

INFERS: An Infrastructure for Experience Record in Smart Spaces

Weisheng He, Yuanchun Shi, and Weijun Qin

Department of Computer Science, Tsinghua University, Beijing, PR China
hws99@mails.tsinghua.edu.cn, shiyc@tsinghua.edu.cn,
qinweijun99@mails.tsinghua.edu.cn

Abstract. The experience record in smart space is an useful service for education activities, either for the purpose of course review or multimedia courseware creation. In this paper, we present the concept of collaborative experience record according to the new feature of pervasive computing paradigm. A supporting platform is implemented to enable the rapid prototyping of experience record applications in smart space. A group of common functions for experience record are contained in the platform and can be directly used in the application. Standard interfaces are also provided to evolve the information sensing services in smart space so that they can cooperate with experience record application at runtime. We have prototyped an augmented classroom application based on this platform and user study results are also presented in the paper.

Keywords: Smart Space, Experience Record, Supporting Platform, Live Courseware Creation.

1 Introduction

The aim of pervasive computing paradigm is the transparent integration of technology in the physical environment so as to remove the barriers prohibiting the efficient use of such information services. Smart space is a typical testbed to explore the various techniques relating to this paradigm. With the abundant information services embedded in smart space, future educational activities can be substantially augmented. Experience record, as one thrust of smart space, is an important service in future learning environment since the captured record of live classroom experiences can be employed either for course review or courseware creation.

There has been a considerable amount of work focusing on the experience record in smart space, such as classroom capture¹²³ or meeting room capture⁴⁵. However, a major drawback is the lack of reusability of their systems. Closely related to some specific application environment, these works turned to be ad hoc solutions to specific environments. Although it is widely believed that the underlying barriers that hinder the reusability of experience record applications, such the difference in hardware and software or application modes, are not likely to be eliminated in the near future, part of the functional modules are common in different applications. In such modules can be encapsulated in an underlying platform which provides standard interfaces to develop the application-specific modules, the prototyping of experience record application in smart space can be much more efficient.

Besides, the unique features of smart space which are likely to effect the experience record are hardly explored in previous works. Thus previous works are not very adaptive or compatible to the smart space environment.

In this paper, we discuss the principles unique to smart space that govern the design of experience record applications, which has led to the concept of collaborative experience record. Based on this concept, a supporting platform, INFERS (INfrastructure For Experience Record System) is implemented to enable the rapid prototyping of experience record applications in different smart space environments. To testify the usability of the platform, a classroom application is implemented on the platform as a prototype.

The paper is organized as follows: in section 2 we present a brief introduction of smart space and its unique features that are likely to effect the design of experience record application; the concept of collaborative experience record is discussed in section 3. The implementation details of INFERS are presented in section 4. Section 5 demonstrates the classroom application developed on the INFERS platform and a user study to this prototype is discussed in section 6. Finally, section 7 concludes the paper and outlines the future work.

2 Smart Space

Smart spaces are working environments with embedded computers, information appliances and multi-modal sensors allowing people to perform tasks efficiently by offering unprecedented levels of access to information and assistance from computers⁶. In smart space, the information world is seamlessly integrated with the physical world, where humans can interact with machines through natural, multi-modal interfaces and are served by the information services proactively.

Thus, the pervasive computing paradigm in smart space has brought many new features to the design of experience record applications, including:

- Distributed processing. The information devices are distributed and connected by network and should be coordinated and synchronized.
- Dynamicity and mobility. The environment is highly dynamic: the number and property of information services involved in the experience record may change from time to time and mobile devices are free to join or leave the session at any time.
- Rich sensing services. The seamlessly integration of the information world and physical world require the sensing of human activities by the machine. Multifarious sensing devices, such as microphone array, location tracker and touchable whiteboard, are thus embedded in smart space. These services can be leveraged by experience record application to capture the information of the live experience.
- Unobtrusiveness. The applications running in smart space should require the minimum of user attention and the services should be provided proactively.

With the above features in mind, we believe the previous design mode of experience record applications are not appropriate to be employed in smart space,

especially in terms of mobility support and leverage of the rich sensing devices in smart space, triggering the concept of collaborative experience record.

3 Collaborative Experience Record

Previous experience record applications¹⁴⁵⁷⁸⁹ are mostly designed in stand-alone mode: they are functionally closed and all modules are self-contained, having little or no interoperability with external services at runtime. Stand-alone applications tend to be ad hoc solutions since some of the functions are heavily dependent on the application environment. One example is the data capturing module where the data type and hardware setup must be clearly defined prior to development. When the environment changes, the experience record applications following the stand-alone mode can not adapt to the evolution.

In order to solve the problem, K. Truong et al. in ¹⁰ proposed the idea of INCA, an infrastructure for experience capture and access. INCA contains a set of abstractions of the common functions in experience record application and systems can be developed by inheriting the abstract JAVA classes provided by INCA. This idea materially aids the development of experience record applications.

However, if common functions are provided in the form of directly usable modules instead of JAVA classes, the application implementation can be even more efficient. Besides, the rich sensing services in smart space can also be leveraged by experience record application. These two considerations have informed the concept of collaborative experience record, which emphasize the loosely-coupled cooperation between record application and sensing services in smart space at runtime. The sensing services are not necessarily dedicated to experience record. For instance, location tracking does not only provide user location data to experience record, but may also be used for other purposes such as context-aware computing and user preference mining. These services, termed as external friend entities, can be employed to provide the functions that are specific to different environments.

Based on the concept of collaborative experience record, we can implement a supporting platform to contain the common modules that will be used across different environments. The supporting platform will also rely on the external friend modules

	Live Record	Access
Application Data Layer	Capture	Storage Access
Coordination Layer	Coordination	
Transmission Layer	Communication	

Fig. 1. Layered model and function modules for experience record applications

to minimize the development efforts needed in a specific environment. To put this idea into reality, further investigations are required to clarify the modules and their interrelations. A three layered functional model is thus proposed as in Figure 1.

As we can see, the modules in an experience record application can be divided into five classes: capture, coordination, communication, storage and access. They can be organized in a matrix-like structure according to the function reliance and temporal stages. From Figure 1 we have the following observations. Firstly, only the capture module is directly related to the application data and internally unreusable. Secondly, the data flow is from top to bottom (functional direction) and from left to right (temporal direction) and module functions will only be called by modules in upper layer, e.g., the functions in coordination layer will only be called by modules in application data layer but not by those in transmission layer. In light of these observations, we have the conclusion that all the modules, except the capture module, can be implemented as common solutions in a supporting platform, provided that the data flow has a standard format.

4 INFERS

In this section, we will discuss the mechanisms adopted by the platform design. INFERS adopts multi-agent and is implemented on the Smart Platform1112, a software infrastructure for smart space. To build the supporting platform on the software infrastructure of smart space will facilitate a better interoperation between the experience record application and other smart space services. All the modules involved in experience record at runtime are encapsulated as agents to realize the loose-couple design principle. Figure 2 illustrates the overall architecture for experience record applications built on INFERS, and external friend entities are represented by gray rectangles.

The agents in live experience record can be divided into source agent and sink agent. Source agents function as the data capturing clients, while the server-like sink

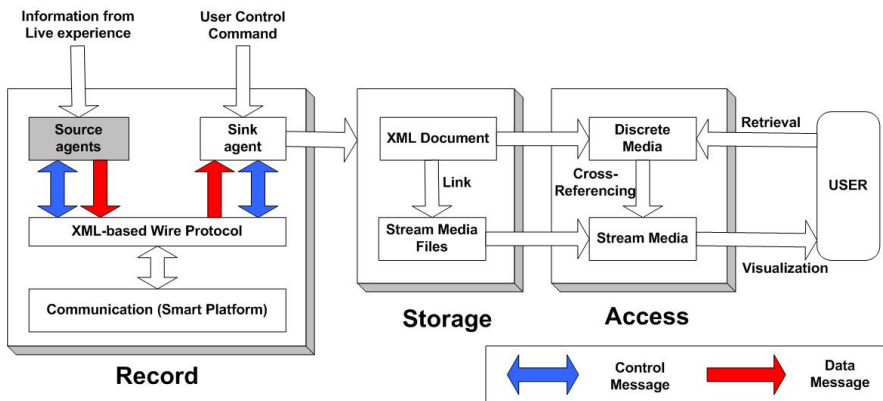


Fig. 2. Overall architecture for applications built on INFERS

agent gathers the data and submit them to persistent storage. To simplify the structure, coordination function is also implemented in the sink agent. Besides the external friend entities, some reusable source agents are also implemented in INFERS, including video/audio capture and computer screen capture.

4.1 Coordination Mechanism

Messages are leveraged for coordination in INFERS and four types of control message are used as shown in Figure 3. Source agents subscribe messages published by the sink agent, and vice versa. The following part describes this mechanism in detail.

After the record is started, the sink agent keeps a timer and at each timeout publishes a <RecordStart> message containing the session information to all agents through multicast. When a source agent receives this message for the first time and decides its data are to be recorded, it publishes a <RecordStartAck> message, which contains the metadata of the source agent, as a reply to register itself at the sink agent, which maintain a list of the registered source agents. When record is stopped in the end, the sink agent will again publish a <RecordStop> message. On receiving this message, the source agent checks whether all the data that are to be recorded have been successfully transmitted to the sink agent. If so, it will reply to the sink agent with a <RecordStopAck> message telling it is ready to quit. The sink agent will keep a timer after publishing the <RecordStop> message and continue gathering the data, until all registered source agents have reported ready to quit or timeout occurs.

As the <RecordStart> message is continuously published, a source agent dynamically joining the session sometime after it begins will also receive a <RecordStart> message. The offset information (it informs how long the record has lasted) contained in the message content enables the data generated by this latecomer

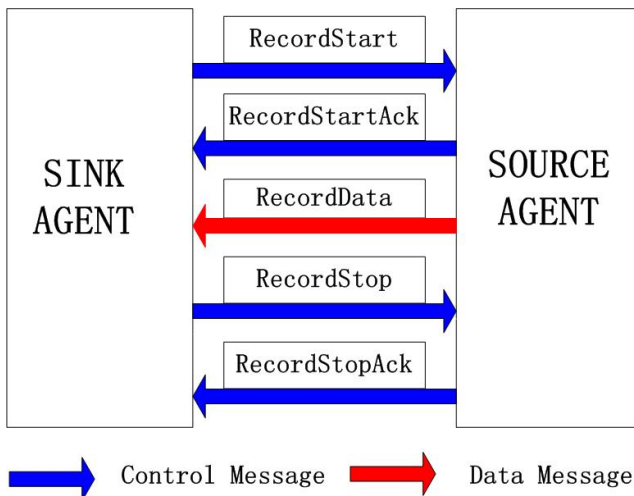


Fig. 3. Hand-shake negotiation and the message exchange

to be synchronized with other existing media. Similarly a source agent can leave before the record is stopped by actively publishing a <RecordStopAck>. Through such a hand-shake negotiation, INFERS can support the unpredicted changes during the live experience, allowing the mobile users to join, participate freely in a highly dynamic fashion.

4.2 Data Modalities

The captured information can be stored in two data modalities: stream media and discrete media. Stream media is a continuous, unstructural form of media, such as video and audio. Discrete media is comprised of structured units, each of which can independently express a semantic meaning, like a multimedia document.

The same information can be represented by both data modalities. For instance, a slide presentation can be captured either as a video clip, or a XML document containing the URLs of the slides. Hence there exists a choice on which media type should be used to present the captured information. Considering the one principle mentioned in last section that data flow should follow a standard format, we argue that stream media be used to visualize the recorded experience and discrete media be used for retrieval. This is because:

- For the purpose of experience rendering, stream media has relatively standard formats like MPEG or AVI. Discrete media is high diversified and there is no standard format or interface to render the discrete media. Thus, stream media should be leveraged for experience rendering to maintain standardization.
- Stream media is more comprehensive and coherent, containing the context information around the important points during live experiences. For instance, from a video showing how a decision is made, we can probably see the reaction of the participants to the decision through their facial expressions. However, discrete media is usually captured without saving the context information.
- A traditional disadvantage discouraging designers from leveraging stream media is its large size. However, this concern is becoming increasingly trivial with the rapid development in data compression and hardware facilities,. For example, while using the TSCC free trial version as the compressor to capture the computer screen displaying presentation slides, the bit rate of the generated video is about 13kbps, meaning that the video file which can visualize the slides of a one-hour presentation will be around 6MB in size.
- While stream media is suitable as the visualization approach, discrete media is ideal for indexing, retrieval and summarization because the structured semantic can be understood and processed by computers.

Due to these reasons, users should retrieve the desired information through discrete media and the retrieval result is used for cross-referencing into stream media based on the temporal synchronization. The indexed clips of stream media will present the information to the user. The interoperations between them are illustrated in Figure 4.

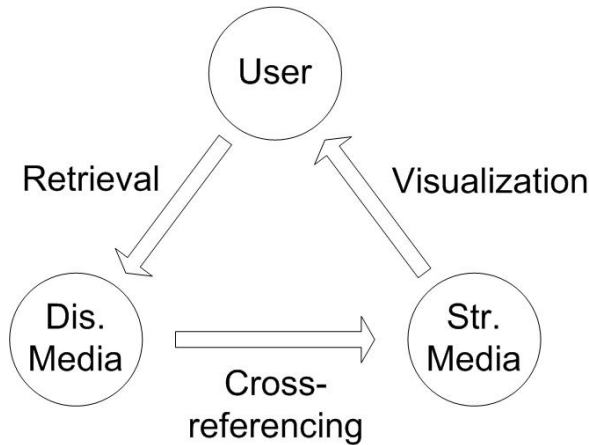


Fig. 4. User, discrete media, stream media and their interoperation

4.3 Communication and Storage

Communication mechanisms in INFERS are provided by Smart Platform, which supports three mechanisms: message transmission, streaming transmission and bulk transmission.

Corresponding to the two data modalities, the communication and storage also adopts two different mechanisms, respectively. Stream media is transmitted to sink agent by bulk transmission which is implemented on the FTP protocol.

```

<Event>
<User>Mike</User>
  <Action>annotate</Action>
  <Object>Ontology.ppt</Object>
  <Begin>41</Begin>
  <End>46</End>
<Location>Room527:iBoard</Location>
</Event>
  
```

Fig. 5. The XML content of an event

Discrete media, mainly the information of meaningful events captured by external friend entities, are sent in `<RecordData>` messages via XML-based wire protocol. Since the external friendly entities are involved, the data format should be standard, extensible and flexible. Therefore we use the communication service provided by the infrastructure of smart space and the plain-text XML structure as the format. While transmitting event information, source agents generate a XML element containing the

information and convert it into binary bytes using XML-based wire protocol. These binary bytes are transmitted as the content of <RecordData> messages and parsed at the sink agent using the same protocol to extract the semantic information of the event. Figure 5 is the XML content of an event.

Similarly, XML is used in INFERS to store live experiences. Each session is represented as a single XML document conforming to a specific schema and an excerpt is shown in Figure 6. XML documents are stored in Berkeley DB XML.

```
<Session version="1.0">
  <Context>
    <Duration>2503</Duration>
    ....
    <SessionName>GroupMeeting2</SessionName>
  </Context>
  <Streams>
    <Stream Src="D:\video files\2005_12_22_OverviewVideo.avi" Name="OverviewVideo" />
    ...
    <Stream Src="D:\video files\2005_12_22_13_45_27_eMapVideo.avi" Name="eMapVideo" />
  </Streams>
  <Events>
    <Event EventID="0" User="Shi" Action="talk" Object="Schedule" Begin="5" End="17"/>
    <Event EventID="1" User="Chen" Action="openDoc" Object="Ontology.ppt" Begin="36"
End="38"/>
    ...
    <Event EventID="4" User="Li" Action="talk" Object="N/A" Begin="99" End="107"/>
    <Event EventID="5" User="Jiang" Action="talk" Object="OWL language" Begin="111"
End="132"/>
  </Events>
</Session>
```

Fig. 6. An excerpt of the record document

The document root contains three child elements: context, streams and events. <Context> element contains the general information about the live experience, including time, place, users involved and the main subject (if there is one) etc, to provide an overview of the recorded experience.

<Streams> element represents links to external media files relating the stream media and the metadata about the stream media. Each of its children, the <Stream> element, is related to one data stream.

The <Events> element represents the discrete media to describe meaningful events which will be used as indices. Each of its children, the <Event> element, contains the semantic information of an event in five dimensions: time, location, user, action and object.

4.4 Access Interface

INFERS provides an access interface, shown in Figure 7, for users to efficiently visit the records. In Figure 7, the left-most window lists all the record documents stored in the database. Two time windows are placed in the upper part: the upper window demonstrates all indexing events in the live experience record as blue icons (the selected event will turn red). The lower window is scaled to present the events in a

specific period corresponding to the slider position of the slide bar below, so that users can accurately manipulate the icons. The three windows in the middle are used to render the stream media. By clicking on the window, user can watch the video content in a much larger pop-up window. The three windows in the lower-most part are for retrieval purpose. The left one demonstrates the semantic information contained in the selected event. The middle one lists all the previously stored retrieval queries and the results are displayed in the right-most window.

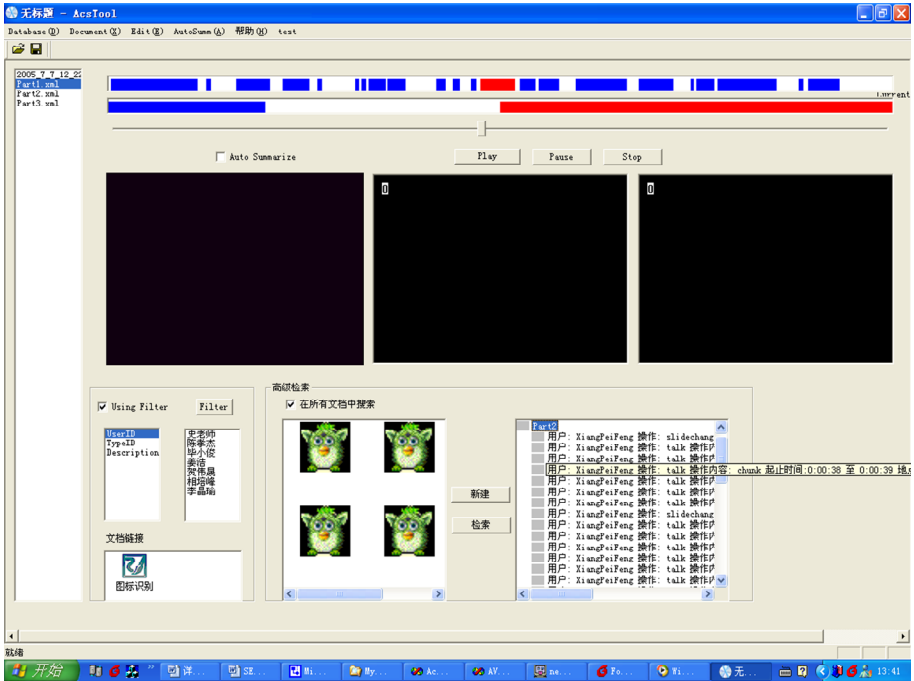


Fig. 7. A snapshot of the access interface

The playback of the record can be sequential as traditional video ad controlled by the slide bar. It also can be fast-forward to rewind to a specific position by the icons in the time window or the entries in the retrieval result window. Thus, users can have flexible methods to access the information.

To further improve the access efficiency, we have also implemented in INFERS an information model for semantic retrieval and an auto-summarization algorithm. Since details of the information model and summarization algorithm are already discussed in other papers¹³¹⁴, we just give some brief introductions here.

The information model has four components: primitive ontology, query expression model, archive expression model and query processor. The primitive ontology, which uses OWL DL 15 language to describe the basic semantic primitives in the application scenario and their interrelations, is the foundation of semantic reasoning in the retrieval. Semantic queries are formalized by the query expression model through SQL-like syntax and can be defined in wizard-like interface. The query

expression model structurally stores and expresses the experience information, just as what is described in section 4.3. The query processor matches the user query and experience information formalized by query expression model and archive expression model. A reference engine, the core of the query processor, will make necessary semantic reasoning in the matching process.

The summarization algorithm is based on the previous user access patterns. A statistical model is leveraged to analyze the pattern data to predict the user interests and pick up the important segments accordingly. This algorithm has two advantages: no priori knowledge is required so that it can be used as a common solution in different application environments; the analysis is conducted in a personalized manner so different users can be served with summaries tailored to their own needs.

5 Prototype: Augmented Classroom

Based on the INFERS platform, we have implemented a classroom prototype. This prototype is an augmented version of our previous Smart Classroom 2project. More multi-modal interfaces, which can function as the external friend modules, are embedded in the prototype, including:

- RFID-based Location tracker. The tracker can report location events such as user entering or leaving the classroom. When the tracker detects that a teacher enters, it will publish a control message to start the experience record.
- Microphone Array. The microphone array can detect the events of user speech and keywords contained in the speech. When combined with the location tracker, the identity of the current speaker can be derived (the speaker location detected by microphone array can be matched to the user location data).
- Touchable display. User can open documents and annotate on the display. Predefined icons can be recognized online. All these data recorded.
- Interaction with remote students. Remote students can use either desktop computer or PDA to access the augmented classroom. The access information, including login, logout and interactions with local users are recognized by the software module.

All these modules are established on Smart Platform and encapsulated as agents. Thus, the efforts to evolve them into external friend modules are generally easy. The Smart Platform SDK provides an abstract CAgent class, available both in C++ and Java, for the encapsulation. The necessary operations include instantiating a CAgent object, calling its Register() method at the beginning and Quit() method in the end. About twenty lines of code are added to realize the communication function of external friendly modules.

Other modules involved in experience record, including coordination, communication, storage and access, are contained in the supporting platform and can be directly leveraged in the prototype application. The augmented classroom prototype is then employed to generate multimedia courseware through experience record in authentic environment for some courses.

6 User Study

To testify the usability of the prototype system, a user study was conducted after its authentic use in a course of our department. About 160 undergraduates from our department are involved in the course. Among them, 20 students participated in the live classroom and the rest 140 students used the courseware created through live experience record to conduct self-study. After the class, we distributed questionnaires to the students and asked them if they agreed that the courseware created through experience record were significantly better than the traditional courseware like PowerPoint or HTML. Scores were ranged from 5 to 1, corresponding to “strongly agree”, “agree”, “neutral”, “disagree” and “strongly disagree”, respectively.

The average score was 3.7025, which was interpreted as a slight preference for the live-experience-based courseware. The result was out of our expectation since we believed this courseware was apparently more vivid and attractive. To find the reason, we investigate the comments given by the students whose rating was 1 or 2. They mainly complained that the courseware, although powerful and comprehensive, was not easy to use. They had to first install the access interface, then download the courseware and finally learn to manipulate the interface. These requirements greatly discourage the access to the courseware. From this point, we can conclude that the interface should be friendly enough to be accepted by the users. For instance, if the interface can be evolved as controls that are embedded in websites, then it will be much easier for students to access the information since no downloading or installation efforts are needed.

Some major positive comments are as follows:

- Compared with traditional education video, the record of real classroom experience is much more attractive. For instance, when the teacher raises a question, I will reflect by myself, wait for the answer given by other students and compare it with mine. Then I will listen to the teacher’s explanation and this process is very impressive.
- I can tailor the lecture to my own needs. I can skip the unimportant part and focus on the difficult questions. I do not have to get up early and still can enjoy the real classroom experience.
- The new mode is not as efficient as ordinary classes since we need some time to adapt to it. But gradually students will accept the new mode. The blackboard data are very useful. In ordinary classroom it is sometimes difficult to watch the blackboard clearly since the classroom is too big. Now I can definitely have a very clear view!

From these comments, we can see that the live-experience-based courseware is advantageous in bringing more reality and comprehensive content. If the problem in user-friendliness can be solved, it is likely to be welcomed by users. Besides, once the experience record system has been mounted, this method is much more cost-effective since no extra effort is needed.

7 Conclusions and Future Work

In this paper, we present our work on a supporting platform for experience record applications in smart space. The supporting platform is the outcome of the

collaborative experience record concept, which is proposed so that such applications can comply with the pervasive computing paradigm in smart space. A group of common function modules are implemented in the platform. Interfaces to evolve application-specific modules are also provided.

Given the lessons we have learnt from the user study, our primary focus will be to evolve the access interface so that users will not be discouraged due to the extra efforts needed to access the courseware.

Acknowledgement. The work is Supported by the Key Project of Chinese Ministry of Education (No.106016) and the National Development Project for the Next Generation Internet (CNGI-04-15-3A).

References

1. Abowd, G.D.: Classroom 2000: An Experiment with the Instrumentation of a Living Educational Environment. *IBM Systems Journal*, Vol. 38(4). (1999) 508-530.
2. Shi, Y., Xie, W., Xu, G., Shi, R., Chen, E., Mao, Y., Liu, F.: The Smart Classroom: Merging Technologies for Seamless Teleeducation," *IEEE Pervasive*, Vol 2., no. 2, (2003) 47-55.
3. Mukhopadhyay, S., Smith, B.: Passive Capture and Structuring of Lectures, *Proc. ACM Multimedia'99*, (1999) 477-487.
4. Heather, R.A., Abowd, G.D., Geyer, W., Fuchs, L., Daijavad, S., Poltrock, S.: Integrating Meeting Capture and Access within a Collaborative Team Environment, *Proc. 3rd Int'l Conf. Ubiquitous Computing*, (2001) 123-138.
5. Gross, R., Bett, M., Yu, H., Zhu, X.J., Pan, Y., Yang, J., Waibel, A.: Towards A Multimodal Meeting Record, *Proc. IEEE ICME 2000*, (2000) 1593-1596.
6. NIST Homepage. <http://www.nist.gov/smartspace/index.html>
7. Hindus, D., Schmandt, C.: Ubiquitous Audio: Capturing Spontaneous Collaboration, *Proc. ACM CSCW'92*, (1992) 210-216.
8. Mueller R., Ottmann T.: The Authoring on the Fly System for Automated Recording and Replay of (tele)Presentations, *ACM/Springer Multimedia Systems Journal*, Vol 3, no. 8. (2000) 158-176.
9. Chiu, P., Boreczky, J., Girgensohn, A., Kimer, D.: Liteminutes: an Internet-based System for Multimedia Meeting Minutes, *Proc. of the International World Wide Web Conference*, (2001) 140-149.
10. Truong, K.N., Abowd, G.D.: INCA: A Software Infrastructure to Facilitate the Construction and Evolution of Ubiquitous Capture & Access Applications, *Proc. Pervasive 2004: 2nd International Conference on Pervasive Computing*, (2004) 140-157.
11. Xie, W.K., Shi, Y.C., Xu, G.Y., Mao, Y.H.: Smart Platform - A Software Infrastructure for Smart Space (SISS), *Proc. 4th IEEE International Conference on Multimodal Interfaces*, (2002) 429-434.
12. Smart Platform Homepage. <http://media.cs.tsinghua.edu.cn/~pervasive/projects/platform/>
13. He, W., Xiang, P., Shi, Y.: A Semantic Framework for Meeting Data Retrieval, *Proc. of the 2nd ACM Workshop on Capture, Archival and Retrieval of Personal Experience*, (2005) 809-819.
14. He, W., Shi, Y., Xiao, X.: Auto-summarization of Multimedia Meeting Records Based on Accessing Log, *Proc. of the 2005 Pacific-Rim Conference on Multimedia 2005.* (2005) 53-59.
15. OWL Web Ontology Language <http://www.w3.org/TR/2004/REC-owl-features-20040210/#s2.2>