LATTE: Location And Time Triggered Email

Andronikos Nedos, Alex O'Connor, Graham Abell, Siobhán Clarke, Vinny Cahill

Distributed Systems Group,
Computer Science Department,
Trinity College Dublin,
Ireland
{nedosa,oconnoat,sclarke,vjcahill}@cs.tcd.ie, gabell@tcd.ie

Tel.:+353-1-6081539
Fax.:+353-1-6772204

Abstract. *Computer users, especially mobile ones, have developed a significant dependency on electronic communication. Enhancing the capabilities of such communication by adding an awareness of context such as location and time increases the quality and intuitiveness of the services available to the mobile user. This paper describes the design and implementation of a context-aware messaging system called LATTE (Location And Time Triggered Email). LATTE is based on the email paradigm, and extends it to include dynamic consideration of location and time to determine the appropriate recipients for a message. It uses SMTP (Simple Mail Transfer Protocol) for message delivery and an event middleware as the underlying communication infrastructure.*

Keywords: Location-based Systems, Pervasive Computing, Context Awareness, Applications.

1 Introduction

Context-aware applications gather and analyse context so that they may increase the relevance of the service provided to the user. A natural fit for this category of applications is the design of context-aware messaging services. For the current email user, especially the mobile one, a significant proportion of email received is not relevant for a variety of reasons. Conversely, mobile users often do not receive messages that may be of interest to them because these messages relate to the location the users are currently visiting. Adding a level of context awareness to messaging services is likely to ameliorate these problems, because the relevancy of a user's email is significantly enhanced due to filtering based on context. Requirements for such a service include offering relevant information to recipients under specific spatio-temporal constraints. It should be possible to augment messages with a variety of contexts (for example, any combination of location, time and identity) and deliver them when their contextual conditions are satisfied.

It is not surprising that a large number of messaging services have been built around this concept. Most of these messaging services are variations either of the "tourist" model where stored information is displayed depending on the location [1; 2] or the "digital graffiti" model where physical space is annotated with digital information [3; 4; 5]. Here, users in physical proximity can view and use this information.

Some sociological research has also been conducted on how to express information on context-aware messaging systems and on where to extract useful metaphors [6]. In this paper, we present a new approach to providing a context-aware messaging service that draws its metaphor from the interaction model of email and was developed by extending SMTP and by using a publish-subscribe model as the underlying communication mechanism. A benefit of this approach is that the user interacts with the system in a way that is familiar using a model that is mature and stable.

Section 2 provides a motivation for context-aware messaging, and introduces the LATTE context model. In Section 3, we describe the LATTE messaging model, outlining the main differences with standard email. Section 4 illustrates the overall architecture and implementation, with related work described in Section 5. Section 6 concludes.

2 Motivation

In this section we present the rationale behind the development of context-aware messaging, discussing shortcomings of the current email approach. We describe some categories of usage scenarios we considered in order to identify suitable context, and our resulting model for location, time and identity.

2.1 Current Email Services

With current email applications, each user has an identifier by which he is known for the purposes of sending and receiving messages. "Mobility" in the email world means that users that travel outside their offices/homes may read their email in whatever global location they find themselves, so long as they have access to the Internet. The actual email they receive will be the same regardless of where they are, or what time they read it. As most of us have experienced, the implications of this can be quite tiresome. For example, when you're away from home, have you ever had to wade through email notifications of meetings that are scheduled for when you're not there (but you might have attended if you had been), or emails of the type "let's all go out to lunch today", or "the system will be down for five minutes this afternoon"? The list of emails that we're *not* interested in because we're not at home sometimes appears endless. In contrast, if we'd only managed to get ourselves onto the distribution list that told people about the tickets available for the U2 concert in the city we're visiting (while we're still there), then that would have been nice!

On the other hand, the email model has been enormously successful, and we are entirely dependent on it. From a technical perspective, open Internet standards such as SMTP (Simple Mail Transfer Protocol) [8] work well, and are very widely deployed. However, the concept of the mobile user is becoming more and more the norm. Therefore, the traditional model of mobility that means users can read their email anywhere there's an Internet connection, is no longer sufficient. The mobile user needs an email service that takes into account various elements of his context other than his identity to determine the emails to be delivered to him. We believe that this should be provided *on top of* the email services currently available – in other words,

as extensions to SMTP. We therefore have designed and implemented a communication infrastructure that:

- delivers messages to intended recipients whose identity can remain undefined but their address is determined by a set of contextual attributes (e.g., location, time);
- provides context processing as extensions to the existing infrastructure;
- uses a flexible underlying infrastructure suitable for mobile computing that makes it easy to communicate the user context to applications or different back-ends for further processing.

2.2 Context-Aware Email Service

As an initial step, we identified a number of use case scenarios that a context-aware messaging service should handle. These can generally be categorised as:

1. **Alerts:** In this category, messages are attached to a location to indicate some warning. For example, well-wishing individuals or companies who undertake traffic monitoring as a commercial service may provide a traffic alert. Another example is a *periodic* form of alert message that is delivered to users in particular locations – for example, "this museum closes in fifteen minutes" is a message that should be delivered to people in the museum at the appropriate time every day the museum is open.
2. **Digital signposts:** This category provides similar functionality as the "digital graffiti" model [3; 4; 5] where a message is sent to a specific location with any user in the proximity receiving it. There is likely to be no information of any secrecy in this kind of message.
3. **Intended Delivery:** In this category, recipients are identified by some combination of identity, location and some timing constraints.

In general, context aware messaging can help reduce the amount of irrelevant information delivered to email users by using different kinds of context as a filter. In addition, it can help ensure that email users receive information relevant to them because of a change to their context. From scenarios within the categories listed above, we identified *two* context attributes, in addition to identity, that should be used as such filters: *location* and *time*. The notion of identity as a contextual filter for email shall remain of course, but with some interesting differences as illustrated in section 2.2.3. The goal of the context-aware email system is to deliver contextually valid messages to the right entities at the appropriate time. Where any combination of context attributes have been specified, validity is a function of testing the actual context of recipients against the one defined by the sender of the message. It is not necessary to specify values for all of the context attributes – as long as there is a recipient specified (either a named email user/group or a location), the other contextual attributes take sensible default values that try to emulate the familiar email behaviour as much as possible. We discuss each of the context attributes in the following sections.

2.2.1. Location

Location is defined as the spatial region a recipient must occupy in order to receive a message. Location is modelled as a series of discrete regions with defined, non-overlapping boundaries. Each region may be subdivided into smaller ones, which are arranged in a hierarchical manner. This approach was adopted from the Intentional Naming System (INS) [9] for its expressiveness and simplicity. Each node in the hierarchical tree has a logical name which maps into physical coordinates. There are many approaches to determining geographical coordinates, both indoor and outdoor – for example GPS, Infrared or RADAR [10]. Our context-aware email service is designed in an extensible manner to support input relating to coordinates from any source. In the current version, we have implemented location input for outdoors and indoors environments by using GPS and Infrared receivers respectively.

The location tree approach provides a scheme, in which different administrative domains (e.g. different universities in the same city) can control and publish their local mappings. As will be explained in section 4.4.2, LATTE's design and implementation follows this model, by allowing users to connect and use multiple LATTE servers. We enable in this way, a decentralized location management service in which LATTE servers correspond to certain geographic areas and each server publishes a unique map of logical to physical coordinates.

From the perspective of the email sender, location may be either explicitly specified, or left "anonymous". Where the location is specified, emails (without any further filtering based on identity/time) are valid to all users in that location. Messages can therefore be left at a location without requiring the sender to physically be there to tag it with a message. Where the location is anonymous, location is not considered in the filtering process for the email message.

As derived from the Intentional Naming System model, a location is named in an abstract, hierarchical plaintext format with the following syntax:

```
[location=value [division=value [subdivision=value]]..]
```

This gives a partial tree resembling Fig.1.

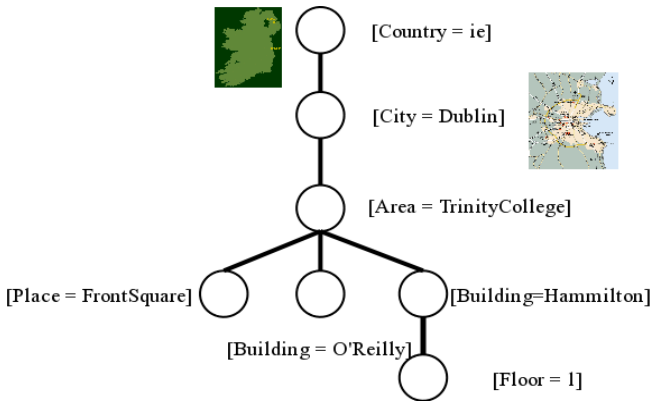

Fig. 1. Location representation in LATTE

For example, the front square in Trinity College might be described by

```
[Country=ie [City=Dublin [Area=TrinityCollege
[Place=FrontSquare]]]]
```

2.2.2. Time

Time is another important property to consider when evaluating whether to deliver particular messages. In LATTE, time can be *bounded* or *unbounded* and *periodic or non-periodic*. If a message is time bounded, then delivery of that message is valid for certain duration. For example, a sender might want to send a meeting notice reminder to relevant participants any time from one hour before to five minutes before the meeting starts. If a message is not time bounded, then time is not a factor that is considered when filtering email for delivery. Time-bounded messages may also be tagged as periodic or non-periodic. Periodic messages are delivered to appropriate recipients at the stated period – for example, every day at 16:45, send a "museum closes in 15 minutes" message. Time-bounded messages that are non-periodic are valid only within the explicitly specified time bounds.

2.2.3. Identity

The main difference between the notion of identity from classical email systems and identity in the context-aware extensions relates to the issue of when it is decided that a message should be delivered to a particular recipient. In classical email, the identity of the recipient is known at composition. This can be considered *early* binding. Adding contextual properties to a message, however, means that the message may not have its destination identity explicitly specified before transmission. Instead, there is a set of contextual attributes that can be matched against potentially multiple recipients at delivery time. This can therefore be termed *late* binding. Late binding improves flexibility and makes it possible to realise the categories of scenarios based on alerts, digital signposts and intended delivery as described in section 2.2.

In addition, we further classify the contextual values of identity as individual, group and anonymous. An *individual* identity is defined when the recipient's identity is known during composition and is therefore an early-binding mechanism similar to email. A g*roup* is defined by a group name and is a dynamic contextual attribute that LATTE clients can choose to transmit as part of their context. Lastly, when the identity is *anonymous,* any potential recipients who conform to the remaining contextual attributes of location and time will receive the message.

3 LATTE Messaging Model

In this section, we describe the LATTE messaging model, and outline the main differences with standard email. In essence, the messaging model has two main requirements: 1) to maintain messages and user profiles with their associated contextual information and 2) to reason about message context and match against potential recipient context. To achieve this, the basic email model of sender-receiver was extended with the addition of an intermediate entity. This entity consists logically of a data management and a context management server. Fig. 2 shows the basic model. A

sender composes an email and defines some contextual attributes. This message is sent and stored in the repository. Recipients on the other hand, communicate their context back to the Context Server which queries the repository for messages relevant to this recipient's current identity, location and time. If a match is found the message is made available to the recipient, who may then access the message using their standard email client.

Fig. 2. LATTE Messaging Model

3.1 Data Management

Message and user management is achieved using two separate repositories. The message repository stores messages that are not yet "valid" (i.e., the context is not yet appropriate for their delivery), so that they can be retrieved later when their context is valid. In this way, messages are treated as a form of persistent data where delivery and expiration depends on the context defined during message composition. This model also allows for standard email extensions such as attachments and encrypted messages to be stored and delivered which constitutes one of the great strengths of using standardised technology like email.

The user repository is responsible for managing the contextual information for all users of the system. It maintains information regarding user profiles, records group memberships and updates records of dynamic contextual information such as recipient location. The LATTE model of registration with the user repository is analogous to the standard model used when someone signs up for an email service. In LATTE, users provide their identities and register with different groups in the same way that people register with mailing-lists and newsgroups. After registration every profile is updated with dynamic contextual information (e.g. location, group membership) coming from the user's device.

3.2 Context Management

The approach used to manage context is a simplified version of the context framework developed in [7]. LATTE's contextual model supports:

- Context Specification by the sender,
- Context processing, storage and evaluation by the Context Engine and
- Context transmission by the recipient.

As was noted in section 2.2.1, users are presented with an intuitive syntax for specifying location context that tries to map physical coordinates to a logical familiar space. Time context is a matter of defining the time the message should be delivered, the period in which the message is valid for further delivery to recipients whose context become appropriate during that time, and a time offset defining repetition intervals.

Context input is retrieved from the recipient when it is dynamic in nature (e.g. location, group) or from the repository when it is static (e.g. identity, time). Time is treated here as static in nature since it does not need to be transmitted from the recipient's device. Dynamic context received from recipients is checked against attributes of the same type stored from the incoming message and if a match is found the server will deliver the message to the appropriate recipient.

4 Implementation

LATTE on the client side can use any email client for receiving email messages. Both a web and a Java interface have been implemented for email composition, as a way to provide an easy and intuitive way for addressing emails based on context. The dissemination of the client's dynamic context information to the LATTE servers uses the STEAM event service [14] described in section 4.5. The event service itself is written in C++, though Java and Perl interfaces are also provided for convenience. The client has been tested on PDAs and laptop computers running Windows, Windows CE or the Linux Operating System.

The Context Engine uses Perl for defining the logic of the core context rules and to communicate to a MySQL database that stores messages and user profiles and interfaces to the SMTP server. Perl was used in the server as the "glue" for tying together all the existing components with the event infrastructure. In this section we describe the overall architecture of LATTE.

4.1 Overall Architecture

The architecture to support the delivery of LATTE messages is depicted in Fig.3. In the current implementation, every LATTE email is sent via SMTP to a special account (e.g. latte@domain.org) at a mail server (see section "4.2 Sender Model"). The current design of LATTE allows multiple servers to be responsible for distinct geographical areas. This means that clients wishing to address an email to a specific geographical area need to be informed of the LATTE server that is available at the desti-

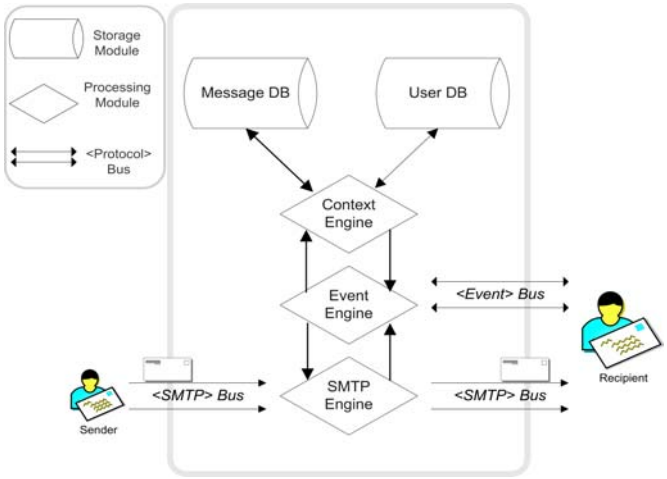

Fig. 3. LATTE Architecture

nation area. A directory based service can be implemented that displays available LATTE servers given a coordinate input. However, this is orthogonal to the LATTE service itself and is not in the current implementation. After the message has been delivered to the designated LATTE server, it is processed and stored in the message database for later retrieval (see section "4.4 Context Engine"). After their initial contact with the LATTE Server, recipients periodically transmit their dynamic context, (i.e., location, group) by using an event-based protocol. The recipient model is described in section 4.3, while the event mechanism in section 4.5.

4.2 Sender Model

Message transmission is achieved through the use of the SMTP protocol [8]. SMTP was chosen for a number of reasons. It provides a familiar interface to users and is interoperable with existing clients. Furthermore, SMTP allows the transmission of messages (emails) that can contain an extended set of headers [12]. These headers are marked as "X-headers", and are designed for clients to transmit user-defined information. LATTE uses these X-headers to capture the required contextual extensions to email, calling them X-LATTE-* headers. The following headers have been defined to encapsulate the required message context from the sender:

- `X-LATTE-Identity` - Specifies the LATTE extensions to identity.
- `X-LATTE-Location` – Specifies the logical location in the INS format explained in section 2.2.1.
- `X-LATTE-Time` – Specifies the time the system should deliver the message.
- `X-LATTE-Duration` – Specifies the time for which the message is valid – new users moving into a valid contextual state in this duration may be considered as a recipient.
- `X-LATTE-Offset` – Allows for a message to be repeated at regular intervals.

LATTE clients may, of course, send LATTE emails using their usual desktop/laptop. However, given the significance of mobility for the LATTE system, we have also implemented a version for the client that uses a PDA. As illustrated in Fig.4, the interface is designed to look and feel like a standard email client, with the additional contextual information added in a similar manner to named recipients, etc. in standard email.

4.3 Recipient Model

The event middleware described in [14] is the underlying technology used by LATTE to notify servers of the dynamic context of clients in the proximity. The same model is also used to notify recipients of the existence of servers and of pending messages. Having such an event infrastructure as the underlying message delivery model allows the participating entities in LATTE to interact in a more autonomous and asynchronous manner. Since the whole system is aimed towards mobile users, these characteristics are even more pertinent. This stateless model of communication also means that users avoid unnecessary registration and de-registration procedures and complex hand-off procedures.

Once the LATTE server knows about the user and the user's context, the only significant difference relating to actually receiving emails is that the recipient will get only messages that are relevant to him. From a technical and interface perspective, receiving emails is the same as standard email – i.e., using POP or IMAP.

Fig. 4. LATTE client interface on an IPAQ

4.4 Context Engine

4.4.1. Context Evaluation

Context evaluation refers to the processing of recipient context against a set of evaluation rules. Resolving location is the most challenging task if it is to be done efficiently and not overload the server. The location model described in section 2.2.1 is encoded in the server as a tree. Each node in the tree represents a geographical area and is identified by its logical name. The client is actually responsible for converting physical coordinates into an INS formatted name. To achieve that, the client downloads a file from the server containing the physical to logical mappings. As a message arrives containing the INS description of a place, this is matched against the nodes in the server's tree. The matching node will then hold a reference to the message's sequence id.

When recipients transmit their location, the tree is traversed and any messages found in the parent and child nodes get tagged as "location ready". Then time and identity of the messages are evaluated and if they match those of the recipient, they are made available for delivery.

4.4.2. Domain Handling

LATTE was designed to operate in a partitioned rather than a flat namespace. Every LATTE server defines an administrative domain which is a geographic area. The rationale behind this is to be able to better handle location addressing and ease of administration. Separate administrative domains allow a more decentralized architecture as well as a finer granularity in the mapping between logical and physical locations.

There is no inherent restriction in the domain size, apart from restrictions in wireless connectivity. Domains can be as large as university campuses or as small as floors in a building. Furthermore, allowing location input from multiple sources, offers greater accessibility to the LATTE service. Provided a mapping exists, users can address their emails to indoors or outdoors locations, offering greater coverage and a more complete service.

However, having different administrative domains can also become problematic. LATTE's decentralized nature can allow geographic areas to be mapped by two or more servers, potentially mapping the same physical area under different logical names. LATTE addresses that, by allowing a client to be bound to multiple LATTE servers. The event infrastructure makes this easier to implement, since location events produced at the client can be disseminated to any server that has declared interest in receiving these events. In Fig. 5, this is depicted with event number 2. When different location mappings are used between different LATTE servers, the client location event contains a tuple with multiple location values.

4.5. Event Infrastructure

We utilise an event-based one-to-many protocol, in order to provide a more flexible and less tightly coupled communication paradigm than the traditional one-to-one.

As shown in [11], this type of publish-subscribe interface is well suited for mobile dynamic environments where frequent topology changes are expected. In addition, the underlying communication model used in event-based middleware maps well to the communication model required by the LATTE system. As shown in Fig. 5, the one-to-many interaction required in many cases such as the dissemination of server announcement events to any subscribed clients or the transmission of message notification events to nodes belonging in the same group are examples of this.

Fig. 5 shows the sequence of events during a default client-server interaction in the LATTE architecture. Both clients and servers are composed of producer and consumer components that communicate through events to achieve transfer of data as well as change notifications. As shown, a server's producer component regularly invokes server notification events. Any client interested in receiving LATTE messages will subscribe through his consumer component to events of this type, thereby receiving such events when inside the server's coverage area. Subsequently, the client will be producing location change events through his producer component each time his location changes. The granularity of what constitutes a location change is configurable and depends on the specific technology used for location input. If the server matches a client's location, identity, time and group context with a pending message, it will deliver a message notification event to the appropriate recipient, allowing him to receive emails in the standard way by running a POP or an IMAP client (see section "4.3 Recipient Model").

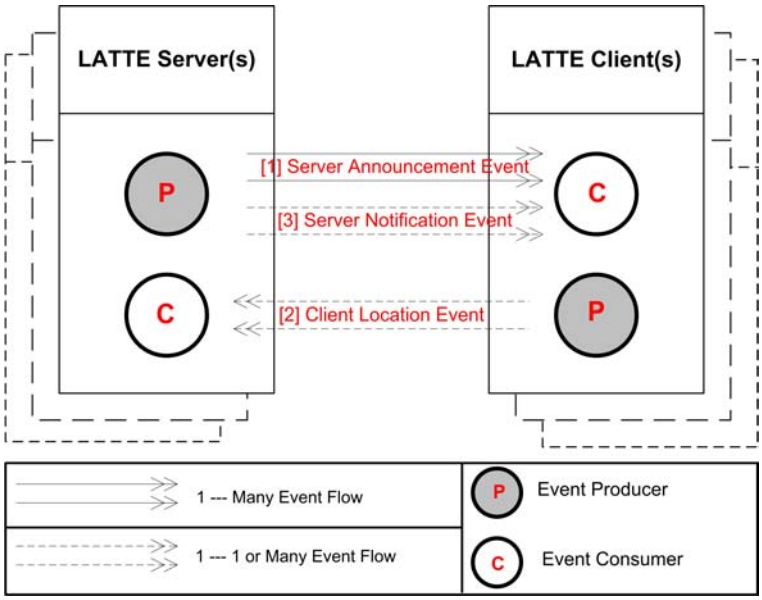

Fig. 5. LATTE's Event Model

5 Related Work

LATTE has three key differences with regards to other location-aware messaging projects. It supports late binding of message identity; it considers multiple types of context for message validation; and it is built using a flexible event infrastructure and extending the open standard SMTP protocol. LATTE is intended as a complement to normal email for providing more relevant information to mobile users. Below we briefly describe and characterise the key differences between LATTE and other relevant projects.

The ActiveCampus project is the implementation of the smart spaces model by San Diego College, University of California [5]. The ActiveCampus system divides its location messaging into two common metaphors: *Messages* and *Graffiti*. The Messages portion of the metaphor models two `buddies' (mutually recognised users) communicating short messages with each other. This method means that only people who know each other through the system can communicate in a one-on-one fashion. This system is similar to most commercial Instant Messaging programs (e.g. ICQ, AIM, etc.) and takes location into account only implicitly, by indicating which buddies are in proximity at a given time.

The second portion of the ActiveCampus messaging model is the *grafitti* system. This system permits the electronic tagging of an object or location within the smart space, allowing messages to be attached to it in an electronic form to be read by all. The dual messaging system employed by ActiveCampus does not permit the use of location as an addressing model. Combined with this, the Graffitti model has only a limited sense of identity. Although this is a simple and clear model, it does not accommodate some of the use case scenarios that a context-aware system supporting identity by context can offer.

The Geonotes system is under development by the HUMLE lab of the Swedish Institute of Computer Science [3]. It is currently rolled out within the Campus of IT-Universitat Kista, Stockholm, in a limited form. The architecture of the system takes into account some of the potential difficulties arising from large volumes of notes attached to a particular location. One feature designed to alleviate the deluge of traffic is the inclusion of a *buddy* system, similar in authentication method to the ActiveCampus method. This buddy system allows for filtering and selection of messages (positively and negatively) at locations.

The Geonotes system utilises a Client-Server model to handle interactions between the author and the annotated location. Identities and buddies are maintained locally on the Client device. While the contextual model for Geonotes is intentionally simple, the adopted design for the LATTE system provides a more formal approach to contextual modelling. Instead of employing a primary metaphor as a basis for behaviour, LATTE follows the basic email metaphor but extended with a well specified contextual structure. In this model, messages are more strictly defined and contexts such as location and identity adopt a canonical form.

The SpaceTags system [4] is very similar to Geonotes in terms of its interaction model, as its aim is to provide an overlaid virtual system on top of the physical world. Its main differentiation is the type of data that can be attached to physical locations are not restricted to messages but can include audio, images, URIs, etc. Furthermore, although it does not support some form of identification, SpaceTags include some no-

tion of time during which they are valid. LATTE offers the same availability of content through the use of MIME attachments and supports identity as part of its contextual attributes.

Websign [13] is another project that matches LATTE closely in the interaction model. It is part of HP's CoolTown project and extends the familiar browsing model with location awareness. Websign has a different system model than LATTE, with messages being cached at the client's device and then being used when their context matches the user's context. Websigns offer spatial and temporal context evaluation and support some notion of groups.

6 Conclusion

This paper has described a simple approach to adding an awareness of context such as location and time to the existing email model. Based on our work on analysing a number of usage scenarios for a context-aware messaging service, we described categories of messages such as alerts, digital signposts and intended delivery. Each of these categories requires a level of context management and reasoning to support a sender sending messages without a need to bind to recipients at message composition time, and also to increase the relevance of messages delivered to recipients. Three important elements of context are relevant for the usage categories – location, time and identity. Using any combination of these elements (except time by itself), determining the appropriate recipient(s) for messages at the latest possible stage results in an email model that provides an intuitive approach to sending messages when the recipients cannot be known at message composition time, and also that is sympathetic to mobile recipients.

While we have identified a number of other approaches to context aware messaging, the main reason LATTE differentiates itself is because it is based on the existing interaction model for email, and was developed by extending an open Internet standard. By taking this approach, LATTE may be deployed on top of existing, widely used email systems, with an intuitive learning curve for users, as the interface simply extends the existing model.

Future work on LATTE will further refine the filtering capabilities of the context engine, evaluate different user interface models for selecting location, and address potential scalability issues.

Acknowledgements

This work is partially supported by the EU FP5 GLOSS Project (IST-2000-26070), in collaboration with The University of Strathclyde, Université Joseph Fourier, and The University of St. Andrew's.

References

1. Abowd, G, Atkeson, C., Hong, J., Long, S., Kooper. R. & Pinkerton, M. (1997) Cyberguide: A mobile context-aware tour guide, *Wireless Networks*, 3 (1997), 421-433.
2. Cheverst, K., Davies, N., Mitchell, K., Friday, A. and Efstratiou, C. (2000) Developing a context-aware electronic tourist guide: some issues and experiences; *Proceedings of the CHI'00*, ACM Press, 17 – 24.
3. Sandin, A, et al. (2001) GeoNotes: Social and Navigational Aspects of Location-Based Information Systems, *Ubicomp 2001*.
4. Tarumi, H., Morishita, K., Nakao, M., Kambayashi, Y. (1999) SpaceTag: An Overlaid Virtual System and its Applications, *IEEE International Conference on Multimedia Computing and Systems Volume I-Volume*.
5. William G. Griswold, Robert Boyer, Steven W. Brown, Tan Minh Truong, Ezekiel Bhasker, Gregory R. Jay, R. Benjamin Shapiro ActiveCampus - Sustaining Educational Communities through Mobile Technology.
6. Dieberger, A., Dourish, P., Höök, K., Resnick, P., and Wexelblat, A. (2000) Social Navigation: Techniques for building more usable systems, in *Interactions*, November-December issue, ACM, 2000.
7. Anind K. Dey, (2000) Providing Architectural Support for Building Context-Aware Applications, Ph.D Thesis, Georgia Institute of Technology.
8 J. Postel. RFC 821: Simple mail transfer protocol. Technical report, DDN Network Information Center, August 1982.
9. William Adjie-Winoto, Elliot Schwartz, Hari Balakrishnan, and Jeremy Lilley. The design and implementation of an intentional naming system. *Proceedings of the seventeenth ACM symposium on Operating systems principles*, pages 186–201. ACM Press, 1999.
10. Bahl, P., and Padmanabhan, V. (200) RADAR: An in-building RF-based user location and tracking system. *Proceedings of IEEE INFOCOM*, volume 2, pages 775–784, March 2000.
11. G. Cugola and E. D. Nitto. Using a Publish/Subscribe Middleware to Support Mobile Computing, Workshop on Middleware for Mobile Computing (IFIP/ACM Middleware 2001), Heidelberg, Germany, 2001.
12. David H. Crocker. Standard for the Format of ARPA Internet Text Messages. Request for Comments 822, DDN Network Information Center, SRI International, August 1982.
13. Pradhan, S., Brignone, C., Cui, J-H., McReynolds, A., Smith, M. (2001) Websign: Hyperlinks from a Physical Location to the Web. Technical Report, HP Laboratories Palo Alto.
14. René Meier and Vinny Cahill. STEAM: Event-Based Middleware for Wireless Ad Hoc Networks. *Proceedings of the International Workshop on Distributed Event-Based Systems (ICDCS/DEBS'02)*. Vienna, Austria, 2002.