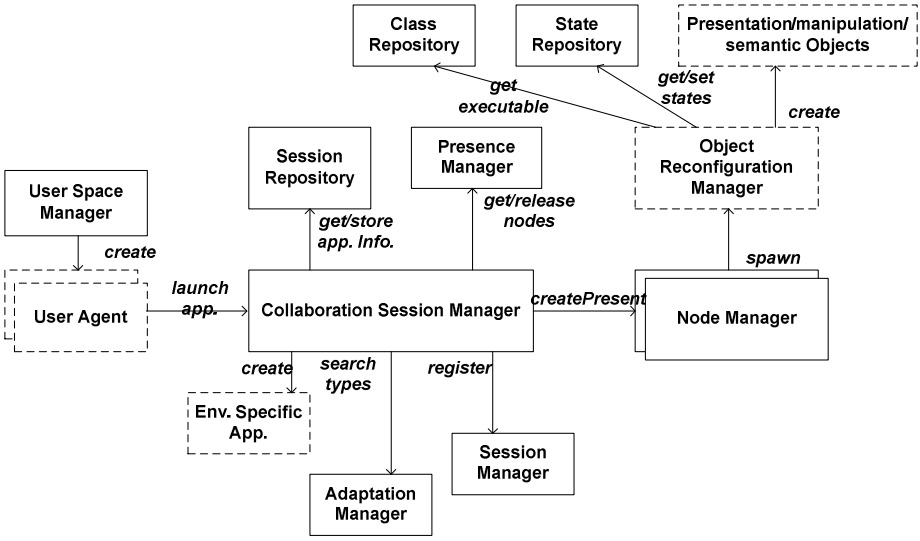


Fig. 4. Runtime support of the proposed application framework

We built a collaborative telemedicine system for real-time and interactive segmentation of volumetric medical images [4] based on the proposed application framework. In this application, users can utilize diverse devices, i.e., mobile phone, UMPC, notebook, interactive table, to collaborate with each other as shown in Fig. 5. Users can also seamlessly collaborate with each other while moving from one place to another with minimal distractions from the computing devices.

Fig. 5. Collaborative telemedicine system built based on the proposed application framework

5 Conclusion

One of key differences of synchronous collaboration in ubiquitous computing environments from mobile computing is that users can dynamically change their devices adapting to current device availability. The key to enable it is to provide tools to simplify the development of collaborative applications that easily adapt dynamic context changes. Furthermore, these collaborative applications must hide details of dynamic changes in the environment to the end users in order to minimize involvement of the users. In this paper, we proposed an application framework that provides a standard way for the development of collaborative applications and adapts dynamic context changes, e.g., user mobility and resources availability changes. Transparent context adaptation, which supports dynamic changes of interaction devices without losing the consistency of shared information with minimal user involvement in the presence of context changes, is also addressed. Experience so far shows that the proposed application framework is appropriate to support transparent context adaptation of a user without having interference among users.

Acknowledgments. This work is supported by the InterMedia (38419) project in the framework of the EU IST FP6 Programme.

References

1. Abowd, G., Mynatt, E.: Charting past, present, and future research in ubiquitous computing. ACM Transactions on Computer-Human Interaction 7(1), 29–58 (2000)
2. Cheng, S., Huang, A., Garlan, D., Schmerl, B., Steenkiste, P.: Rainbow: Architecture-Based Self Adaptation with Reusable Infrastructure. IEEE Computer 37(10), 46–54 (2004)
3. Dey, A.: Providing Architectural Support for Building Context-Aware Applications, PhD thesis, College of Computing, Georgia Institute of Technology (December 2000)
4. Han, S., Nijdam, N., Schmid, J., Kim, J., Magnenat-Thalmann, N.: Collaborative telemedicine for interactive multiuser segmentation of volumetric medical images. The Visual Computer Journal 26, 639–648 (2010)

5. Keremitsis, E., Fuller, I.: HP Distributed Smalltalk: A Tool for Developing Distributed Applications. Hewlett-Packard Journal, 85–92 (1995)
6. Marsic, I.: An architecture for heterogeneous groupware applications. In: Proceedings of the 23rd International Conference on Software Engineering, pp. 475–484 (2001)
7. Roman, M., Hess, C., Cerqueira, R., Ranganathan, A., Campbell, R., Nahrstedt, K., Gaia: A Middleware Infrastructure to Enable Active Spaces. IEEE Pervasive Computing Magazine 1(4), 74–83 (2002)
8. Talukder, A., Yavagal, R.: Mobile Computing: Technology, Applications, and Service Creation. McGraw-Hill Professional (2006)
9. Ulmer, B., Ishii, H.: Emerging Frameworks for Tangible User Interfaces. IBM Systems Journal 39(3/4) (2000)
10. Weiser, M.: The Computer for the 21 Century. Scientific American 265, 94–101 (1991)