Towards Pervasive Instant Messaging and Presence Awareness

Xiaolei Zhang, Chun-Fai Law, Cho-Li Wang, Francis C.M. Lau
The University of Hong Kong

Received: January XX 2005; revised: November XX 2005

***Abstract*— This work envisions the benefits of applying the Instant Messaging (IM) paradigm in pervasive computing environments. With IM in such an environment, all smart entities, human or not, can interact using IM as the unified interface. To realize this vision, the design of a Smart Instant Messaging (SIM) system is proposed, which features context-aware presence management, user-centric resource configuration, and adaptive grouping. This system extends the Jabber-based IM framework and relies on an ontology-based supporting middleware to handle the chore of retrieving and interpreting contextual information. Three prototype versions of a client have been implemented and their performance in terms of memory usage and response time is evaluated.**

***Index Terms*— Pervasive computing, instant messaging, presence-awareness, ontology-based context middleware**

I. INTRODUCTION

Instant Messaging (IM) enjoys enormous popularity nowadays. According to a 2005 AOL's survey [1], 70% of Internet users use instant messaging and 38% of them swap instant messages no less than they send emails. With the proliferation of handheld devices, mobile instant messaging is quickly becoming mainstream, and demands the attention of both industry and research. IM now pervades almost every aspect of daily living—at home, at work, at school, and on the road.

Currently, the usage of IM is limited to supporting human-to-human communication. Conceptually, however, in pervasive computing environments, anything having some computation and communication capability could be made a "talking" buddy. It is no longer the privilege of human users to initiate a conversation; rather, all smart resources could interact with each other, or address a human user by their own initiation. This yields a grand arena for the development of IM. By extending the coverage to ubiquitous devices and resources, IM would become a unified interface for all communications. The benefits of a unified interface are as follows.

First of all, IM provides the simplest and a convenient form of interaction, i.e., near-synchronous message exchange. For human users, it connects people with more immediacy than email and without the expense of a phone call. The interaction approximates the natural face-to-face dialog of human, and multi-tasking is possible. The quick, short and informal conversation of IM favors abrupt encounters. These features are in line with the goal of pervasive computing which is to support users and their daily activities on an extended scale of time and space.

The attractiveness of IM is also manifested in the aspect of presence awareness. For IM, this is the feature that differentiates it from other communication means such as email or telephone. Presence at the basic level means the availability status or responsiveness of a user to engage in a conversation. People use this information to decide whether it is the proper time to initiate a communication. In pervasive environments, such awareness is essential, as the surrounding is expected to be dynamic, and any attempt of communication could easily return a failure. A pre-knowledge of the corresponding party's presence can improve the situation. The concept of presence also applies to describing the availability status of ubiquitous resources. In general, given instances of human (H) and resource (R), presence awareness can be enabled for all H-H, H-R, R-H and R-R pairs.

A common function of IM systems, contact list management, can serve as an interface to browse environmental resources. Currently, an IM user can keep a roster of contacts (or "buddies"), which is displayed on the user's device upon logon. This list provides an immediate view of the buddies and their status, and the user can contact them directly via the list. By extending this function to include environmental resources, the user can quickly locate and interact with whatever resources they need, using the same familiar IM interface.

The above features suggest that IM can indeed serve as a unified interface for pervasive communications. However, current IM systems have yet to upgrade themselves in order to realize this vision, as explained below.

First, the vocabulary and semantics of presence need to be enriched. Primarily, presence is the direct opposite of absence, indicating whether a person has logged on and is reachable. Typical presence includes such status as "onlne", "busy", "away", "be right back" and "on the phone", etc. But such presence is insufficient to convey a user's real situation. As a result, users often have to leave a message to supplement. Mobile users suffer more from the limited vocabulary. For instance, a device can be connected and the user is online, and yet the person can miss the message if the device is left in pocket. The situation is worst for resources in general, for which there is a lack of meaningful descriptions. Therefore, the need to extend the presence vocabulary is imperative.

Updating of presence should be automatic. Current IM

products require the users to consciously change their status, which could be troublesome and error-prone. In commercial IM systems like MSN Messenger and ICQ, a user's desktop activity is sometimes exploited to infer presence—for example, the user's status is set to "away" after a certain period of non-activity; such result however is neither accurate nor sufficient. For mobile IM users, manual update is an even more formidable task, as their location could virtually be everywhere and their activity anything. These users are more likely to encounter abrupt situations and experience more frequent changes. On the other hand, in contrast with desktop users, it is less probable for a mobile user to be continuously engaged with the device. Automatic presence update by the system thus appears to be necessary.

In most products, a user's presence appears the same to all buddies, which leads to the block-all-or-block-none situation. This is clearly in contrast to human interactions in which the acceptance or refusal of a conversation is not only dependent on the principal's status, but also on the relationship between the two parties as well as the priority or urgency of the message. Therefore, presence also implies the subjective willingness of the user at a specific moment towards a specific conversation. A more fine-grained mechanism for sharing presence has yet to be devised.

Most current systems display all the contacts of a user in the IM client. This practice, however, is ill-suited for the pervasive environment as there could be hundreds or even thousands of buddies. Displaying or searching in such a large collection is difficult. Although grouping mechanisms are available for a user to categorize their contacts, these mechanisms do not scale well. Furthermore, the grouping is manually set by users and would remain static thereafter. They may not actually reflect the reality where people come and go and their relationships evolve with time. Therefore, adaptive filtering and grouping mechanisms are both necessary for the extended IM usage.

Dealing with resources as buddies arouses a few issues. In open, heterogeneous environments, the interaction of resources could be somewhat serendipitous. Two processes are required to make it happen, i.e., to discover each other and to establish an acquaintance. Ideally, the discovery should only include those resources that the user is interested in, and the resources should be able to talk using free-form messages, in order to fulfill the user-centric motto of pervasive environments. However, such a feature can rarely be found in current technologies.

Targeting these requirements, this paper presents Smart Instant Messenger (SIM), an extended IM system for pervasive environments. The essential idea behind is to treat every smart entity as a buddy and use IM as the unified interface for all interactions. In the proposed system, the context is exploited to extend the presence vocabulary; and two types of grouping mechanism, namely vicinity-based grouping and activity-based grouping, are devised to support dynamic and adaptive organization of the contact list. The underlying infrastructure manages and monitors the environments' resources; distributed intelligent proxies select and configure the resources with context-sensitivity, and the user interfaces are dynamically generated based on a device's capability. A Jabber-based framework is modified to incorporate the above new features in the IM system. Context-awareness support is achieved by an ontology-based middleware whose duty is to gather and interpret context information on behalf of the applications as well as the system.

The rest of the paper is organized as follows. Section 2 elaborates on the issues and design principles for the extended IM usage. Section 3 overviews the architecture of the proposed SIM system and discusses how the new features are realized. The ontology-based context-aware supporting middleware (CASM) is also introduced. Section 4 reports the implementation details and the experimental results. Similar work is compared and discussed in Section 5 and the paper is concluded with reflection on experiences and an outlook for future work.

II. Issues and Design

This work attempts to expand the potential of Instant Messaging by extrapolating from this communication paradigm a similar but new means of communication for the pervasive computing environment. The previous section has discussed new requirements that challenge existing IM concepts and systems. This section will go over the issues and elaborate on the approaches suggested by this work.

A. Context-aware presence management

The extended IM in this work demonstrates the features of a real pervasive application which is used by anything, anytime, anywhere and in any aspect of life. Aside from a ubiquitous messaging service, the presence awareness function needs to work on an unprecedented scale. As identified in the previous section, three limitations exist in current presence services: limited vocabulary, manual update, and unitary distribution.

For a human user, presence conveys the ability and willingness to communicate. Though the concept of presence is popularized by IM systems, its root can be traced back to awareness research in Computer Supported Cooperative Work [2], where the cues to promote awareness include virtually everything: social, cognitive status, location, interaction and communication status [18]. A more general view has suggested that presence information should answer the questions of Who, Where, When, How and Why [4]. Recent works have also identified the insufficiency of basic information and responded with some rich presence proposals [5][6]. However, there is still a lack of consensus on what the vocabulary of presence should contain.

We find presence to be a mixed concept, which involves the observable, external situation of a user as well as the user's willingness at a specific moment towards a specific communication. The willingness is based on an internal re-interpretation of the current situation, a process subconsciously carried out by the human user. We figure also that the observable situation of a user in fact is a subset of the user's context which is highly relevant to the communication act.

The observation above prompted us to consider factoring a user's presence into two levels. At the general level, we

enrich the vocabulary of presence using context information, including the user's activity, location, device capability, etc. The other level encapsulates the user's preferences and willingness towards various communications, which is represented by a set of user-defined rules.

The design of a presence service is thus also divided into two parts. The generic presence service automatically gathers and releases context information as the user's enriched presence. At the other level, mechanisms are provided for the user to override the default presence. The user can manually select a level of willingness, or use adaptive rules to adjust the willingness against different people or situations. Such knowledge is also used to guide the system for adaptive sharing of presence information.

B. Activity-based grouping and vicinity-based grouping

In current IM systems, by a subscription request for a buddy, a user can add the buddy to the contact list. This way, the list grows or shrinks under the user's manual control. The mechanism is good enough for today's use, since such operations only happen occasionally and users tend to interact with the people they actually know. However, this is not the case when moving on to pervasive IM scenarios. We argue that, to take advantage of environmental resources and chance encounters, the contact list should not be pre-configured by the user. Rather, the system should automatically adapt the content to display to reflect the interest of the user at the current moment.

Generally, there are two kinds of interaction for a human user: intended and spontaneous. Intended interaction is typically based on a arrangement beforehand such as a shared task requiring the users' collaboration. Spontaneous interaction usually takes place where the human users or computational participants coincide temporarily at a location [7]. We can therefore infer two indicators to be used by grouping mechanisms, namely, shared activity and vicinity.

When people collaborate in a task, they need a means for quick questions and timely responses. They also would like related electronic resources to be ready for sharing. Consider the situation of a clinical discussion about a certain patient. If a doctor can see in his contact list the other doctors (remote or co-located), nurses, electronic patient records, projector, the real-time monitoring sensors of the patient, and so on as a group, the efficiency of communication and accuracy of diagnosis will both be improved.

Given the enormous growth of the number of mobile users and the likely scenario of "hundreds of computers per room", chances for spontaneous encounter of human and all sorts of resources will multiply. Many of us will benefit from the serendipitous availability of other people and computational resources as we move around during our everyday living. Such are the situations when virtual and physical interactions can integrate. Users can quickly switch to the real experience of talking with someone, or discovering and manipulating a device on hand. The physical vicinity can further be complemented by the proximity of interest and disposition, just like many online discussion groups where users are grouped according to interests and background. This enhanced way of grouping promotes socializing and knowledge communication among human users.

In open, dynamic environments with evolving usage scenarios, it is very difficult to predict a set of grouping mechanisms that will be be sufficient. In this work, we choose to focus on two mechanisms—activity-based grouping and vicinity-based grouping, and leave the space of extension for future work.

C. User-centric resource configuration

In our envisioned environment, "resource" refers to anything that has an identifiable functionality and is able to compute and communicate on its own. In pervasive environments, the number and types of such resources could be staggering, and the user would need a highly efficient means to access them. These resources can initiate the interaction too, for example, to report an exception, or to reveal the progress of their tasks.

An important theme of pervasive computing is user-centricity. The system design should be directed to support the users and their tasks as the top priority, and to reduce the amount of distraction. For a human user, the baseline requirements are to be able to find an appropriate resource and then to interact with it in a simple way. The system needs to streamline the process for the user, which will discover and select a suitable resource and establish the communication. In recent years, we have seen technologies that deal with advertisement and discovery of resources [8], including Jini and UPnP; however, their focus is more on software infrastructure support. In this work we aim at satisfying the user's needs at a higher level.

Discovering and selecting a resource need to take into consideration such factors as the user's preference, specific task requirements, the current situation and status of the resources, etc. The real-time decision must be made in a context-aware fashion. Such a mechanism is especially useful for mobile device users, as their devices could not put up with a long list of choices to be displayed, or a lengthy, energy-consuming process of manual selection. A practical consideration for the user would be to provide them with a customable graphical user interface or service menu which is dynamically downloaded to the client device. Jini has the concept of a proxy to connect to the service interface and the underlying protocols. However, it cannot easily adapt to specific user preferences or device displays.

This work proposes to describe a resource's interface in an extensible and machine-interpretable XML document. Instead of downloading the whole UI software, the user client acquires this document and generates a UI dynamically. This also reduces the user's and the device's burden, since unnecessary functionalities can be filtered out, and the UI can be organized according to the user's preferences. For example, when a user looks for a photo editing application, the icon of a suitable application appears on the contact list; and the only the "resize" and "blur" functions show up on the right-click menu. The result is so because it has been the user's habit to use only these two operations for photo-editing.

On the other hand, since the resource can also initiate a conversation, it needs the capability to discover a user

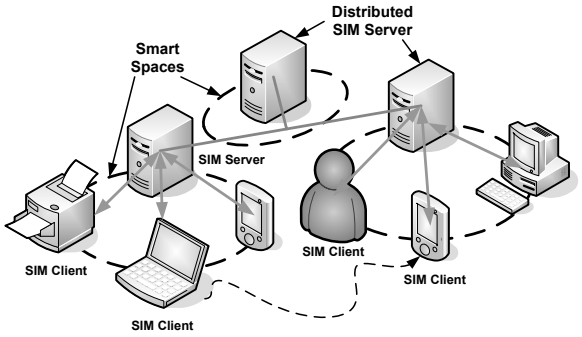


Fig. 1. Deployment of the SIM framework over smart spaces

and to "understand" a user's presence. In order to reduce unnecessary annoyance, the message delivery act may take different approaches in different situations. A resource buddy could delay the conversation until the user is available, send the notification to the user's mail box, or leave a voice message in the user's cell phone. Such flexibility is possible with the necessary support implemented in the system.

III. System Architecture

We have designed and implemented a prototype of the Smart Instant Messenger (SIM) system to address the requirements discussed in the previous sections. Our approach is realized using two layers. The IM Framework layer extends the existing Jabber Instant Messaging platform to cater for new behaviors and features. The Context-Aware Supporting Middleware (CASM) underpins the IM framework and handles the chore of context provisioning, including retrieving context information from various context providers, interpreting and reasoning about contexts, and monitoring context changes on behalf of the applications.

The SIM system is deployed within a pervasive environment in a peer-to-peer architecture (Figure 1). In each smart room, an SIM server operates to serve two purposes. First, it serves as a message switch and collaborates with other servers to relay IM messages. Second, it is the local repository of environmental resources. All local and incoming resources register to the server with their service descriptions. It is also where discovery and resource selection would take place.

The main components of the SIM system and their interactions are shown in Figure 2. Inside the SIM client, the *Instant Message* module provides the basic message exchange functions. The *roster* module is extended to include the presence, dynamic grouping and resource buddy features. The *context* module handles the context-triggered behaviors through subscription to interested events; it also monitors the user's conversational behavior, collects the IM context (i.e., context inferred from chatting and typing) and passes it on to CASM.

All the messages are XML formatted and go through the server. The SIM server parses the message, retrieves the target address and relays the message to the destination. It provides several message handlers to deal with different message types, including chat message, presence update, grouping mechanism and resource configuration, etc. The *resource repository* in the

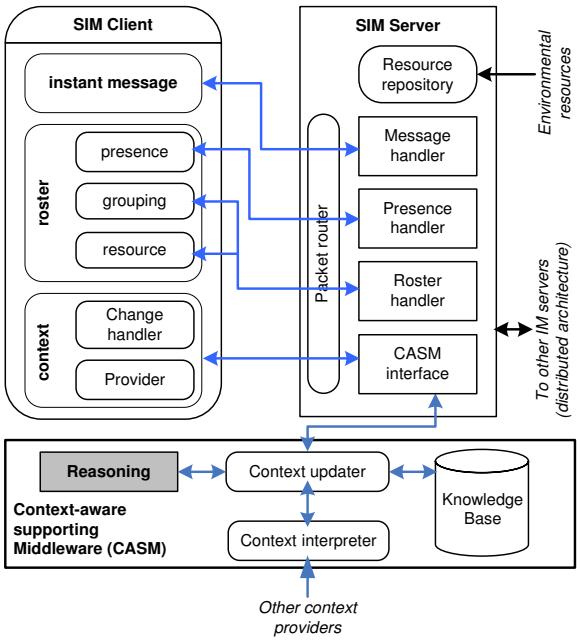


Fig. 2. Interactions between the SIM components

SIM server admits and keeps track of all the local resource descriptions. The *roster handler* takes charge of dynamic resource discovery and selection, and feeds the resulting list of resources back to the SIM client which later renders the roster based on the information. During execution, the SIM server would interact with CASM to query or subscribe to the interested context via *the CASM interface*.

A. Realizing the features

This section will elaborate on how our system design fulfills the requirements presented in section 2.

1) Context-aware presence management: As discussed in section 2, presence is categorized into context presence and user willingness. Resources have only context presence, or a variation of that when they "prefer" to put specific users on their priority list, for example, when they recognize a owner.

In order to support meaningful sharing of context presence, a common vocabulary is needed. Ontology is an obvious choice for this purpose, since it provides a formal, interoperable and extensible vocabulary for describing entities and concepts. An ontology-based vocabulary also provides a common language for defining user-specific rules.

Context presence includes situational information which the user is willing to disclose, such as activity, location, the people nearby, etc. Time can be added as an attribute to context presence, as it usually indicates whether a conversation will eventually be carried out, and if it will, when is the next suitable moment for retry.

Sharing of context presence is enabled by CASM, which actively collects user contexts on behalf of the applications. When a user's context is queried by a buddy for presence information, it would be encapsulated in an XML-formatted message, routed to the buddy's client, and parsed and displayed in the roster.

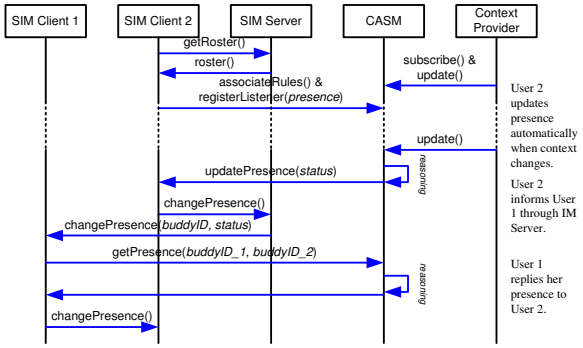


Fig. 3. Sequence diagram for the adaptive presence notification process

A user's willingness is either manually set by the user, or inferred by the system according to some predefined rules. The reasoning of the latter is based on the actual situation of the user and the user's relationship with the buddy. Therefore, the sharing of presence is adaptive and different buddies may see different statuses at the same instant. The process considers the user's preferences and exercises fine control over how a user's availability is distributed.

The following scenario illustrates the idea above. Suppose user A is attending a presentation with other group members, a situation which is detected by the CASM from environmental clues. The context presence of A is inferred to be "in the meeting", and optionally with a more detailed description about the meeting. The context presence could be masked, if A would like to signal a "Do Not Disturb" ("DND") to A's buddies. This can be done either manually or via some pre-specified rules:

Rule 1: *If I'm in a meeting, set my presence as "DND"*;
Rule 2: *If I'm in a meeting, set my presence to my group members as "Available"*;
Rule 3: *If I'm in a meeting, set my presence to my supervisor as "Available"* .

Rule 1 is the general rule for automatic update of A's presence. Rules 2 and 3 specify the exceptions for which the user is willing to raise the priority for those specific persons in question.

The negotiation protocol takes into consideration the priority of the message itself. Suppose user B has an important message for A; an indication of "urgency" could be delivered to A, and the negotiation process might result in changing A's status to "available", which happens exclusively for B.

Figure 3 shows the adaptive presence notification process. Upon initialization, the SIM client first registers the user preference rules to the CASM, describing the conditions under which the presence should be changed. It also prescribes the different statuses that should be displayed for different buddies. When a user's status changes, for each buddy that has a subscription, the presence to be displayed needs to be reevaluated. The process is carried out by CASM, based on inference with the pre-defined rules. In this way, a list of updated presence is generated, and then dispatched by the SIM server to the buddies accordingly.

Although presence is popularized by and intimately related to IM systems, the technology of presence is indeed orthogonal to that of Instant Messaging. Therefore, the mechanism for presence awareness can also be integrated with other applications such as email and calendar. In pervasive computing environments, it is highly desirable that presence support be built as a generic, stand-alone system service, to be shared by different applications and entities.

2) Activity-based grouping and vicinity-based grouping: The SIM system provides an extensible set of grouping mechanisms. In the current stage, we focus our attention on vicinity-based grouping and activity-based grouping. Activity-based grouping captures the user's collaborative intention to complete a task together with others. According to the structure of an activity model [9], entities that constitute an activity also include tools and objects that to be operated on. Therefore, the definition of a group goes beyond those by social software, and captures the temporary associative relationship of both human and resources.

This work exploits the activity model to pre-define the structure and participants of a collaborative activity. Our definition could specify a particular buddy, or the type of a resource. Requirements can also be associated with components in the model. When the condition is satisfied for an activity to start, the system is notified and a process of discovery is initiated. Based on the specified requirements, screening of resources and buddies is done to form a workable group. The result is then returned to the roster module in the SIM client, which in turn updates the display.

Vicinity-based grouping is based on the discovery of users and resources within the current locale. When a location change event of a user is detected, a nearby SIM server would register this client and retrieve from CASM the user's context and profile, as well as the profiles of the pending tasks. Such information might be dispersed in various personal applications such as calendars and address books, which is compiled periodically by CASM to synthesize knowledge about a user. Based on this knowledge, the server browses its repository of all registered resources and human users to find those that would interest the user. The resulting list is organized and wrapped in an XML-format message and returned to the user's client device, which will then show up in the roster.

3) User-centric resource configuration: A resource in the SIM system needs to have a description file specified in XML format. The file contains the profile of the resource, its functionality, run-time behavior and possible context conditions. Such a description file is registered to a proximate SIM server upon registration, and is later used for matchmaking in the discovery process.

In order to facilitate interaction with a user, a service menu is provided with each resource in the user's contact list. When the user invokes the service, the requested operations are wrapped in messages and directed to the resource. The service menu, which could be in terms of a complex graphical user interface, can be dynamically generated. The SIM system uses the User Interface Markup Language (UIML) [10] for abstractly defining the user interface of a resource. UIML is

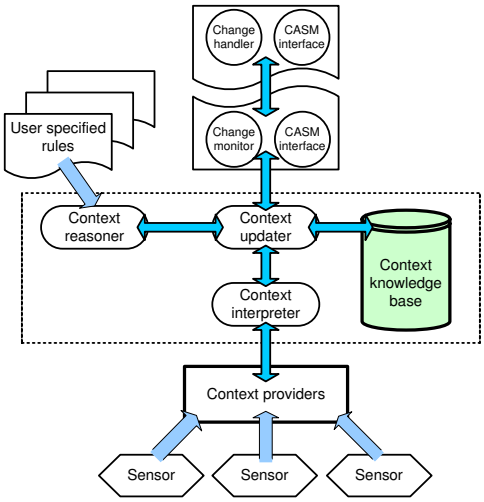


Fig. 4. Detailed design of the context-aware supporting middleware

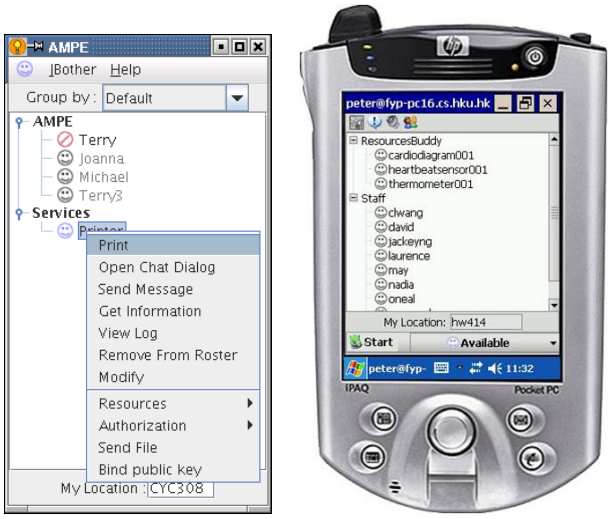


Fig. 5. Client-side GUIs for PC and PDA

an XML markup language with tags to describe the structure of UI elements, their properties and behaviors when being manipulated. The advantage of using UIML is that it separates interface code from non-interface, application logic, and it allows multiple user interface descriptions to be associated with one application logic. For example, for devices of different resolutions, the GUI of the same application can be different.

In the SIM system, the UIML description of a resource is also registered with the SIM server. When an interaction begins, the corresponding description is shipped to the requesting client, based on which the interface is dynamically rendered. This way, the network load is reduced, and the adaptation and user customization can be done on-the-fly.

B. Context-aware supporting middleware

The SIM system explicitly separates the context processing routines from the application logic. A generic context-aware supporting middleware (CASM) handles the chore of processing, interpreting and reasoning for context information retrieved from various context providers. The separation not only alleviates the task of programming the context-aware features of applications, but encourages reuse of context related processes and results.

CASM features an ontology-based model for the formal representation of contexts, which makes easy knowledge sharing in the open, heterogeneous pervasive environment, and enables various mechanisms to do logic-based context reasoning. Contexts are classified into five categories: Device, Person, Location, Time, and Activity. There also exist properties on relationship among these main classes. For example, an instance of class “Person” can have a relationship called “hasLocation” which links to an instance in the “Location” class. All classes and relationships can be added or removed as needed.

Figure 4 shows the detailed design of CASM. The Context Interpreter translates the context compiled from heterogeneous sources to form an OWL data instance which stores all the dynamic context information (e.g., location, time, current activity) in files. The Context Updater directly manipulates the context model. When the context model is first created, the schema file will be parsed and the data type of the domain and range for each property are specified. The Context Updater validates the data type each time an add/remove request is received. Upon a context query, it inquires the context model and forms the answer in a format that can be used on the client side easily. The Context Reasoner provides two kinds of reasoning over the context ontology—transitive reasoning and rule-based reasoning. The former is used to store and traverse class and property lattices. The latter hosts a general purpose rule engine, which performs rule-based inference over the ontology. Depending on the schema and domain of the ontology bound to the context model, rules can be written to derive some implicit information or map information to a standard format for the benefits of the applications.

CASM also provides a set of standard methods for the application developers to update, query and register context event listeners to the middleware. An application registers interested context events to CASM, and relies on the latter to monitor the environment on its behalf. A notification will be issued when any of these events takes place, prompting the Change handler module in the application to invoke the corresponding event handling methods.

IV. Implementation and Experimental Results

The SIM server uses and extends the open-source Jive Messenger, which is now called Wildfire [11]. We extend Jabber’s Extensible Messaging and Presence Protocol (XMPP) [12], which is currently an Internet Engineering Task Force draft, for the mechanism that reports the presence of buddies and handles the interactions among human, software and devices through XML messages. Both wireless LAN (802.11) and Bluetooth connections are enabled in the environment. We have implemented three versions of SIM clients: PC, PDA and cell phone. The PC and PDA clients are from modifying the open-source Jabber client program “JBother” [13], and the cell phone version is built from scratch. All three

Fig. 6. Cell phone version of the GUI running in a Nokia emulator.

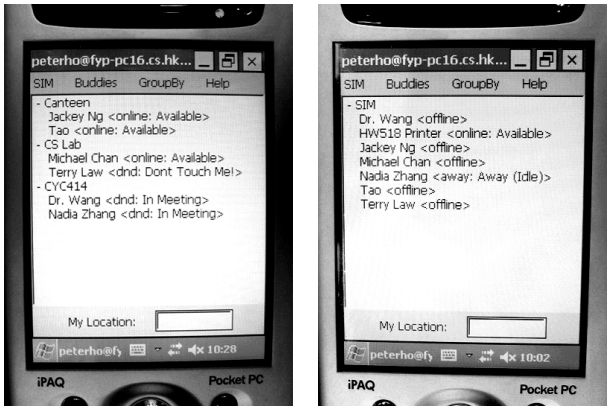


Fig. 7. Client-side GUI showing vicinity-based grouping and activity-based grouping on an HP iPAQ H5500.

clients possess the basic functionality of IM dialog as well as some advanced features such as context-aware presence management, user-centric resource configuration and dynamic grouping. The PC version has been used and tested in a university campus environment, while the PDA version has been adopted in a hospital to assist clinical staff in their daily tasks. Figure 5 shows the client-side GUIs for the PC and PDA clients respectively, where the PC client has six human members and a printer in the roster; a right-click menu is enabled for the printer, which can be used to invoke a series of services. The roster in the PDA shows the resources and clinical staff that are in a room.

Cell phone has gradually become a special platform where technologies converge. The subscriber identity module in the cell phone associates the device with the identity of of the owner, such that the status and the location of a cell phone provide a useful indication of the user's situation. The potential of SIM on cell phone is that it would improve users' experiences via presence sharing among a variety of communication modes and applicated hosted in the phone, including phone calls, email, and Short Message Service (SMS), etc. and would unify them by means of a common interface. Figure 6 shows a cell phone version of SIM running in a Nokia emulator. The roster combines image icons and text descriptions for a visualization of a buddy's status. The client program is currently implemented in J2ME (MIDP2.0).

Figure 7 demonstrates the two grouping mechanisms in the SIM system. The first SIM client running on a PDA groups people and resources according to their locations which could be canteen, lab or office. The second one shows an automatically formed group relating to a project, containing six human members and a printer resource.

Sensors are used to obtain a snapshot of the environment: light, noise, motion and temperature (Figure 8). At the current stage of implementation, only in-door locations with the granularity of a room are exploited. The various locations are each assigned a symbolic identifier. The RFID technology is used to identify people entering or leaving a room or place. Users' mobile phones are attached with RFID tags, and an RFID reader at the door of each room would trigger an event

Fig. 8. Sensors used in the trials: RFID card reader, bluetooth access and light sensors.

upon someone entering or leaving the room.

We have constructed several ontologies for pervasive computing environments. Figure 9 shows one ontology used for modeling the campus life. The Web Ontology Language (OWL) is selected as the ontology language because of its expressivity and standardization. Reasoning and inference over the context models are based on the Jena [14] framework. A set of rules have been developed to infer high-level contexts from low-level facts. We notice that one of the most time-consuming parts of the system (wireless delays excluded) is related to the operation of the CASM middleware. As more context instances are added into the context knowledge base, the overhead of the middleware grows accordingly. This is an area whether future optimization efforts can be targeted.

To test the performance, we evaluate the responsiveness and memory consumption against the increase of the number of instances. A PC (Intel Pentium4 2.26GHz, 512MB memory, Linux FC 3.0) running Jena version 2.2 hosts the context-aware supporting middleware. A typical sequence of operations is compiled as a sample test, including two adds (adding instance data to the ontology), one remove (removal of instance data), one class query and one instance query. The numbers of instances we try are 300, 700, 1000, and 1800. At each stage, the sampling sequence is performed and the total processing time and memory usage are recorded. The result shows an approximately linear growth of memory usage varying from 17MB to 22MB and an average processing time of 3.4s with variations within 0.2s. The performance of the system is acceptable for non-crisis scenarios and adding more instances would not cause much degradation.

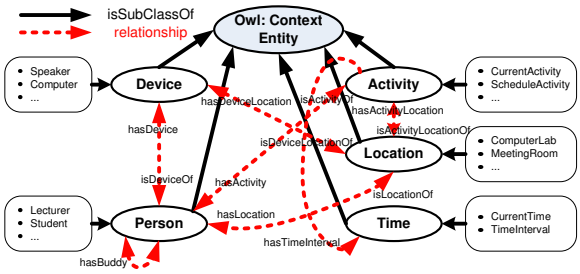


Fig. 9. Diagrammatic view of the campus ontology model

V. Related Work

The concept of combining awareness with communication is not new. In the computer supported cooperative work community, awareness of a remote partner's activity has been exploited to facilitate coordination. For example, media space [15] and porthole [16] use live video shows or snapshots to display the partner's office activities. The benefit of presence has also been recognized by researchers to improve telephony. For instance, Live addressbook [17] has taken advantage of presence cues to provide status information in the telephone book. Similar ideas can be found in [18] and [19].

Presence usage within Instant Messaging has also attracted some research interest. Past research work proceeded mainly along two dimensions. The first dimension aims to enrich presence information, such as PLIM [20], ConChat [21] and F@ [22]. PLIM combines functionalities related to location, presence and instant messaging in a context-aware mobile application framework. Conchat automatically offers cues to users about the context of their contacts, enabling them to exchange contextual information outside the main channel of conversation. Both projects employed mobile devices and described the extension of presence to work with them. However, sharing of presence is not adaptive and they do not take resources into consideration. In the F@ framework, awareness information is divided into four types: awareness of multiple concurrent conversations, conversational awareness, presence awareness of a group conversation, and visibility of moment-to-moment listeners and viewers. This work, however, focuses only on enhancing the awareness functions in the current IM paradigm, and has not touched on the potential of IM to the larger extent.

The second dimension is to devise system support for automatic presence inference. Lilsys [23] exploits the ambient sound, phone usage and computer activities to infer the unavailability of a user. The Awarenex [24] system yields cues about a user's presence from location and calendar events, and the idea of Rhythm is proposed there to model the pattern of a person. These two projects resemble our work in the aspect of automatic interpretation of context presence, but they did not attempt to define a vocabulary of presence, and the sharing of presence remains unitary. Perttunen et al. propose to utilize both the initiator's and the recipient's contexts in inferring presence [25] [26], which is similar to our SIM's support for user willingness. However, their work has not done anything about the need to enlarge the presence vocabulary.

Kranz et al. [27] also promote the idea of extending the presence in IM to a ubiquitous presence system. They offer a tangible user interface for communicating one's physical state to a virtual communication system. Their work echoes our concept of extending presence but the two approaches are different.

Activity-based grouping is mainly addressed in groupware or collaborative software, i.e., those computer-based systems that support groups of people engaging in a common task (or goal) and that provide an interface to a shared environment. The focus in this research category caters to human users and their collaborative operations, such as co-editing a paper. Our work emphasizes more on providing an immediate and familiar view of participants and available tools (resources), where the "outeraction" [28] of IM is to be found. The idea of vicinity-based grouping is similar to location-based services [29], which is particularly true for the recommender systems in tourism [30], where location-relevant information or services are pushed to the user. However, our aim is different in that we choose a popular application (IM) and apply its familiar interface for browsing and interaction to nearly everything encountered.

Using mobile device as the interface to access environmental resources is also promoted in [31] and [32]. The former extends IM and its presence concept to include resources, which is similar to our SIM, and the latter proposes to use Short Message Service for communicating with resources. Their work has a lot in common with the SIM system in dealing with resources. However, our work does not only focus on resources; instead, the vision of SIM is to extend the IM style of communication and awareness to a much wider scale covering ubiquitous devices and resources. Our design for presence awareness targets at a generic service that could benefit a wide range of applications. To our best knowledge, there is no similar work on extrapolating the IM paradigm to an extent like ours.

VI. Conclusion

In this research, we explore the vision of extrapolating the instant messaging paradigm, fitting it to the pervasive computing environment. We discuss the new requirements arising from this vision, and revisit the somewhat obscure concept of presence. We present solutions that meet these requirements, and have implemented them in the Smart Instant Messenger (SIM) system. This system transcends current IM products with new features including context-aware presence management, user-centric resource configuration, and adaptive grouping support. An ontology-based context-aware middleware underpins the IM framework, which takes care of retrieving, interpreting and reasoning over contextual data. Experimental results suggest that our design is feasible. To summarize, our design has followed the following principles:

1) *User-orientation*. An important theme of pervasive computing is user orientation, which is to reduce distraction on and maximize the personality of the user during computation. The SIM system design has investigated ways of fulfilling the user's requirements and improving the user's experience. Presence update, for example, is

automatic, grouping is devised to fit real-life scenarios, and resources are selected and configured according to user preferences. We believe, in the future, putting the human user at the center of any system design will always be the optimal choice.

2) *Separation of context provision from context consumption*. The chore of retrieving and managing contexts need not be a part of the application; rather, a separate middleware layer or the underlying system infrastructure should be made responsible, which is the approach adopted by SIM. On one hand, it makes life easier for the programmers and reduce the load on the small devices; on the other hand, the separated functions can form a generic middleware which could potentially benefit many different applications.
3) *Design for extensibility*. Extensibility is crucial in pervasive environments as users, applications, devices and sensors come and go dynamically. Also, the users' requirements might change rapidly over time. SIM chooses the Jabber protocol for its extensibility features, adopts a distributed architecture, and exploits an ontology-based context modeling solution to facilitate the re-use and integration of knowledge.
4) *Prototype for real life usage*. Pervasive computing is by and large in its germinal stage. We believe live applications will stimulate and inspire more useful research. Therefore this version of SIM has been designed to work for a campus and clinical environments, where there is a rich collection and variety of resources, users and use cases. We believe these test environments could bring about insights that can benefit future applications and research.

The dual tradeoff between privacy and awareness, and between awareness and disturbance [33] still exists. As SIM treats nearly everything as an IM buddy, it aggravates the problem, which we will address in our future research.

Ongoing work includes the refinement of the concept of presence, the design of a pervasive presence ontology and the exploration of ontology-mapping support. We also need to devise ways to improve the performance of CASM. Another plan is to deploy SIM in a range of real-life environments, in order to gather more insights from diverse usage scenarios.

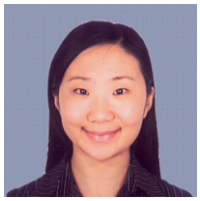
ACKNOWLEDGMENT

This research is supported by CERG grant HKU 7146/04E from the Hong Kong Government.

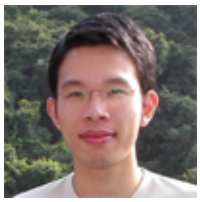
REFERENCES

[1] America Online. Third annual instant messaging survey. Online resource. http://www.aim.com/survey, 2005.
[2] P. Dourish, and V. Bellotti. Awareness and coordination in shared workspaces. Proceedings of the ACM Conference on Computer-Supported Cooperative Work CSCW92 (Toronto, Ontario), 107-114. New York: ACM.
[3] A. Oulasvirta, M. Raento, and S. Tiitta. ContextContacts: re-designing SmartPhone's contact book to support mobile awareness and collaboration. *In proceedings of the 7th international conference on Human computer interaction with mobile devices services*, 167–174, Salzburg, Austria, 2005.
[4] R. Chakraborty. Presence: a disruptive Technology. *Presentation at Jabber Conference*, Denver, 2001.
[5] J. Jachner, S. Petrack, et al. Rich presence: a new user communications experience. *Alcatel telecommunications review*, technical whitepaper. Q1: 73–77, 2005.
[6] H. Schulzrinne, V. Gurbani, et al. RPID: rich presence extensions to the presence information data format. IETF RFC 4480, 2006. Online resource: http://www.ietf.org/rfc/rfc4480.txt.
[7] M. Esborjörnsson, and M. Östergren. Issues of Spontaneous Collaboration and Mobility. Workshopt on Supporting Spontaneous Interaction in Ubiquitous Computing Settings, the 4th International conference on Ubiquitous computing. Göteborg, Sweden, 2002.
[8] G. Richard. Service Advertisement and Discovery. *IEEE Internet Computing*, **4**(5): 18–26, 2000.
[9] Z. Gregory, et al. Activity theory: history, research and application. *Theoretical issues in ergonomics science*, **1**(2): 168–206, 2000.
[10] User interface markup language. Online resource. http://www.uiml.org/.
[11] Jive software. Online resource. http://www.jivesoftware.org/Wildfire/.
[12] Jabber/XMPP protocols. Online resource. http://www.jabber.org/protocol.
[13] JBother Homepage. Online resource. http://www.jbother.org/.
[14] Jena: a semantic Web framework for Java. Online resource. http://jena.sourceforge.net/.
[15] S.A. Bly, S.R. Harrison, and S. Irwin. Media spaces: bringing people together in a video, audio, and computing environment. *Communications of the ACM*, **36**(1):28–46, 1993.
[16] P. Dourish, S. Bly. Portholes: supporting awareness in a distributed work group. *In proceedings of the SIGCHI conference on Human factors in computing systems*, 541–547, Monterey, California, United States, 1992.
[17] A. E. Milewski and T. M. Smith. Providing presence cues to telephone users. *In proceedings of the 2000 ACM conference on Computer supported cooperative work*, 89–96, Philadelphia, Pennsylvania, United States, 2000.
[18] A. Oulasvirta, M. Raento, and S. Tiitta. ContextContacts: re-designing SmartPhone's contact book to support mobile awareness and collaboration. *In proceedings of the 7th international conference on Human computer interaction with mobile devices services*, 167–174, Salzburg, Austria, 2005.
[19] A. Schmidt, et al. Context-aware telephony over WAP, *Personal and Ubiquitous Computing*, **4**(4): 225–229, 2000.
[20] A. J. H. Peddemors, M. M. Lankhorst, and J. de Heer. Presence, location, and instant messaging in a context-aware application framework. *In 4th International Conference on Mobile Data Management (MDM), volume 2574 of Lecture Notes in Computer Science*, 325–330, Springer, 2003.
[21] A. Ranganathan, R. H. Campbell, A. Ravi, and A. Mahajan. Conchat: A context-aware chat program. *IEEE Pervasive Computing*, **1**(3):51–57, 2002.
[22] M. H. Tran, Y. Yang, and G. K. Raikundalia. The F@ Framework for Designing Awareness Mechanisms in Instant Messaging. *the Australasian Journal of Information Systems AJIS*, 2006 (to appear).
[23] J. B. Begole, N. E. Matsakis, et al. Lilsys: sensing unavailability. *In proceedings of the 2004 ACM conference on Computer supported cooperative work*, November 06-10, 2004, Chicago, Illinous, USA.
[24] J. C. Tang, N. Yankelovich, J. B. Begole, et al. ConNexus to Awarenex: Extending Awareness to Mobile Users. *In proceedings Conference on Human Factors in Computing Systems CHI'01*, 221–228, Seattle, WA, New York, NY, 2001. ACM press.
[25] M. Perttunen, J. Riekki J. Inferring Presence in a Context-Aware Instant Messaging System. *the 2004 International Conference on Intelligence in Communication Systems*, November 23-26, 2004.
[26] M. Perttunen, J. Riekki, et al. Experiments on mobile context-aware instant messaging. *In proceedings of the 2005 International Symposium on Collaborative Technologies and Systems*, 305–312, May 2005.

[27] M. Kranz, P. Holleis, and A. Schmidt. Ubiquitous presence systems. *Proceedings of the 2006 ACM Symposium on Applied Computing (SAC)*, 1902–1909, Dijon, France, April 23-27, 2006.
[28] B. A. Nardi, S. Whittaker, and E. Bradner. Interaction and outeraction: instant messaging in action. *In proceedings of the 2000 ACM conference on Computer supported cooperative work*, pp. 79–88, Philadelphia, PA, US, 2000. ACM Press.
[29] S. Barnes. Location-based services: the state-of-the-art. *e-service journal*, **2**(3), 2003.
[30] S. Stabb, H. Werther, et al. Intelligent systems for tourism. *IEEE Intelligent Systems*, bf 17(6): 53-66, 2002.
[31] J. Favela, et al. Extending instant messaging to support spontaneous interactions in ad-hoc networks. *In proceedings of ACM 2002 conference on computer supported cooperative work*, new orleans, Louisiana, 2002.
[32] F. Siegemund. Spontaneous interaction in ubiquitous computing settings using mobile phones and short text messages. Workshop on supporting spontaneous interaction in ubiquitous computing settings, the 2rd international conference of ubiquitous computing, 2002.
[33] S.E. Hudson and I. Smith. Techniques for addressing fundamental privacy and disruption tradeoffs in awareness support systems. *In proceedings of the 1996 ACM conference on Computer supported cooperative work* pp. 248-257, November 16-20, 1996, Boston, Massachusetts, US.



Xiaolei Zhang is a Ph.D candidate in the Department of Computer Science at the University of Hong Kong. Her research interest includes pervasive computing, context aware systems and applications. She received her B.S. and M.S from Nanjing University, China in 2000 and 2003 respectively, all in computer science. Contact her at the Department of Computer Science, the University of Hong Kong, xlzhang@cs.hku.hk.



Chun-Fai Law received the B.Eng degree in Computer Engineering from The University of Hong Kong in 2005. He is currently an M.Phil candidate in the Department of Electrical and Electronic Engineering at the University of Hong Kong. His research interests include pervasive computing, distributed computing, wireless networking, and sensor network security. He is a student member of IEEE. Contact him at the Department of Electrical and Electronic Engineering, the University of Hong Kong, cflaw@eee.hku.hk.

Cho-Li Wang received his Ph.D. degrees in Computer Engineering from University of Southern California in 1995. Dr. Wang's current research involves context-aware software systems for Pervasive Computing, Grid middleware with migration support, and distributed Java Virtual Machine on clusters. He is now on the editorial board of the IEEE Transaction on Computers, the International Journal of Pervasive Computing and Communications (JPCC), and Multiagent and Grid Systems (MAGS). He also serves as a regional coordinator (Hong Kong) of IEEE Technical Committee on Scalable Computing (TCSC). Contact him at the Department of Computer Science, the University of Hong Kong, Hong Kong, clwang@cs.hku.hk.

Francis C.M. Lau received his PhD in computer science from University of Waterloo. He is a Professor of the Department of Computer Science at the University of Hong Kong. His research interests are in parallel and distributed computing, object-oriented programming, operating systems, Web and Internet computing, computer graphics, and AI. He is a senior member of the IEEE. Contact him at the Department of Computer Science, the University of Hong Kong, Hong Kong, fcmlau@cs.hku.hk.