A Hybrid Peer-to-Peer Solution for Context Distribution in Mobile and Ubiquitous Environments

Xiaoming Hu[1], Yun Ding[1], Nearchos Paspallis[2], Pyrros Bratskas[2], George A. Papadopoulos[2], Yves Vanrompay[3], Manuele Kirsch Pinheiro[3] & Yolande Berbers[3]

[1] European Media Laboratory GmbH, Schloss-Wolfsbrunnenweg 33,
69118 Heidelberg, Germany. {xiaoming.hu, yun.ding}@eml-d.villa-bosch.de

[2] Department of Computer Science, University of Cyprus, CY-1678, Nicosia, Cyprus.
{nearchos, bratskas,george}@cs.ucy.ac.cy

[3] University of Leuven, Celestijnenlaan, 200A B-3001 Leuven, Belgium. {yves.vanrompay, manuele.kirschpinheiro, yolande.berbers}@cs.kuleuven.be

Abstract. With the proliferation of mobile devices such as PDAs and smartphones, users get accustomed to using them in their daily life. This raises the expectations for user-customized and environment-aware services. However, mobile context-aware systems inherently feature characteristics of distribution and heterogeneity which pose great challenges to their developers. In this paper, we focus on context distribution in mobile and ubiquitous computing environments. After describing the requirements in such environments, we propose a hybrid peer-to-peer based context distribution approach, which is built on top of the JXTA framework, a standard for peer-to-peer systems. We categorize context-aware system entities into three types of peers according to their device capabilities and their roles in context distribution. The peers are able to dynamically discover each other along with their offered services, form groups, and communicate with each other. The proposed approach is evaluated against the derived requirements and illustrated through a motivating scenario.

Keywords: Context awareness, Context distribution, P2P systems, JXTA

1. Introduction

The proliferation of embedded sensors and mobile devices increases the importance of context-aware services. Consider for example a train station where the context conditions are monitored from distributed sensors, travelers are guided according to their current location and trip plan, entertainment programs are proposed to groups of travelers according to their common interest and location, *etc*. Along with a wealth of context information becoming available, a key challenge which is faced by developers is to enable wider dissemination of such context information across distributed sensors, application components and context processing components that manage the flow of context information between the sensors and the applications. Apparently, in mobile and ubiquitous environments, allowing individual context-aware devices to communicate with each other can provide increased levels of synergy in terms of power conservation and reusability (rather than replication) of hardware equipment (such as context sensors). In this respect we here endeavor to study and propose a hybrid peer-to-peer (P2P) [1] infrastructure solution to the problem of context distribution within mobile and ubiquitous computing environments.

Unlike the pure P2P paradigm, which employs the flooding technique in completely decentralized manner to decide how to route query messages by letting nodes broadcast query messages to all of their neighbors, in our hybrid model, some *super-peers* with rich computing resources are selected to maintain meta-information, such as the identity of other peers on which certain context is stored. The same peers are also responsible for processing raw sensor data to generate high-level context information, which can then be retrieved by surrounding resource-limited peers to enable their own context-aware applications.

We performed an early analysis of the basic requirements for a context distribution system by interviewing several pilot application developers, and by examining the state of the art. We have identified the following requirements:

- *Heterogeneity*: Inherently, mobile and ubiquitous computing environments imply the involvement of multiple heterogeneous devices (e.g., context sensors, mobile devices and service nodes), a plethora of available networking configurations and protocols (such as Bluetooth, WiFi, *etc*), as well as different context modeling technologies.
- *Scalability*: Mobile and ubiquitous environments can involve large numbers of participating devices, which implies the need for a decentralized approach enabling scalability. This can be partly achieved by using approaches which enable *localized scalability* [2] for context dissemination.
- *Security*: Security provisions are needed to guarantee the *privacy* of sensitive context information. Such information includes, for example, user-related data like their mood, location, activity and preferences.
- *Robustness*: Due to the mobile nature of targeted environments, context providers and consumers spontaneously appear and disappear. Unreliability of the wireless network connections (e.g., drop of bandwidth and network partition) is generally the rule rather than the exception. Thus, the system must be able to detect these changes dynamically and cope with them.
- *Light-weight*: The context system must be light-weight to be capable of being deployed in embedded devices of various sizes and capabilities.
- *Ease of use*: A main objective of any context distribution system is to reduce the complexity which is inherent in the development of context-aware applications. For this purpose, the context system must be easy to use.

The rest of this paper is organized as follows. We describe our hybrid peer-to-peer infrastructure for context distribution in Section 2. Then we compare our approach with related work in Section 3 and evaluate it against the aforementioned requirements in Section 4. Finally, we summarize our results and provide pointers to our plans for future work in Section 5.

2. A hybrid peer-to-peer based context distribution system

In mobile and ubiquitous environments, context data may be transferred and disseminated among several system entities. Such data can be generated by context sensors, refined and reasoned by context processors, and consumed by context clients. This can be achieved in a highly distributed system using peer-to-peer (P2P) computing. According to their resources (e.g., computing power and memory use) and roles, we classify system entities into three categories (see Fig.1 (a)):

Sensor peers are the sources of context data. They generate low-level raw context data. These distributed sensors can be physical sensors such as thermometers installed in a room or software sensors monitoring the system performance.

Disseminator peers are resource-rich devices which act as context processors, distributors and consumers. Playing an important role in the proposed context distribution system, each disseminator peer has four main responsibilities:

- First, it can search for available sensors to acquire the raw context data.
- Second, it can extract, reason about and transform low-level context data to higher-level abstractions in order to meet the different application needs.
- Furthermore, it can record context data for retrieval of history context data.
- Finally, it delivers and distributes context information to other peers.

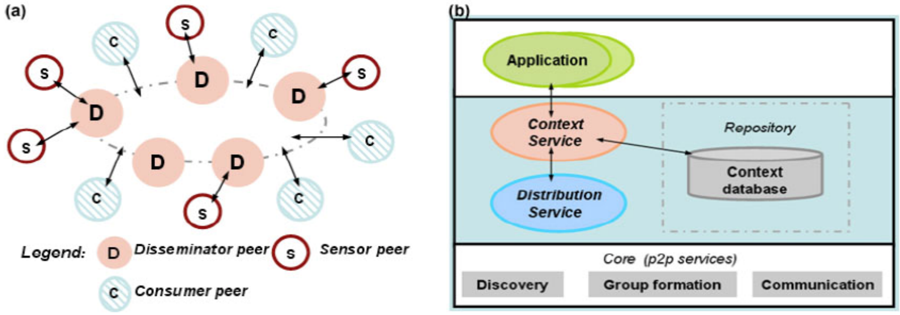


Fig.1: (a) Three types of peers in the context distribution system
(b) A hybrid peer-to-peer infrastructure for context distribution

Consumer peers are only context consumers. Usually, they have limited resources. Thus, they can neither afford to aggregate low-level context data nor reason about it.

2.1 A hybrid peer-to-peer infrastructure for context distribution

Fig.1 (b) illustrates the proposed P2P-based infrastructure for context distribution. This framework is hosted by every disseminator peer, while a consumer peer can omit the repository due to its resource constraints. Adapters or software drivers are necessary to integrate the sensor peers. The infrastructure services are described as follows:

The *Core* provides the essential services required in P2P systems, which can be divided into three categories:

- *Discovery* includes peer discovery and service discovery. It allows a peer to be discovered by other peers, and to search for other peers and services. A peer can actively poll for other peers or services using certain criteria. Moreover, a peer can be notified of the arrival or departure of peers and services. Each peer is associated with a self-description, which is published to announce its presence and the context information it can offer.
- *Group formation* enables peers to create, join and leave peer groups dynamically. Peers of a group will be notified of new or lost group members. A transient peer group can be built to let the peers collaboratively compose and provide a group service. A group self-description is also published for each group.
- *Peer communication* provides communication primitives for a peer to send messages to other peers, or propagate messages within a peer group.

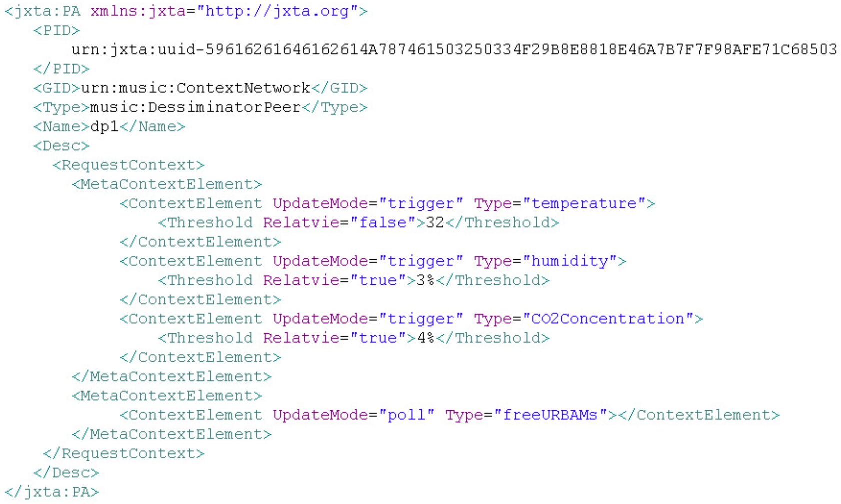
The *Context Service* is designed as a single interface for context providers to insert context information, and for context clients (e.g., a context-aware application) to acquire context information. Context clients can either actively pull context data, or alternatively they can subscribe for context update events which are then pushed to them. Context clients are not concerned about the location of their required context information as this is encapsulated inside the *Context Service* and thus it is transparent for them. The task of the *Repository* is to maintain context information.

The *Distribution Service* is an internal service, which enables federated context services. Context requests which cannot be satisfied by the local *Context Service* are forwarded to connected, remote *Context Services* via the *Distribution Service*. As already mentioned, it remains transparent for context clients whether a context request is satisfied by the local *Context Service* or by a remote one. The Distribution Service is described in more detail in the next section.

2.2 Context network and distribution service

Context Network Consumer peers and disseminator peers dynamically form groups to exchange context data. The *Context Network* is the default group which each peer joins during its bootstrap. This root group is identified by a special group identifier and provides the *Context Service* and the *Distribution Service.* Only members of the *Context Network* group are allowed to access these services. It is worth mentioning that both *Context Service* and *Distribution Service* are so-called "group services" meaning that they are provided collectively by the members of the group. In our case, the disseminator peers dynamically form a federation. If one of them fails, other disseminator peers can still continue providing services to the consumer peers.

Whenever entering into the network, a disseminator peer publishes its self-description (see Fig.2). Each peer has a unique peer ID and by default joins the "*urn:music:ContextNetwork*" group. This default group ID is used to limit the scope of discovery, meaning that only peers having the specified group ID can be discovered. Each disseminator peer also declares the context elements it requires in order to be discoverable by the corresponding sensor peers. Additionally, it searches for existing disseminator peers. Disseminator peers are *super-peers* in the system and stay relatively stable in the network. Each peer maintains a disseminator view which is a list of known disseminator peers in the group. This is achieved by using the *Discovery Service* provided by the *Core*. Since each disseminator peer can disappear and new ones can appear at any time, we use the following algorithm to converge local views of peers. Each disseminator peer periodically computes a list of randomly selected disseminator peers from their local view, and sends the list to a random list of their known disseminator peers. Additionally, each disseminator peer periodically pings a list of randomly selected disseminator peers and purges the non-responding ones from their local view. In this way, the local views are kept loosely consistent among the peers. The algorithm described above is similar to the algorithm used by rendezvous peers of JXTA [3]. In contrast to rendezvous peers, the local view of each disseminator peer includes not only peer information but also the context elements which these peers can provide. In this way, the disseminator peer is easy to locate another proper disseminator peer to forward a context request which can not be satisfied locally.



```
<jxta:PA xmlns:jxta="http://jxta.org">
   <PID>
       urn:jxta:uuid-59616261646162614A787461503250334F29B8E8818E46A7B7F7F98AFE71C68503
   </PID>
   <GID>urn:music:ContextNetwork</GID>
   <Type>music:DessiminatorPeer</Type>
   <Name>dp1</Name>
   <Desc>
     <RequestContext>
       <MetaContextElement>
            <ContextElement UpdateMode="trigger" Type="temperature">
                <Threshold Relatvie="false">32</Threshold>
            </ContextElement>
            <ContextElement UpdateMode="trigger" Type="humidity">
                <Threshold Relatvie="true">3%</Threshold>
            </ContextElement>
            <ContextElement UpdateMode="trigger" Type="CO2Concentration">
                <Threshold Relatvie="true">4%</Threshold>
            </ContextElement>
       </MetaContextElement>
       <MetaContextElement>
            <ContextElement UpdateMode="poll" Type="freeURBAMs"></ContextElement>
       </MetaContextElement>
    </RequestContext>
   </Desc>
</jxta:PA>
```

Fig.2: Self-description of a disseminator peer

Whenever a consumer peer first appears in the network, it joins the *Context Network*. In fact, it connects to one or multiple disseminator peers in this group through the infrastructure. As a result, a virtual group is created on top of the existing P2P infrastructure (see Fig.3 (a)).

Context Service Instances of the *Context Services* are connected via the federation of their associated peers. As illustrated in Fig.3 (a), when a consumer peer searches for a certain context, it uses its *Distribution Service* to propagate a *request* message within the group. Actually, the message is merely sent to its known disseminator peers by utilizing the peer *Communication Service* provided by the *Core* (step 1). Upon receiving the request for context information, the *Context Service* of a disseminator peer checks the local context repository. If the required information is not available, the *Context Service* utilizes the *Distribution Service* to find the information from remotely connected *Context Services* (step 2). Obviously, the *Distribution Service* relies on the P2P interaction among the peers. The receiving remote disseminator peer passes the incoming request to its local *Context Service* for processing, and possibly sends a positive answer directly back to the requesting consumer peer (step 3).

Distribution Service: According to the requirements of the application and the characteristics of context information, the Distribution Service of each peer provides two mechanisms to distribute the context. By *polling*, any context change event will be propagated to remote peers within the group (e.g. any available URBAM[1] public display nearby). And *triggers* are used to enable asynchronous

619

[1] URBAM terminals are special Internet-connected devices which are made available in some RATP Metro stations to offer information services to the passengers.

feedback on specific events (e.g. CO_2 concentration arising above a certain threshold in the waiting hall). The way context dissemination should be handled is also reflected by the "*UpdateMode*" context property in the peer description (see Fig.2).

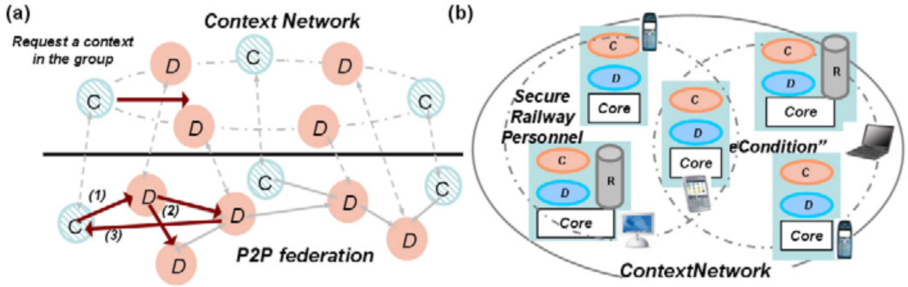


Fig.3: (a) Distribution Service maintains a physical network to a virtual Context Network (b) Hierarchically organized Context Groups

2.3 Scope of context distribution

The default group *Context Network* can be divided into subgroups to restrict information sharing. Each subgroup *Context Group* consists of a dynamic set of peers that have common interests for certain context data, and have agreed upon a set of policies (e.g., group authentication). For example, assume a disseminator peer *A* which produces *environmental condition* data by combining measurements of the ventilated temperature from a thermometer, the humidity from a hygrometer and the CO_2 concentration from an infrared gas analyser. Furthermore, assume that peer *A* would like to share this information with other peers. So peer *A* searches for a subgroup with the name "envCondition", and tries to join it. If this group does not exist, peer *A* creates an "*urn:music:ContextNetwork:envCondition*" group and publishes its group advertisement. Then, another peer *B* can discover this "envCondition" group and become its member, thus enabling the acquisition of shared preprocessed information. Obviously, each peer may belong to several subgroups simultaneously.

Whenever joining the network, a consumer peer registers a list of context data (more precisely, a list of types of context data) of interest to the *Context Service*. During runtime, its interest may change. The *Distribution Service* maps the interests of consumer peers to dedicated subgroups and joins them on behalf of the associated consumer peers.

Groups and subgroups can be hierarchically organized (Fig.3 (b)). A subgroup inherits all the group services of its parent. As a group member, a disseminator peer can propagate messages either to request context information or to notify a context update within the group. *Context Groups* serve to subdivide the context network into regions, providing a scoping mechanism for restricting the propagation of search requests and update events. Thus, context clients will not be bothered to handle irrelevant context information. Context information required by context clients might change from time to time. They join or leave the corresponding *Context Group* according to the information they need to receive.

Moreover, taking into account the protection of private or sensitive context information, a specific policy can be enforced within a Context Group to control access rights. Peers, which would like to join a Context Group, should be approved by a group authentication service. For example, a “Railway Personnel” subgroup provides work forces in the station. Only peers which can provide a registered identifier are permitted to join the group and obtain the right to use these resources.

3. Comparison with related work

A plethora of related work studies both centralized and distributed context systems. According to Baldauf *et al* [4], centralized approaches with one or more centralized components remain the most common in the literature. Centralized context-aware systems use a local service which provides applications with contextual information. Such a local service is usually part of a middleware which acquires raw contextual data from sensors and provides interpreted context to applications via a predefined interface. Furthermore, the middleware is assigned to monitor particular context changes and dispatch relevant events to interested applications as needed. Two well-known examples are the Context Toolkit [5] and CoBrA [6]. In contrast to centralized approaches, distributed context-aware systems allow the generation and management of context information at several locations, thus avoiding potential bottlenecks and unnecessary hardware duplication. Despite the fact that decentralized architectures increase the communication cost, they are more resilient to errors as they do not require a central server to maintain the context information.

The Context Management System of the PACE [7] middleware consists of a distributed set of context repositories. Context-aware components are not statically linked to a single repository, but they can discover repositories dynamically. However, scalability or tolerance for node failures is not provided.

Paganelli *et al* [8] propose a multi-domain approach for context management, in which local domains handle geographically bounded context entities, and application domains manage context entities within organizational or application defined boundaries. However, this approach remains centralized, since context management inside each domain is performed by a server and coordination between different domains is based on a set of Web services allowing the exchange of context information among servers.

A different approach partly based on message multicasts is described by Abdelaziz et al [3]. In this approach clients broadcast their location queries to all the members of a group, while interested parties anonymously listen to the queries. When they match a query and their privacy policy allows it, they reply to the query. The main disadvantage of this approach lies in the increased computation and communication cost of broadcasting context information. This is also the case for the approach proposed by Ye et al [9]. The authors adopt a P2P approach for context sharing, in which context information remains locally stored on the peers and only an access reference, represented by a *Context Database Agent* (CDA), is registered on remote peers. The discovery of new peers is performed by broadcasting messages and context queries to remote peers are sent through unicast messages. The broadcasting of registering messages and the flat structure may raise scalability and security issues in ubiquitous environments, since every peer potentially registers its available context information on every known peer.

Reichert et al [10] propose a distributed architecture with context sources, context processors and context sinks. Context coordination and management are carried out in a distributed and self-organized way by means of P2P overlay networks. A context association is a directed link between a context source and a context sink. A centralized Context Coordinator is used as a registry for context source locations and resolver of context queries. Distributed context querying is possible, but the Context Coordinator stays the central node from which the query starts.

Paspallis et al [11] describe the architecture of the distributed context management system used in the MADAM middleware. Providers and subscribers of context can reside on different, network connected nodes. Localized scalability is enabled by assigning higher importance to local context, and by limiting the number of hops (i.e. transitions) to which specific context types can be communicated. Information in the local repositories is shared between nodes. The approach is based on the periodic broadcast of heartbeat messages which are used for both the formation of loosely coupled membership groups and for disseminating context information of selected types. The main drawback of this approach however, is that it limits the participating nodes to the full MADAM nodes only, which limits the possibility for synergies. In contrast to this approach, this paper proposes an approach which enables devices of varying sizes and capabilities (i.e. both laptops, smart-phones and embedded sensors) to accommodate the required software and act as either context providers, context consumers, or both.

Considering traditional P2P works on content dissemination, Delmastro et al [12] argue that the P2P paradigm is particularly suitable for creating ad hoc networks to share content. They designed a cross-layer optimized protocol (XScribe) to enable P2P multicast services for sharing content among groups of users interested in the same topics. Group membership is managed and a lightweight structure-less approach based on a *Distributed Hash Table* to deliver data to group members is used. This work focuses on the protocol and does not propose architectures for context dissemination.

Bisignano et al [13] have designed a middleware layer over JXTA, named *Expeerience*. This middleware provides high level support for MANET (Mobile ad-hoc network) developers exploiting P2P technology and the mobile agent programming paradigm over mobile ad hoc networks. This approach fulfills requirements for code mobility, discovery of services, and intermittent connectivity.

Bonificacio et al [14] propose a peer-to-peer architecture for distributed knowledge management on top of JXTA in which so called K-peers provide services needed to organize local knowledge from an individual's or group's perspective. Protocols of meaning negotiation are introduced to achieve semantic coordination between peers. The system allows each individual or community of peers to build their own knowledge space within a network of autonomous K-peers, to make knowledge available to other K-peers and to search for relevant knowledge.

The majority of the works in [12][13][14][15] do not explicitly focus on context dissemination or context-aware systems. Consequently, they often ignore intrinsic characteristics of context systems such as the highly dynamic environments, uncertainty and error-proneness which affect context distribution.

4. Evaluation

A prototype of the proposed system is currently under development. The Core functionality has already been implemented (http://www.igd.fhg.de/igd-a1/dynamite-project/software/tools/ubinet.zip). It is based on JXTA, which is a standard for P2P systems. JXTA defines a set of open protocols that allow any devices connected to the network to communicate and collaborate in a peer-to-peer manner [16]. To illustrate the motivation for this work, we present a scenario where context distribution is needed. This example validates the requirements mentioned in Section 1 and explains how the proposed approach satisfies these requirements.

The example is partly based on a pilot application developed by the MUSIC [17] consortium. It refers to an application designed for passengers of the Paris Metro. The application is an intelligent guide which senses the location of the user along with her or his agenda, and automatically displays relevant information. For example, consider a user who is heading to a football game. Since there is sufficient time for a drink before the game, the application shows a map of the area with bars serving beers and snacks. Furthermore, because the user’s smart-phone is equipped with a small display, the smart-phone detects a nearby URBAM terminal and delegates its visual output to it.

To enable this kind of scenario, it is assumed that a number of context information is needed. For instance, a RFID sensor is needed to provide location information (GPS normally does not work in roofed areas such as metro stations or tunnels). Furthermore, a specialized software sensor is needed to access and analyze the user's agenda. For example, such a sensor can be designed to access the user's syndicated calendar feed (as it is available from services like Google's Calendar), and infer the user state. In this example, the system combines the location information (e.g. Metro station adjacent to the football stadium of the upcoming game) and the user's agenda entry (e.g. "attending the PSG football game at 7pm") and, combined with the current time, it infers that Bob is going to the game and that he has one hour free before that. Additional context information needed is the availability of devices and network connectivity in the Metro station. In terms of devices for example, the smart-phone detects a nearby URBAM terminal which is free, and notifies the user about it. The user walks to the URBAM terminal and accepts the smart-phone's suggestion to delegate the user interface (UI) to its touch-screen display. At that point, the UI is delegated to the URBAM terminal and the user analyzes his options for a quick drink and snack before the game.

The proposed hybrid P2P based approach is requirements-driven, and as such, we briefly discuss how these requirements identified in Section 1 have been satisfied by the proposed mechanisms within the scope of this example.

- *Heterogeneity*: Because of the multiple types of devices involved in this scenario (smart-phone, URBAM terminal and Metro RFID sensor), it is important to enable a common method for interoperability among these devices. Our proposal is based on the JXTA framework, which provides different implementations suitable for varying device types like embedded sensors, smart-phones and PDAs, laptops, desktops, *etc*. As such, different and varying implementations of the proposed mechanism can be installed on the RFID sensor, the URBAM terminal and the smart-phone, each one fitting the appropriate category. The former acts as a *sensor peer*, the URBAM terminal as a *disseminator peer*, and the latter as a *consumer peer*. In this paper we are primarily concerned with the *communication-aspects* of heterogeneity, but in practice more aspects need to be tackled. Most notably, interoperability at the *model-level* is needed to guarantee that the devices have a common understanding of the exchanged data. In this perspective, the MUSIC project follows an ontology-based approach [18].
- *Scalability*: Because of the large number of users expected in such scenarios (i.e. passengers with smart-phone trying to detect available RFID sensors or URBAM terminals), it is important that the proposed solution is scalable, allowing dozens or even hundreds of users exchanging context information. Our approach tries to limit the increased communication cost by using a hybrid P2P infrastructure solution. Resource-rich devices running as "super-peers" (disseminator peers) contribute by organizing and limiting the propagation of context information. The feature of group formation prevents the replication of already available context data sensors. In addition, it defines the boundaries for the dissemination of specific context data. This ability, together with the ability of context clients to join and leave groups according

to their context needs, protects from delivering redundant information to uninterested peers.

- *Security*: Context information such as *location* and *available URBAM terminals* can safely be published, because there are no privacy concerns in sharing this information. On the other hand, the user's agenda is private information and thus access to that data should be limited to authorized entities only. To mitigate security concerns, the proposed context distribution system provides the notion of protected groups which provide access to authorized users only. In this example, a Calendar application running on the desktop of the user can form a subgroup containing information about the user's agenda. Access to that group is restricted, and only users with authorized access can join it.
- *Robustness*: Taking advantage of the discovery service provided by JXTA, our proposed system satisfies the requirement for *robustness*, arising from the instability of network connections which is common in mobile networks. This service allows for the discovery of both dynamically appearing and disappearing peers as well as of peer groups. In the proposed scenario, new smart-phones (consumer peers) can connect and disconnect as needed, and dynamically discover URBAM terminals.
- *Light-weight:* The implementation on top of JXME [19], which offers a light-weight JXTA implementation targeting mobile devices, allows the proposed solution to be deployed on resource constrained devices. In this scenario, the distinction between consumer and disseminator peers offers the possibility of selecting the implementation that best fits the capabilities of the devices.
- *Ease of use*: The proposed solution provides a single interface (the *Context Service* interface) that enables prospective context clients and context providers to access (i.e., insert and query) context information both synchronously and asynchronously, in an easy and intuitive manner.

5. Conclusions

This paper introduces a hybrid peer-to-peer based context distribution system. We have argued that this approach offers significant improvements over related solutions. The approach builds on top of a thoroughly-studied standard, the JXTA framework. This framework provides many facilities for discovering, communicating and safeguarding information. We have shown that our approach meets the identified requirements as examined with a motivating scenario. Concerning future plans, the implementation of this approach is already in progress, and plans have been made for validating and testing it in the context of a set of field trials.

Acknowledgements

This paper was financially supported by the European Union (MUSIC-IST project, 6th Framework Programme, contract number 35166). The authors from the European Media Laboratory GmbH are supported by the Klaus Tschira Foundation.

References

1. D. Milojicic, V. Kalogeraki, R. M. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins, and Z. Xu. Peer-to-peer computing. technical report HPL-2002-57 20020315, Technical Publications Department, HP Labs Research Library, Mar. 2002. http://www.hpl.hp.com/techreports/2002/HPL-2002-57.html.
2. Satyanarayanan, M.: Pervasive Computing: Vision and Challenges, IEEE Personal Communications, pp. 10-17 (2001).
3. M. Abdelaziz, E. Pouyoul, B. Traversat, Project JXTA: A Loosely-Consistent DHT Rendezvous Walker (Available at: http://research.sun.com/spotlight/misc/jxta-dht.pdf).
4. Baldauf, M., Dustdar, S., Rosenberg, F.: A survey on context-aware systems. International Journal of Ad Hoc and Ubiquitous Computing, Vol. 2, No. 4, pp.263–277. (2007).
5. Dey, A., Salber, D., Abowd, G.: A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications, Human Computer Interaction, Vol. 16, No. 2-4, pp. 97-166 (2001).
6. Chen, H.: An Intelligent Broker Architecture for Pervasive Context-Aware Systems, PhD Thesis, University of Maryland, Baltimore County. (2004).
7. K. Henricksen, J. Indulska, T. McFadden, S. Balasubramaniam, Middleware for distributed context-aware systems, International Symposium on Distributed Objects and Applications (DOA), 2005.
8. Paganelli, F.; Bianchi, G. , Giuli, D.: A Context Model for Context-Aware System Design Towards the Ambient Intelligence Vision: Experiences in the eTourism Domain. In Stephanidis, C. & Pieper, M. (ed.s), Universal Access in Ambient Intelligence Environments, 9th ERCIM Workshop on User Interfaces for All (ERCIM UI4ALL), Lecture Notes in Computer Science, Vol. 4397, Spring-Verlag, pp. 173-191 (2006).
9. Ye, Jian, Li, Jintao, Zhu, Zhenmin, Gu, Xiaoguang, Shi, Hongzhou: PCSM: A Context Sharing Model in Peer-to-Peer Ubiquitous Computing Environment. International Conference on Convergence Information Technology, 21-23 Nov. 2007, pp.1868-1873 (2007).20. D. Milojicic, V. Kalogeraki, R. M. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins, and Z. Xu. Peer-to-peer computing. technical report HPL-2002-57 20020315, Technical Publications Department, HP Labs Research Library, Mar. 2002. http://www.hpl.hp.com/techreports/2002/HPL-2002-57.html.
10. C. Reichert, M. Kleis, R. Giaffreda, Towards Distributed Context Management in Ambient Networks, 14th IST Mobile & Wireless Communications Summit. Proceedings, 2005.
11. N. Paspallis, A. Chimaris, G. A. Papadopoulos, Experiences from developing a distributed context management system for enabling adaptivity, J. Indulska and K. Raymond (Eds.): DAIS 2007, LNCS 4531, pp. 225–238, 2007.
12. F. Delmastro, A. Passarella, M. Conti, P2P Multicast for pervasive ad-hoc networks, Pervasive and Mobile Computing, Vol. 4,, N. 1, pp. 62-91 (2008).

13. M. Bisignano, A. Calvagna, G. Di Modica, O. Tomarchio, Design and development of a JXTA middleware for mobile ad-hoc networks, 3rd International Conference on Peer-to-Peer Computing (P2P 2003), Linkopings, Sweden, September1-3, 2003.
14. M. Bonifacio, P. Bouquet, G. Mameli, M. Nori, Peer-mediated distributed knowledge management, Technical Report DIT 03-032 Dept. of ICT, University of Trento, 2003.
15. F. Delmastro, A. Passarella, M. Conti, P2P Multicast for pervasive ad-hoc networks, Pervasive and Mobile Computing, Vol. 4,, N. 1, pp. 62-91 (2008)
16. JXTA, https://jxta.dev.java.net.
17. The MUSIC project, http://www.ist-music.eu/
18. Michael Wagner, Roland Reichle, Mohammad Ullah Khan, Kurt Geihs, Jorge Lorenzo, Massimo Valla, Cristina Fra, Nearchos Paspallis, George A. Papadopoulos, A Comprehensive Context Modeling Framework for Pervasive Computing Systems, 8th IFIP International Conference on Distributed Applications and Interoperable Systems (DAIS), 4-6 June, 2008, Oslo, Norway, Springer Verlag, to appear.
19. https://jxta-jxme.dev.java.net/