Context Modelling and Management in Ambient-Aware Pervasive Environments

Maria Strimpakou[1], Ioanna Roussaki[1], Carsten Pils[2], Michael Angermann[3], Patrick Robertson[3], and Miltiades Anagnostou[1]

[1] School of Electrical and Computer Engineering, National Technical University of Athens, 9 Heroon Polytechneiou Street, 157 73, Athens, Greece
{mstrim, nanario, miltos}@telecom.ntua.gr

[2] Department of Computer Science 4, Ahornstr. 55, 52056 Aachen, Germany
pils@i4.informatik.rwth-aachen.de

[3] Institute of Communications and Navigation, German Aerospace Center, D-82234 Wessling/Oberpfaffenhofen, Germany
{michael.angermann, Patrick.Robertson}@dlr.de

Abstract. Services in pervasive computing systems must evolve so that they become minimally intrusive and exhibit inherent proactiveness and dynamic adaptability to the current conditions, user preferences and environment. Context awareness has the potential to greatly reduce the human attention and interaction bottlenecks, to give the user the impression that services fade into the background, and to support intelligent personalization and adaptability features. To establish this functionality, an infrastructure is required to collect, manage, maintain, synchronize, infer and disseminate context information towards applications and users. This paper presents a context model and ambient context management system that have been integrated into a pervasive service platform. This research is being carried out in the DAIDALOS IST Integrated Project for pervasive environments. The final goal is to integrate the platform developed with a heterogeneous all-IP network, in order to provide intelligent pervasive services to mobile and non-mobile users based on a robust context-aware environment.

1 Introduction

Pervasive computing is about the creation of environments saturated with computing and communication capabilities, yet having those devices integrated into the environment such that they disappear [1]. In such a pervasive computing world [2], service provisioning systems will be able to proactively address user needs, negotiate for services, act on the user's behalf, and deliver services anywhere and anytime across a multitude of networks. As traditional systems evolve into pervasive, an important aspect that needs to be pursued is context-awareness, in order for pervasive services to seamlessly integrate and cooperate in support of user requirements, desires and objectives. Context awareness in services is actually about closely and properly linking services, so that their user is relieved from submitting information that already exists in other parts of the global system. In this manner services are expected to act in a collaborative mode, which finally increases the user friendliness. Yet, context awareness cannot be achieved without an adequate methodology and a suitable infrastructure. In this framework, the European IST project DAIDALOS[1] works on the adoption of an end-to-end approach for service provisioning, from users to service providers and to the global Internet. DAIDALOS aims to design and develop the necessary infrastructure and components that support the composition and deployment of pervasive services. The middleware software system currently being developed provides, amongst others, the efficient collection and distribution of context information. Eventually, DAIDALOS will offer a uniform way and the underlying means that will enable users to discover and compose services, tailored to their requirements and context, while also preserving their privacy.

Quite a few articles in the research literature on context awareness have outlined the benefits of using context and have proposed various, albeit similar to each other, context definitions. Most popular is the definition of Dey [3]: “Context is any information that can be used to characterise the situation of an entity.” Evidently, context comprises a vast amount of different data sources and data types. Thus, collection, consistency, and distribution of context data is challenging for the development of context aware systems. This paper presents how the DAIDALOS middleware addresses these challenges in pervasive computing environments. The rest of the paper is structured as follows. Section 2 gives a short overview of context-aware approaches known from research literature and outlines their advantages and shortcomings. Subsequently, in Section 3 the DAIDALOS context data model is described and Section 4 proposes a context management infrastructure. Sections 5 and 6 present how DAIDALOS captures context information and ensures the consistency of the data. Finally, Section 7 concludes the paper and gives an outlook towards future work.

2 Context-Aware Systems Overview

In this section, a short overview of the most important context-aware systems is provided, while the wide variety of fields where context can be exploited is identified.

Early work in context awareness includes the Active Badge System developed at Olivetti Research Lab to redirect phone calls based on people’s locations [4]. Subsequently, the ParcTab system developed at the Xerox Palo Alto Research Center in the mid 90’s could be considered as an evolution of the active badge that relied on PDAs to support a variety of context-aware office applications [5]. A few years later, Cyberdesk [6] built an architecture to automatically integrate web-based services based on virtual context, or context derived from the electronic world. The virtual context was the personal information the user was interacting with on-screen including email addresses, mailing addresses, dates, names, URLs, etc. While Cyberdesk could handle

[1] This work has been partially supported by the Integrated Project DAIDALOS (“Designing Advanced network Interfaces for the Delivery and Administration of Location independent, Optimised personal Services”), which is financed by the European Commission under the Sixth Framework Programme. However, this paper expresses the authors’ personal views, which are not necessarily those of the DAIDALOS consortium.

limited types of context, it provided many of the mechanisms that are necessary to build generic context-aware architectures. The Cyberguide application [7] enhanced the prevailing services of a guidebook by adding location awareness and a simple form of orientation information. The context aware tour guide, implemented in two versions for indoor and outdoor usage, could give more precise information, depending on the user's location. In general, tour guidance is a popular context-aware application, which has been explored by several research and development groups.

The Ektara architecture [8] is a distributed computing architecture for building context-aware ubiquitous and wearable computing applications (UWC). Ektara reviewed a wide range of context-aware wearable and ubiquitous computing systems, identified their critical features and finally, proposed a common functional architecture for the development of real-world applications in this domain. At the same time, Mediacup [9] and TEA [10] projects tried to explore the possibility of hiding context sensors in everyday objects. The Mediacup project studied capture and communication of context in human environments, based on a coffee cup with infrared communication and multiple sensors. Using the Mediacup various new applications were developed exploiting the context information collected. The TEA project investigated "Technologies for Enabling Awareness" and their applications in mobile telephony, building a mobile phone artefact.

Other interesting examples of context management are provided by Owl context service, Kimura System and Solar. Owl [11] is a context-aware system, which aimed to gather, maintain and supply context information to clients, while protecting people's privacy through the use of a role-based access control mechanism. It also tackled various advanced issues, such as historical context, access rights, quality, extensibility and scalability [12]. Kimura System [13], on the other hand, tried to integrate both physical and virtual context information to enrich activities of knowledge workers. Finally, Solar [14] is a middleware system designed in Dartmouth College that consists of various information sources, i.e., sensors, gathering physical or virtual context information, together with filters, transformers and aggregators modifying context to offer the application usable context information.

A quite promising approach to context-awareness was introduced by the Context Toolkit [3] that isolated the application from context sensing. The proposed architecture was based on abstract components named context widgets, interpreters and aggregators that interact, in order to gather context data and disseminate it to the applications. On the other hand, the Aura Project [15] at Carnegie Mellon University (CMU) investigated how applications could proactively adapt to the environment in which they operated. A set of basic "contextual services" was developed within Aura, in order to provide adaptive applications with environmental information. While the Context Toolkit focused on developing an object oriented framework and allowed use of multiple wire protocols, Aura focused on developing a standard interface for accessing services and forced all services and clients to use the same wire protocol. This sacrificed flexibility, but increased interoperability.

HotTown [16] is another project that developed an open and scalable service architecture for context-aware personal communication. Users and other entities were represented by mobile agents that carried a context knowledge representation reflecting the capabilities of the entity and associated objects and relations between them. In HotTown, entities could exchange context knowledge, merge it with existing knowledge, and interpret context knowledge in the end devices. The Cooltown project by HP labs attempted to solve the problems of representing, combining and exploiting context information, and by introducing a uniform Web presence model for people, places and things [17]. Rather than focusing on creating the best solution for a particular application, Cooltown concentrated on building general-purpose mechanisms common to providing Web presence for people, places, and things. The Cooltown architecture was deployed for real use within a lab.

Other current research activities include the CoBrA, SOCAM, CASS and CORTEX projects. CoBrA (Context Broker Architecture) [18] is an agent based architecture supporting context-aware computing in intelligent spaces populated with intelligent systems that provide pervasive computing services to users. CoBrA has adopted an OWL-based ontology approach and it offers a context inference engine that uses rule-based ontology reasoning. The SOCAM (Service-oriented Context-Aware Middleware) project [19] is another architecture for the rapid prototyping and provision of context-aware mobile services. It is based on a central server that retrieves context data from distributed context providers, processes it and offers it to its clients. Third party mobile services are located on top of the architecture and use the different levels of available context information to adapt their behavior accordingly. SOCAM also uses ontologies to model and manage the context data and has implemented a context reasoning engine. Another scalable server-based middleware for context-aware mobile applications on hand-held and other small mobile computers is designed within the CASS (Context-awareness sub-structure) project [20]. CASS enables developers to overcome the memory and processor constraints of small mobile computer platforms while supporting a large number of low-level sensor and other context inputs. It supports high-level context data abstraction and separation of context-based inferences and behaviours from application code, thus opening the way for context-aware applications configurable by users. The CORTEX project has built context-aware middleware based on the Sentient Object Model [21]. It is suitable for the development of context-aware applications in ad-hoc mobile environments and allows developers to fuse data from disparate sensors, represent application context, and reason efficiently about context, without the need to write complex code. It provides an event-based communication mechanism designed for ad-hoc wireless environments, which supports loose coupling between sensors, actuators and application components.

Finally, one of the most recent projects that focused on context-awareness is the IST CONTEXT project [22]. Its main objective is the specification and design of models and solutions for an efficient provisioning of context-based services making use of active networks on top of fixed and mobile infrastructure. CONTEXT has proved that active networks are also powerful in context-aware systems for tackling the issues of context distribution and heterogeneity. Furthermore, via the CONTEXT platform it has been demonstrated that gathering and disseminating context using active networks is efficient with regards to traffic and delay parameters.

3 A Context Model for Pervasive Systems

An efficient context model is a key factor in designing context-aware services. Relevant research efforts have indicated that generic uniform context models are more useful in pervasive computing environments, in which the range and heterogeneity of services is unique. In [23] a survey of the most important context modelling approaches for pervasive computing is provided. These approaches have been evaluated against the requirements of pervasive services and the relative results are presented in the following table [23].

Table 1. Evaluation of the existing context modelling schemes against the pervasive computing requirements

Pervasive Computing Req/ments / Context Modelling Approach	distributed composition	partial validation	richness & quality of information	incompleteness and ambiguity	level of formality	applicability to existing environments
Key-Value Models	-	-	- -	- -	- -	+
Mark-up Scheme Models	+	+ +	-	-	+	+ +
Graphical Models	- -	-	+	-	+	+
Object Oriented Models	+ +	+	+	+	+	+
Logic Based Models	+ +	-	-	-	+ +	- -
Ontology Based Models	+ +	+ +	+	+	+ +	+

In DAIDALOS, we developed a context model adequate for the pervasive service platform designed, which demonstrates features of both the graphical and the object oriented models, while it can be extended to incorporate a context ontology. The DAIDALOS Context Model (DCM) addresses, to some degree, all the pervasive computing requirements in Table 1.

A simplified class diagram of the DCM is depicted in Figure 1. This approach is based on the object-oriented programming principles. The DCM is built upon the notion of an **Entity**, which corresponds to an object of the physical or conceptual world. Each Entity instance is associated with a specific **EntityType**, e.g., person, service, place, terminal, preferences, etc, modelled by the homonymous class. Entities may demonstrate various properties, e.g., "height", "colour", "address", "location", etc, which are represented by **Attributes**. Each Attribute is related to exactly one Entity, and belongs to a specific **AttributeType**. An Entity may be linked to other Entities via **DirectedAssociations** (DirAs), such as "owns", "uses", "located in", "student of", etc, or **UndirectedAssocations** (UndirAs), such as "friends", "team-mates", etc. The DirAs are relationships among entities with different source and target roles. Each DirA originates at a single entity, called the parent entity, and points to one or more entities, called child entities. The UndirAs do not specify an owner entity, but form generic associations among peer entities. All Entities, Attributes and Associations are marked with a timestamp indicating their most recent update time. The Attributes and Associations also have an activation status boolean parameter, which indicates whether or not these instances are currently activated. In the set of Attributes of the same type that belong to a specific Entity, only one may be activated at a given time. All Entities, Attributes and Associations implement an XML interface, which enables them to be (de)serialised for the purpose of remote communication.

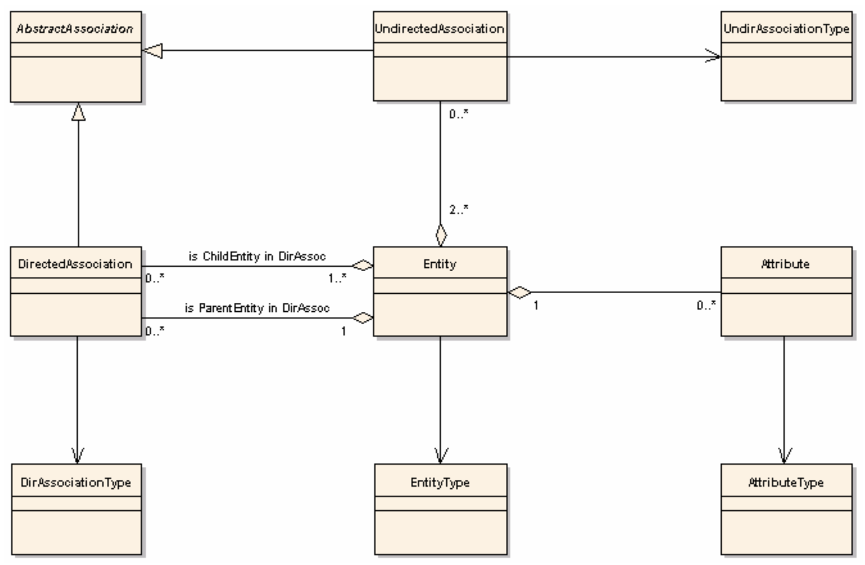

Fig. 1. The DAIDALOS Context Model

The advantages of the DCM are outlined in the following. First, it is sufficiently generic and extendable in order to capture arbitrary types of context information. Second, via the timestamp property, it addresses the temporal aspects of context, a quite critical feature in pervasive computing systems as context information may change constantly and unpredictably. Third, it supports activation/deactivation functionality to indicate the current context value or user selection. Fourth, it addresses the fact that context information is usually highly interrelated, introducing various types of associations between entities. Fifth, it could be seen as a distributed model, where each entity contains its own information and the type of relationship with other entities. Sixth, it is flexible and scalable, as the addition/deletion of an entity does not require the modification of the content or status of the existing entities.

Future plans involve the extension of the DCM in order to support multiple representations of the same context in different forms and different levels of abstraction, and should be able to capture the relationships that exist between the alternative representations. New classes will be added, derived from the Attribute base class, which will be used to add logic to the context data. These classes will serve as “context data translators”, e.g., a class TemperatureAttribute will provide methods for accessing temperature information encoded either in Celsius or Fahrenheit degrees. Furthermore, in future versions the DCM will be extended to express the fact that context may be imperfect, i.e., outdated, inaccurate, or even unknown. To this effect, context quality parameters will be included and stochastic modelling principles will be used.

4 A Context Awareness Enabling Architecture

The Context Management Subsystem (CMS) establishes the context-awareness related functionality in the DAIDALOS pervasive services platform thoroughly described in [24]. It provides context consumers with a uniform way to access context information, thus hiding the complexity of context management. Furthermore, it offers streamlined mechanisms for large-scale distribution and synchronization of personal repositories across multiple administrative domains. The set of the software components that constitute the CMS and offer the above functionality are depicted in Figure 2 and are briefly described in the following paragraphs.

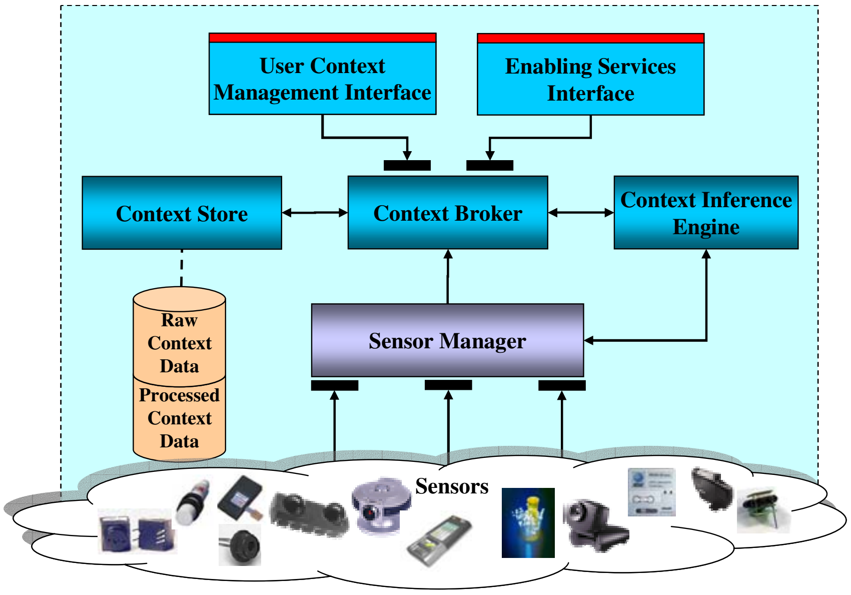

Fig. 2. The Context Management Subsystem architecture

The *Context Broker* (CoB) answers context requests and serves as the access point to the context data. A context consumer requests context from the broker that handles the subsequent negotiation. This negotiation is based on the requestor's context requirements (e.g., type, accuracy, cost, time of update, priorities), the context owner's authorization settings (e.g., access rights for privacy) and the available context and its quality parameters. Based on this information, it decides on the most appropriate context source among multiple sources and providers. It may retrieve the context information from the local Context Store, from the inference engine, from peer CoBs or from another appropriate context provider. CoBs are also responsible for maintaining the consistency of replicated context entities, an issue further examined in a subsequent section. Finally, upon addition, update or removal of context data, the CoB triggers the pertinent subscribed context events using the designed notification mechanisms.

The *Inference Engine* (IE) infers additional and not directly observable context information. It enhances the intelligence of the system as it produces useful information and extracts patterns that have high added value for both consumers and providers. A chain of components is set up to allow the enrichment of context, by combining, converting, or refining context. The functionality of the IE is very important for pervasive services provision, as it greatly contributes to the minimisation of user interaction with the pervasive computing system. A typical example of inference is that of estimating the activity of a person given some lower level sensor data.

The *Context Store* retrieves context from the Sensor Manager (raw data) and the Inference Engine (context data inferred), processes this data, stores it to the appropriate repositories and updates the context databases. Context retrieval implies pulling data or receiving pushed data and converting the data in the context system's uniform format. The Context Store provides context to the CoB on demand.

The *Sensor Manager* keeps track of the sensors which are associated with or attached to the local host, configures them according to the device's requirements, and updates new sample data in the context system. Since the functionality of the Sensor Manager is quite important for the support of context-awareness in pervasive environments, this entity will be studied in more detail in the following section.

Two main interfaces are offered to actors or other entities outside the CMS by the CoB: the User Context Management Interface and the Enabling Services Interface. The *User Context Management Interface* allows a context consumer (user, service, content and context provider) to configure any data previously defined. The *Enabling Services Interface* is offered to the rest of the components in the pervasive system architecture in order to support and control access to the context data maintained by the platform. The designed CMS interacts and cooperates with peer systems in foreign domains via service level agreements and federation mechanisms negotiating the provision/acquisition of additional context information.

Both the presented DCM and a simplified version of the described CMS architecture have already been evaluated in a prototype implementation of the DAIDALOS pervasive service platform. This prototype was built on an OSGi Service Platform [25]. The OSGi™ specifications define a standardized, component oriented, computing environment for networked services. For remote communication SOAP [26] was used, while the discovery of services and components was based on the SLP protocol [27].

5 Sensor Management Supporting Context-Aware Services

Primarily, dynamic information is collected by sensors that monitor the state of resources or the environment at regular intervals. In general, sensors are either attached to a device (e.g., a mobile terminal's CPU load sensor), or are at least associated with a device (e.g., a location or temperature sensor). Since devices may have different configuration requirements with respect to sensors, each CMS has a Sensor Manager (SM). The SM keeps track of the sensors which are associated with or attached to the local host, configures them according to the device's requirements, and updates new sample data in the CMS. Consequently, when a new sensor is activated it must first register with a SM. On receiving a registration request, the SM adds the sensor's identification to the set of active sensors. Subsequently, it retrieves a suitable sensor configuration from the CMS and forwards it to the sensor. A sensor configuration comprises the required sample rate, the accepted sample types (given that the sensor samples different types of data), or some thresholds. Generally, the SM requires that sensors send sample data at regular intervals, allowing the SM to keep track of the sensor state. When the SM does not receive sample data within a certain time frame, it considers the sensor to be inactive and updates its state in the CMS database accordingly. In that sense, the SM is also a data source and can be considered to be a virtual sensor. On receiving sample data updates the SM inspects the sample source and its type and stores it in the CMS database. To this end, each sensor has a corresponding entity data object in the CMS database. Moreover, the SM fires an event and dispatches it to the Inference Engine to inform it about the update. On receiving the event the Inference Engine must decide whether other entity objects must be updated as a consequence.

In general, a sensor (e.g., temperature or GPS sensor) is attached and thus associated with a single device. Obviously, there is also a strong semantic and functional association between the device and the sensor. That is, the sample data is used predominantly by the local CMS and rarely retrieved by remote ones. Yet there is another kind of sensor which is characterised by its association with multiple devices. A location sensor located in the supporting environment which samples location information of mobile devices based on signal strength measurement or on the cell of origin is associated with multiple devices. Such a sensor is also registered with multiple SMs and has thus multiple configurations. To this end, this sensor implements handlers which keep track of the state information of individual SMs. When a device location should be determined with the help of a sensor which serves multiple devices, the SM must look-up a suitable location sensor. Sensor look-up is supported by the DAIDALOS service discovery mechanism. Once a suitable sensor reference has been determined, the SM registers with it. Based on the two sensor concepts, the SM gains the flexibility required for capturing the different data types from various information sources.

6 Consistency of Context Data

The research activity in the field of context awareness is very intense, while various context-aware application paradigms have been introduced to the wide market. The supply and demand for context-aware services are now considerable and are estimated to increase significantly in the years to come. Due to the existence of geographically wide business domains and the extended inter-domain activities, context management systems are expected to collect, store, process, and disseminate context information from data generated at geographically dispersed sites. In such large-scale distributed systems the cost of remote communication is sufficiently high to discourage the system relying on a context entity hosted at just a single store. Thus, the maintenance of multiple context entity replicas across several administrative domains may improve the system performance with regards to the context retrieval delay. Nevertheless, the existence of entity replicas requires the adoption of appropriate synchronisation mechanisms to ensure data consistency. To enforce concurrency control, update transactions must be processed on all entity replicas upon the update of any single replica.

The problem of strong [28] or weak [29] replica consistency maintenance has been studied in various domains since the introduction of digital information. The algorithms and protocols designed to solve this problem may be classified in two main categories: the pessimistic and optimistic ones [30]. Pessimistic strategies prevent inconsistencies between replicas by restraining the availability of the replicated system, making worst-case assumptions about the state of the remaining replicas. If some replica operation may lead to any inconsistency, that operation is prevented. Thus, pessimistic strategies provide strong consistency guarantees, while the replicated system's availability is reduced, as any failure of some replica broker or node freezes the distributed (un)locking algorithms [31]. On the other hand, optimistic strategies do not limit availability. All requests are served as long as the replica broker is accessible, as the requestor does not have to wait for all replicas to be locked before performing the update operation. However, this results in weaker consistency guarantees, and in order to restore replica consistency, replica brokers must detect and resolve conflicts [32].

The DAIDALOS CMS prototype deals with update control of context information replicas across multiple remote repositories. The mechanism used to synchronise context information in the distributed CMS databases (DBs) is based upon the employment of "locks", and bears close resemblance to the "single writer multiple readers" token protocol [33], a quite popular pessimistic strategy. In the designed scheme, a single master copy of each context entity exists, while every CMS has to store a list of references to the entity replicas of all the master entity copies it maintains. Each time a request for a non-locally available context entity is performed, the local CMS contacts the affiliated site holding the master entity, which then stores the requestor CMS's address and delivers a replicated copy. Following the GSM principles [34], the home location (HL) concept has been used to signify the CMS of the master entity. The HL address (i.e., the CMS URL) has been encapsulated in the context information identifier. Thus, the HL of the master copy is obtained by interrogating the context identifier. Each context entity has a single associated token or lock, which allows only the replica holding it to update the entity's master copy, while CoB's of several CMSs can simultaneously access a particular entity master copy (get/read operation) without any restriction. If many CoBs are interested in updating a specific context entity, they have to wait until the entity lock is released. This locking mechanism is essential in distributed context management systems, as it ensures the concurrent synchronisation of context updates originating from various and heterogeneous sources such as sensors, network, users and applications. The lock management current implementation is quite simple, and the update mechanism addresses all replicas irrespective of the context properties and context consumers/sources profiles. We are currently working on exploiting consistency control and replica dissemination algorithms originating in the distributed DB domain, but considering the specific nature of context information.

Maintaining entity replicas in distributed context aware systems involves tradeoffs between storage, processing, communication, and outdated information costs, and it should consider the anticipated rate of incoming context update & retrieval requests. Designing competent schemes for consistent replication of context entities distributed throughout the network becomes imperative, considering the dynamic nature and other inherent features of context. For instance, the location information of people travelling on high speed trains is so frequently updated, that the cost of constantly updating the relative entity replicas is prohibitively high. To reduce the respective overall context update & retrieval cost, efficient algorithms should be employed in distributed context management systems, before they are introduced in the wide market.

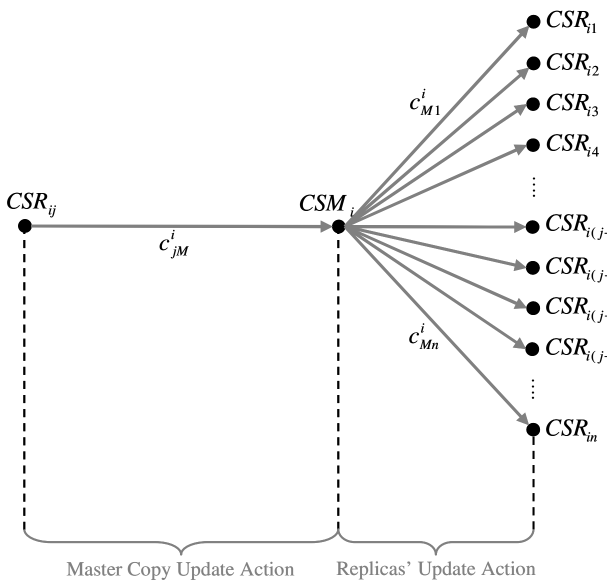

Fig. 3. Context update actions

In our framework, two optimisation problems can be distinguished, that aim to minimise the distributed context management costs (Figure 3). The first one deals with the decision making in the entity-replica-holding CMS, where the context update request is originated. If an update request of context entity i is triggered in the j^{th} CMS domain (CSR_{ij}) that holds a replica of i, the problem is reduced to deciding whether or not it is more efficient to propagate the updated value of i to the master copy (CSM_i), given the estimated retrieval and update requests per time unit, the communication cost, the outdated information cost, and the current and former values of context information (*Master Copy Selective Update Problem*). Given the same parameters, the second problem is reduced to the minimisation of cost during the dissemination of updated context information by the master-copy-holding CMS (*CSM_i*), to the CMSs maintaining the entity replicas (*CSR_{ik}*) (*Replica's Selective Update Problem*). In future versions of the CMS prototype, we plan to study, design, evaluate and enhance algorithms that solve the combination of the aforementioned problems (*End-to-End Selective Update Problem*). To this respect, epidemic propagation [35], lazy update techniques [36], gossip algorithms [37], and antientropy schemes [38] will be considered.

7 Conclusions and Future Work

In this paper a context management architecture has been presented, adequate for large scale ambient systems that have a strong relation to heterogeneous and distributed networks. Since the system is embedded in a larger framework comprising service discovery and composition, configurable networks, distributed event handling and sensor networks, we have aligned the architecture to provide accessible interfaces to both the context consuming services and the context providers. An object oriented context model has been developed, which addresses the main requirements for context representation, distribution and validation, as well as requirements for the provision of pervasive services and applications. The architecture supports context refinement, uniform access to context data by context consumers, finely grained privacy restrictions on context access, as well as federation of context repositories.

Future plans mainly focus on two research areas. The first will address the context model itself and the inclusion of suitable context ontologies. On the one hand we will need to provide the flexibility to include any kind of domain specific ontologies on behalf of the system operator and/or end-users. On the other hand the scenarios we intend to demonstrate at a project wide level need the support of suitable ontologies which have to be selected and possibly modified. We also plan to extend the model to support multiple representations of essentially the same context information and to treat different levels of abstraction. This is an essential element to support features such as inference. Finally, the model needs to be enhanced so as to represent uncertain, partial or outdated context information, also an important precondition for inference. Our second field will be that of enhancing the interrelation and functions of the context management systems, such as improved and cost aware context synchronisation, issues on the performance of large scale context deployment serving potentially millions of active entities and countless sensors. This task aims to provide a solid base on which future operators can build robust and scalable context systems. As indicated above, the topic of context inference will be addressed allowing existing and future inference algorithms to be employed uniformly.

References

[1] Gupta, P., Moitra, D.: Evolving a pervasive IT infrastructure: a technology integration approach. Personal and Ubiquitous Computing Journal. Vol. 8. No. 1. (2004) 31–41

[2] Satyanarayanan, M.: Pervasive Computing: Vision and Challenges. IEEE Personal Communications Magazine. Vol. 8. No. 4. (2001) 10–17

[3] Dey, A., Salber, D., Abowd, G.: A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications. Human-Computer Interaction Journal, Vol. 16. No. 2-4. (2001) 97–166

[4] Want, R., Hopper, A., Falcao, V., Gibbons, J.: The active badge location system. ACM Transactions on Information Systems. Vol. 10. No 2. (1992) 91–102

[5] Schilit, B., Adams, N., Want, R.: Context-Aware Computing Applications. Proc. of the Workshop on Mobile Computing Systems and Applications, IEEE Computer Society, Santa Cruz (1994) 85–90

[6] Dey, A., Abowd, G., Wood, A.: CyberDesk: A Framework for Providing Self–Integrating Context–Aware Services. Knowledge Based Systems, Vol. 11. No. 1. (1998) 3–13

[7] Abowd, G.D., Atkeson, C.G., Hong, J., Long, S., Kooper, R., Pinkerton, M.: Cyberguide: A mobile context-aware tour guide. ACM Wireless Networks. Vol. 3. No 5. (1997) 421–433

[8] DeVaul, R., Pentland, A.: The Ektara Architecture: The Right Framework for Context-Aware Wearable and Ubiquitous Computing Applications. The Media Laboratory, MIT (2000)

[9] Beigl, M., Gellersen, H.W., Schmidt, A.: Mediacups: Experience with design and use of computer-augmented everyday objects. Computer Networks, Vol. 35. No. 4. Elsevier (2001) 401–409

[10] Technology for Enabling Awareness (TEA), http://www.teco.edu/tea/.

[11] Lei, H., Sow, D.M., Davis II, J.S., Banavar, G., Ebling. M.R.: The design and applications of a context service. ACM SIGMOBILE Mobile Computing and Communications Review, Vol. 6. No. 4. (2002) 45–55

[12] Henricksen, K., Indulska, J., Rakotonirainy, A.: Modeling context information in pervasive computing systems. 1st International Conference on Pervasive Computing, Springer, Lecture Notes in Computer Science. Springer-Verlag, Zurich, Switzerland (2002) 167–180

[13] Voida, S., Mynatt, E., MacIntyre, B., Corso, G.: Integrating virtual and physical context to support knowledge workers. IEEE Pervasive Computing, Vol. 1. No. 3. (2002) 73–79

[14] Chen, G., Kotz, D.: Context-sensitive resource discovery. First IEEE International Conference on Pervasive Computing and Communications. (2003) 243–252

[15] Garlan, D., Siewiorek, D., Smailagic, A., Steenkiste, P.: Project Aura: Towards Distraction-Free Pervasive Computing. IEEE Pervasive Computing, Vol. 1. No. 2. (2002) 22-31

[16] Kanter, T.: Attaching Context-Aware Services to Moving Locations. IEEE Internet Computing, Vol. 7. No. 2. (2003) 43–51

[17] Kindberg, T, Barton, J., Morgan, J., Becker, G., Caswell, D., Debaty, P., Gopal, G., Frid, M., Krishnan, V., Morris, H., Schettino, J., Serra, Spasojevic, M.: People, Places, Things: Web Presence for the Real World. ACM MONET (Mobile Networks & Applications Journal) Vol. 7. No. 5. (2002) 365–376

[18] Chen, H., Finin, T., Joshi, A.: An ontology for context-aware pervasive computing environments. Special Issue on Ontologies for Distributed Systems, Knowledge Engi-neering Review. Vol. 18. No. 3. (2004) 197–207

[19] Gu, T., Pung, H.K., Zhang, D.Q.: A Service-Oriented Middleware for Building Context-Aware Services. Journal of Network and Computer Applications (JNCA). Vol. 28. No. 1. (2005) 1–18

[20] Fahy, P. and Clarke, S. CASS: Middleware for Mobile Context-Aware Applications. ACM MobiSys Workshop on Context Awareness, Boston, USA (2004).

[21] Biegel, G., Cahill, V.: A Framework for Developing Mobile, Context-aware Applications. 2nd IEEE Conference on Pervasive Computing and Communications, Orlando, FL, USA (2004)
[22] Xynogalas, S., Chantzara, M., Sygkouna, I., Vrontis, S., Roussaki, I., Anagnostou, M.: Context Management for the Provision of Adaptive Services to Roaming Users, IEEE Wireless Communications. Vol. 11. No. 2. (2004) 40–47
[23] Strang, T., Linnhoff-Popien C.: A Context Modeling Survey. Workshop on Advanced Context Modelling, Reasoning and Management as part of UbiComp 2004 - The Sixth International Conference on Ubiquitous Computing, Nottingham/England (2004)
[24] B. Farshchian, B., Zoric, J., Mehrmann, L., Cawsey, A., Williams, H., Robertson, P., Hauser, C.: Developing Pervasive Services for Future Telecommunication Networks. IADIS International Conference WWW/Internet, Madrid, Spain (2004) 977–982
[25] OSGi Alliance, http://www.osgi.org/.
[26] Simple Object Access Protocol (SOAP), http://www.w3.org/TR/soap/.
[27] Guttman, E.: Service Location Protocol: Automatic Discovery of IP Network Services. IEEE Internet Computing. Vol. 3. No. 4. (1999) 71–80
[28] Triantafillou, P., Neilson, C.: Achieving Strong Consistency in a Distributed File System. IEEE Transactions on Software Engineering, Vol. 23. No. 1. (1997) 35–55
[29] Lenz, R.: Adaptive distributed data management with weak consistent replicated data. ACM symposium on Applied Computing, Philadelphia, USA (1996) 178–185
[30] Ciciani B., Dias, D.M., Yu, P.S.: Analysis of Replication in Distributed Database Systems. IEEE Transactions on Knowledge and Data Engineering, Vol. 2. No. 2. (1990) 247–261
[31] Wiesmann, M., Pedone, F., Schiper, A., Kemme, B., Alonso, G.: Understanding replication in databases and distributed systems. 20th International Conference on Distributed Computing Systems, Taipei, Taiwan (2000) 264–274
[32] Parker, D.S., Popek, G.J., Rudisin, G., Stoughton, A., Walker, B., Walton, E., Chow, J., Edwards, D., Kiser, S., Kline, C.: Detection of mutual inconsistency in distributed systems. IEEE Transactions on Software Engineering. Vol. 9. No. 3. (1983) 240–247
[33] Li, K., Hudak, P.: Memory coherence in shared virtual memory systems. ACM Transactions on Computer Systems. Vol. 7. No. 4. (1989) 321–359
[34] Mouly, M., Pautet, M.B.: The GSM System for Mobile Communications. Telecom Pub (1992)
[35] Holliday, J., Steinke, R., Agrawal, D., Abbadi A.-El.: Epidemic Algorithms for Replicated Databases. IEEE Transactions on Knowledge and Data Engineering, Vol. 15. No. 5. (2003) 1218–1238
[36] Ladin, R., Liskov, B., Shrira, L., Ghemawat, S.: Providing High Availability Using Lazy Replication. ACM Transactions on Computer Systems, Vol. 10. No. 4. (1992) 360–391
[37] Schótt, T., Schintke, F., Reinefeld, A.: Efficient Synchronization of Replicated Data in Distributed Systems. International Conference on Computational Science ICCS 2003, Springer LNCS 2657, (2003) 274–283
[38] Petersen, K., Spreitzer, M.J., Terry, D.B., Theimer, M.M., Demers, A.J.: Flexible Update Propagation for Weakly Consistent Replication. 16th ACM Symposium on Operating Systems Principles, Saint-Malo, France (1997) 288–301