How to make Personal Smart Spaces Context-aware

Ioanna Roussaki[1], Nicolas Liampotis[1], Nikos Kalatzis[1], Korbinian Frank[2] and Patrick Hayden[3]

1 INTRODUCTION

Pervasive computing [1] or as otherwise called Ambient Intelligence [2] aims to assist users in their everyday tasks in a seamless unobtrusive manner. In this framework, there have been various research initiatives aiming towards the design and realization of smart spaces [3] in homes, offices, universities, schools, hospitals, hotels, museums, and other private or public places, where various automation facilities support the users. In these cases, research has focused on developing techniques to support building automation (such as intelligent light controls, window shutters, security systems, electrical appliances, door control, etc.), as well as mechanisms to adapt the behaviour of electronic devices (such as TV, radio and multimedia players), desktops and peripherals, etc. Nevertheless, these are fixed spaces that provide pervasive features and adapt to the user needs in a static and geographically limited environment.

However, mobile users require the same pervasive services wherever they are and whatever devices they carry along. Irrespective of the user's location, such a mobile pervasive system would be expected to provide access to devices and services in the user's current environment. Fixed smart spaces do not address the needs of mobile users, as outside their boundaries, only limited pervasive computing features are offered. For example, a Smart Home may control the devices within it and the services it offers to its residents, but it cannot easily share these with the mobile network of any visitor. To bridge this gap, the notion of self-improving Personal Smart Spaces has been introduced [4].

Personal Smart Spaces (PSSs) aim to couple the facilities offered by next generation mobile communications with the features provided by the static smart spaces to support a more ubiquitous and personalised smart space that is able to follow the user wherever he/she goes. As the owner of the PSS moves to different locations and places, his/her PSS interact with other mobile or fixed PSSs located in the owner's surrounding environment, aiming for a unique support for pervasive service provisioning. In a nutshell, a Personal Smart Space can be defined as a set of services within a dynamic space of connectable devices, where the set of services are owned, controlled, or administered by a single user. It facilitates interactions with other PSSs, it is self-improving and is capable of pro-active behaviour. Thus, a PSS is user centric and controlled by a single user, it is mobile (at least from user perspective), it allows for interactions with other PSSs and is capable of self-improvement and proactivity.

An inherent feature of PSSs is context awareness [5]. Context can be defined as basically all information that is relevant to a human-computer interaction [6]. However, even though context information holds out the prospect of enhancing user experience and increasing revenues; collecting it from various heterogeneous sources, disseminating it across distributed nodes, processing and managing it efficiently are not straightforward. Other context related features that need to be supported are: real-time control and management of the context sources; distribution of context information over heterogeneous networks and devices; inference of future or currently unavailable high level context information from raw context data based on various learning techniques; modelling, management, storage and processing of history of context data; support for privacy-aware access control facilities and secure distributed context event management mechanisms; etc. This paper elaborates on the mechanisms that have been designed in order to address the advanced requirements of PSSs regarding the establishment of a robust distributed context management framework.

More specifically, this paper is structured as follows. In Section 2, the context-awareness requirements that need to be addressed in PSSs are presented. In Section 3, an illustrative use case scenario is described, where the context-awareness aspects of PSSs are highlighted. Then, in Section 4, the context model that has been designed to represent the context information necessary to adapt the services provided by PSSs is briefly presented. Section 5 elaborates on the context management architecture that has been built to address the needs of PSSs. In this respect, the functional components of the context management system are presented, while emphasis is given to the mechanisms that support distributed context management facilities. Finally, in Section 6, the paper conclusions are drawn.

2 CONTEXT-AWARENESS REQUIREMENTS IN PERSONAL SMART SPACES

As already stated, a Personal Smart Space is a set of services that are owned, controlled, or administered by a single user, that are located within a dynamic space of connectable devices and that are capable of self-improvement and of demonstrating pro-active behaviour. This leads to numerous context-awareness requirements that need to be addressed in PSSs. A summary of these context-related requirements is provided hereafter.

- *Efficient Context Modelling and Semantics*, in order to represent the entire set of context data (both dynamic such as location, and static such as preferences) that need to be monitored, collected, stored and utilised in PSSs, along with the necessary metadata. More details on this are provided in Section 4.

[1] School of Electrical and Computer Engineering, National Technical University of Athens, Greece. Email: {`nanario`, `nliam`, `nikosk`}`@telecom.ntua.gr`.

[2] German Aerospace Center (DLR), Institute of Communications and Navigation, Germany. Email: `korbinian.frank@dlr.de`.

[3] Telecommunications Software & Systems Group (TSSG), Waterford Institute of Technology, Ireland. Email: `phayden@tssg.org`.

- Distributed Context Management, including distributed context maintenance, access, update and synchronisation, as well as provision of a transparent interface to context management, support of ad-hoc context exchange, real-time and non-real-time context handling. More details on this are provided in Sections 5.2.4 and 5.3.
- *Context Query Support*. Various context queries need to be supported by PSSs, such as identity-based (or navigational) queries, location-based, semantic-based, and time-based. More details on this are provided in Section 5.2.1.
- *Context Source Management*, including sensor data aggregation, context-source discovery, (de-) registration, configuration, etc. More details on this are provided in Section 5.2.5.
- *Context Source Management*, including sensor data aggregation, context-source discovery, (de-) registration, configuration, etc. More details on this are provided in Section 5.2.5.
- *Preference handling facilities*, in addition to the other context management facilities aforementioned, i.e. preference analysis, preference evaluation, and preference condition monitoring. More details on this aspect are provided in [7].
- *History of Context Modelling and Management* in order to support history-based context inference and access to past context information. In this respect, the user behaviour & status will also be modelled and recorded. More details on this aspect are provided in Section 5.2.3.
- *Context Event Management*, including support for distributed context event creation and propagation.
- *Context Inference*, i.e. extraction of high level context information from raw context data. In this respect, learning of context inference rules (CIR) needs to be supported, as well as preference learning algorithms, CIR individualisation, context association & pattern extraction/matching, and CIR learning from group knowledge. More details on this are provided in Section 5.2.2 and [8].
- *Group context*, i.e. context information of group of persons/users, including group preferences. More specifically, the following need to be addressed: efficient group context modelling and representation, group context management & maintenance, group context estimation & inference, context prioritisation & assessment for resource sharing and context/preference conflict resolution. More details on this are provided in [4].
- *Context privacy & security*. In this respect, the following need to be supported: access control over individual and group context; context integrity, reliability, confidentiality and availability; context-based access control; privacy policy learning; etc. More details on this are provided in [4].
- *Quality of context* modelling, management and exploitation, including soft context and uncertainty in context values and context inference rules.
- *Context sensitivity*, i.e. ability to support adaptation of the provided services to the context of their users.

In the scenario presented in the following section, several of the requirements above are addressed by the Personal Smart Spaces involved.

3 AN EXAMPLE SCENARIO

In this section we will present a scenario that demonstrates the possibilities of Personal Smart Spaces and explains the involvement of context in it. We assume that all actors carry smart phones with them that constitute their PSSs and coordinate communication with all devices in and outside this PSS. Also infrastructure provides services and context information via fixed smart spaces that interact with the actors' PSSs.

Tom, Steve, Susanna and a couple of other researchers have arranged to meet for an international project meeting. Tom is the host, while the other participants fly in and stay in a hotel that Tom has organized close to the meeting venue.

3.1 Scenario description

Steve and Susanna arrive in the airport at the same time. Their buddy finder application informs them, so they can meet and share the trip to the meeting. Right after landing, keeping track of her agenda, Susanna's PSS had identified that they are late and proactively cancelled the reservation at the car rental service and ordered a taxi instead. She offers to Steve to share the taxi.

The taxi's PSS is responsible for adjusting the seats, so that Steve and Susanna are as comfortable as possible given their height and size. The taxi driver is already informed about the meeting venue, the price is agreed by the PSSs and they arrive on time.

When the first guests are about to arrive, Tom is notified at his office PC. He heads to the meeting room. As he enters the climate controls are set according to his preferences. As the weather is fine and warm, the windows are opened automatically to let in the fresh May morning air.

When Steve and Susanna enter the building, a shared display in the foyer welcomes them and informs them of the day ahead. Their PSSs had interacted with the display's PSS and selected the appropriate information for them. Furthermore on their smart phones they are offered the possibility to check the current activity in the meeting room and to check the availability of the remaining colleagues from the project.

While Steve usually wants to glance at the menu at the canteen and the weather report for the rest of the day, if he is in his office and therefore interacts with his own company's PSS, this information is not applied now when he is on a business trip.

Susanna and Steve are guided to the right meeting room. There they meet Tom and a new colleague, Joan. Joan talks to Susanna for a while and their PSSs interact – each infers that they are talking to each other and that they have not met before – at least their PSSs haven't! Susanna takes her SmartPhone from her pocket and sees the proactive prompt that just needs her confirmation to exchange electronic business cards with Joan.

The meeting starts and after some words of introduction by Tom, Steve is the first to present his work. He walks to the front of the room and given the agenda and behaviour, his PSS infers that he is about to start the activity "presenting" and starts interaction with the wall display's PSS. He is shown a context-aware selection of suitable documents – related by last-use and association with the calendar entry – and hits the icon for the slides he prepared. The presentation is started and also logged in the automated meeting minutes that tracks the presentations shown and the speakers. An incoming phone call from his Yoga friend Katie is routed to his voice mail as he is busy presenting and the call urgency was indicated by Katie to be rather low.

Susanna is impressed by Steve's research results. She accesses the presentation from the common automated meeting minutes and wants to print it. Her PSS locates the closest printer, sends the print job, and guides her there automatically.

When the meeting approaches its end for the day, the PSSs interact to select a restaurant for dinner, taking into account the location, the weather and of course the participants' preferences regarding food and dinner time.

Having agreed on a restaurant in the city centre, Steve decides to drop his baggage at the hotel first. Therefore he does not join his colleagues who go the centre immediately, but decides to follow them later by public transport.

Steve's PSS selects the most convenient train to the city centre for him and interacts with the transport management system to acquire a ticket. Finally in the train he falls asleep, as he is tired after the hard working day, but his context aware alarm wakes him up just in time before he reaches the station where he has to get off.

Having enjoyed a wonderful dinner with his colleagues, they head together to their hotel in Joan's rental car. The car's navigation system guides them automatically to their hotel taking into account the broadcasted traffic situation.

3.2 Scenario analysis

This scenario shows a wide range of applications for PSSs and context. Context is used:

(1) in the process of service selection
(2) for service configuration
(3) during service execution
(4) to determine current preferences
(5) to proactively start services

Many different applications and basic services are used by Steve and his colleagues in this scenario, though only some crucial ones can be discussed in detail:

Firstly the car adaptation system, provided by the car PSS is of interest. It starts proactively (5) and incorporates the persons' preferences (4) and context during its execution (3). The relevant context information is height, size and weight of the passengers. Obviously, also the precise location of the passenger is needed, even within the car, and finally the user intent to enter the car resulting from location, movement and future calendar entries is responsible to start the service.

The room adaptation system (3, 4, 5) (as well as the restaurant finder service) furthermore includes weather and temperature information next to personal preferences.

In the call redirection application (2, 4, 5) mainly activity is the relevant context information. It depends on preferences, time, calendar entries, location, movements/status (like standing, walking, sitting and others) and the available as well as the currently used services.

The printing sub-scenario illustrates among others context aware service selection (1), based on features of the available printing services, but also on your current indoor position and proximity (including walking restrictions like walls etc.)

Finally the context aware navigation service or the context aware alarm (3) have to be aware of the current time, the user's target coming calendar entries as well as the map of the relevant area. Monitoring the current activity (walking-direction, but also sleeping) and precise location it calculates the best route to the target respectively initiates the alarm.

4 CONTEXT MODELLING

In order to efficiently represent context information in a PSS environment, the Persist Context Model (PCM) has been developed. The PCM includes all the classes that model the context information to be retrieved, exchanged, maintained and managed in general in the PSS. The scope of the proposed context model is to represent necessary information in an appropriate and uniform way for all the functional components of the PSS framework. The basic informational concepts used by the context model are the data classes: `CtxEntity`, `CtxAttribute`, and `CtxAssociation.` In addition to these core classes, there is the `CtxIdentifier` class that mainly addresses internal context management requirements, as well as the `CtxQualityAttribute` class, which further augments the model with Quality of Context elements.

The core concept upon which the context model is built is the *Entity*. An *Entity* corresponds to an object of the physical or conceptual world. For example an *Entity* could be a person, a device, or a service. The *Attribute* class is used in order to describe an *Entity*'s properties. To this end, many *Attribute* classes might be assigned to an *Entity*. Concepts such as the name, the age, and the location of a person are described by different attributes. Similarly, attributes describing a device's properties might be the identity, the voltage, and the operational status of the device. In a nutshell, the `CtxAttribute` class identifies an entity's status in terms of its static and dynamic properties and therefore, it captures all context information items that are used to characterise the situation of the owner entity. The `CtxQualityAttribute` class provides Quality of Context meta-data to Attributes [9]. Thus, each Attribute may be accompanied by an instance of the `CtxQualityAttribute` class. Examples of the Quality of Context properties provided by this class are: probability of correctness, frequency, precision, price, timeliness, etc.

Relations that may exist among different entities are described by the `Associaton` class. We identify two ways of associating entities. A peer relation among entities is described by the concept of the *undirected association*, while a non-peer relation among entities is described by the concept of *directed association*. In the latter, only one of the associated entities can be the originating point and is then called parent entity, while many entities could be the target points and are then called child entities. On the other hand, the undirected association represents relations among peer entities, where there is no owner or parent entity. Example types of directed associations are: "*owns*", "*uses*", "*locatedIn*", while types of non-directed associations are: "*friends*", "*schoolmates*", "*fellow passengers*", "*colleagues*", etc.

In a nutshell, the proposed context model is built to describe the situation of "*a person owning a device*" as follows: "The `EntityA` of type *`Person`* carries an `Association` of type *`Owns`* that connects it with `EntityB` of type *`Device`*. For this directed association, `EntityA` is the parent entity, while `EntityB` is the child entity.

The aforementioned `CtxIdentifier` class contains all the information that is necessary to uniquely identify the context information items and it is assigned to all entity, attribute and association instances. The string representation of the `CtxIdentifier` can be used by any PSS enabled context management system in order to retrieve the identified context information. The format of this identifier is as follows:

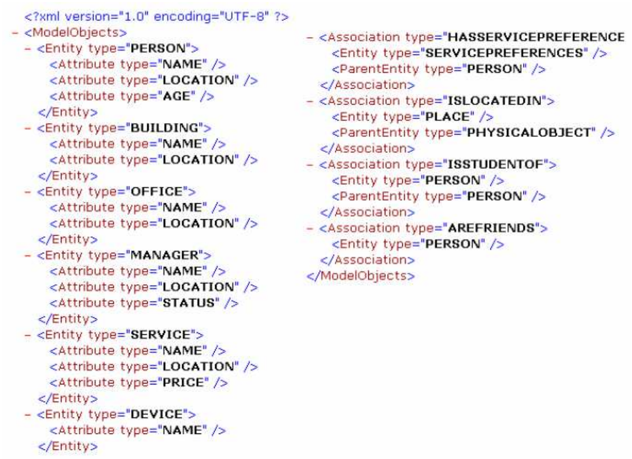
`CtxId: PSSid/DevID/ModType/Type/Number`

The individual parameters involved are described hereafter:

`Pssid`:	A unique identifier of the PSS where context information was first stored.
`DevID`:	An identifier of the device where the respective context information was initially sensed/collected and stored. This is the home device ID, the role of which is described in Section 5.3.
`ModType`:	Describes the type of the context model object, i.e. is one of the following: Entity, Attribute, or Association.
`Type`:	A semantic tag that characterizes the context model object.
`Number`:	A unique number within a single PSS.

As already described, a PSS may include various devices, on each of which a context management system resides. In a system such as this, the wealth and heterogeneity of context data strongly discourages the adoption of a centralised or completely distributed context data management scheme. For this reason and as it will be further described in Section 5.3, we have introduced a *"Hybrid distributed-centralised context management"* approach for managing and storing context data. In order to achieve this, we use specific flags indicating whether changes on a context model object should be forwarded to other context management systems of a PSS or not.

To address the PSS requirements regarding context semantics, a semantic taxonomy has been introduced that includes the various context types as tags and dictates how these various context tags can be combined. A segment of this taxonomy is depicted in Figure 1.



```
<?xml version="1.0" encoding="UTF-8" ?>
- <ModelObjects>
  - <Entity type="PERSON">
      <Attribute type="NAME" />
      <Attribute type="LOCATION" />
      <Attribute type="AGE" />
    </Entity>
  - <Entity type="BUILDING">
      <Attribute type="NAME" />
      <Attribute type="LOCATION" />
    </Entity>
  - <Entity type="OFFICE">
      <Attribute type="NAME" />
      <Attribute type="LOCATION" />
    </Entity>
  - <Entity type="MANAGER">
      <Attribute type="NAME" />
      <Attribute type="LOCATION" />
      <Attribute type="STATUS" />
    </Entity>
  - <Entity type="SERVICE">
      <Attribute type="NAME" />
      <Attribute type="LOCATION" />
      <Attribute type="PRICE" />
    </Entity>
  - <Entity type="DEVICE">
      <Attribute type="NAME" />
    </Entity>
  - <Association type="HASSERVICEPREFERENCE
      <Entity type="SERVICEPREFERENCES" />
      <ParentEntity type="PERSON" />
    </Association>
  - <Association type="ISLOCATEDIN">
      <Entity type="PLACE" />
      <ParentEntity type="PHYSICALOBJECT" />
    </Association>
  - <Association type="ISSTUDENTOF">
      <Entity type="PERSON" />
      <ParentEntity type="PERSON" />
    </Association>
  - <Association type="AREFRIENDS">
      <Entity type="PERSON" />
    </Association>
  </ModelObjects>
```

Figure 1. A segment of the context semantics taxonomy for Entities, Attributes and Associations

Finally, it should be mentioned that all context model objects are marked with a timestamp indicating their most recent update time.

5 CONTEXT MANAGEMENT ARCHITECTURE

The efficient management of context information is central in pervasive computing. Here, we present the Context Management (CM) framework of Persist which was designed based on the context model presented in Section 4 in order to cover the context-awareness aspects of Personal Smart Spaces. More specifically, this section is structured as follows. Subsection 5.1, provides a high-level view of the CM architecture. Subsequently, in Subsection 5.2, the functional components comprising this architecture are described, while Subsection 5.3, presents our approach for scalable distribution of context information among PSS devices.

5.1 High-level architecture

As illustrated in Figure 1, the CM framework acts as an intermediate layer between PSS/3rd party context-aware services and the sources of context information. This figure also provides a high-level view of the CM architecture introducing functional components, as well as, their interdependencies.

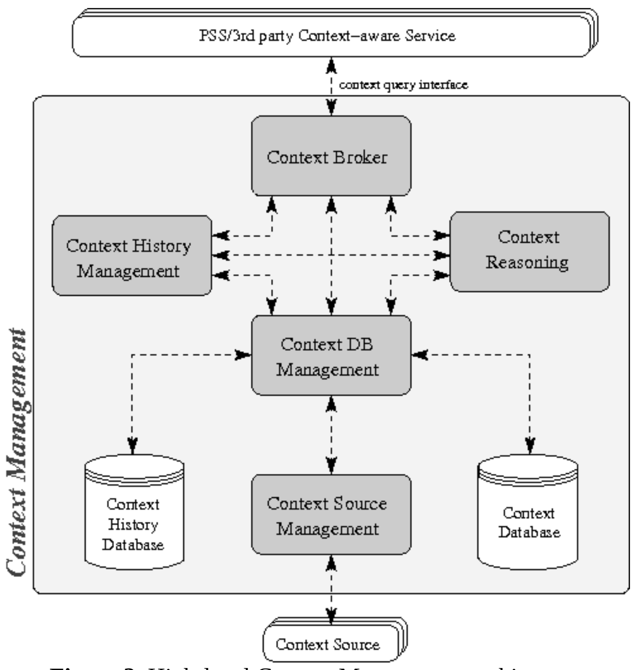


Figure 2. High-level Context Management architecture

An outline of the identified components follows, while a more detailed description is provided in Subsection 5.2:

- *Context Broker*: Provides context consumers with a query interface for retrieving, adding, removing, and updating context data.
- *Context Reasoning*: Uses probabilistic methods in order to derive high-level context information based on raw sensor data and/or context history.
- *Context History Management*: Collects, maintains and processes historic context data.

- *Context DB Management*: Translates context queries into standard SQL queries which are then executed in the underlying databases.
- *Context Source Management*: Manages context sources and collects their information.

Apart from the components above, the CM architecture comprises a *Database Management System* (DBMS) which enables access to the actual context repositories, i.e. the *Context Database* and the *Context History Database*. The former database is used for current context information, while the latter stores past data. Both database schemata conform to the context model described in Section 4.

Any off-the-shelf relational database meets the requirements of this model, however, our framework considers different DBMS solutions depending on device capabilities. In this context, a fully-featured DBMS, such as MySQL, can be deployed on a resource-rich PSS device, while a less-featured DBMS, like Apache Derby or SQLite for example, is more suitable for resource-constrained devices, thus, supporting current mobile smart-phone technology.

5.2 Functional components

The functionality of the components comprising the CM architecture is described in the subsections that follow.

5.2.1 Context Broker

The Context Broker manages access to context information. More specifically, it supports queries for retrieving, updating, adding or removing context data. Context retrieval queries in particular, can be performed both *synchronously* and *asynchronously*. For the latter case, context consumers are required to subscribe for context update notifications through the PSS Event Management component [4].

Context queries can be classified as follows:

- *ID-based*: As already described in Section 4, Entities, Attributes and Associations, which constitute the building blocks of context information, are uniquely identified by their Context Identifier (CID). Context consumers can specify CIDs in their context queries in order to identify the data items they are interested in.
- *Location-based*: This is actually a special use case for id-based queries and allows for discovering new context information based on location hierarchies modelled by Entities and their Associations. For example, given the CID of a place Entity, the “isLocatedIn” Association can be followed to discover the Entities located in that particular place.
- *Semantic-based*: When the CID is not known beforehand, context consumers can specify semantic criteria, i.e. tags, in their queries. The Context Broker uses these tags for best-match retrieval from a taxonomy of semantic tags with which context data items may be associated [10].
- *Temporal-based*: This type of query provides context consumers with access to past, as well as, future, i.e. predicted, values of context data. The actual processing of such data is managed by the Context History Management component which is described in Subsection 5.2.3.

Apart from context query handling, the Context Broker enables inter-communication among the CM subsystems of PSS devices.

To further elaborate the functionality of the Context Broker, we describe the processing of an ID-based context query. First, the Context Broker examines the PSS identifier which is encapsulated in the CID associated with the query in question. Based on this identifier, it can determine whether the relevant context data item is associated with a remote or the local PSS. For the former case, the original query is forwarded to the Context Broker of an available device of the identified PSS, while, for the latter case, the local context repository is checked. In case of an empty result from the local database, the Context Broker attempts to forward the context query to the *master* device of its PSS and, if that is not available, to the *home* one (see Subsection 5.3 for a definition of these terms). As a “last resort”, if the requested context information is not available, either locally, or remotely, the Context Broker can invoke the Context Reasoning component in order to infer this information.

5.2.2 Context Reasoning

Some pieces of context information are directly derived from sensor readings. Location, for instance, can be determined from a number of sensors, such as GPS receivers or RFID readers. However, other pieces of context information, like “user activity” or “busy status” for example, cannot be assessed in a straightforward way. This so called high-level context information can be derived from the Context Reasoning component which provides a probabilistic *context inference engine*. The functionality of this engine is twofold:

1. To extract context estimation rules, i.e. inference rules, based on context history.
2. To infer high-level context based on the relative inference rules and raw sensor data.

Given the high temporal constraints of context-aware services and the computational complexity which is inherent in most probabilistic methods, time efficiency is a key factor in context inference. The Context Reasoning component addresses this issue through the use of *bayeslets* [11], which allow for prompt delivery of otherwise unavailable context data, as well as, refinement of existing, yet inaccurate information.

5.2.3 Context History Management

The Context History Management component is responsible for the collection and processing of *History of Context* (HoC) [12]. Maintaining this data is of great importance to the CM framework of Persist as it supports:

1. Inference of *current* context information, which is no longer available, based on HoC.
2. Prediction of *future* context information based on periodic context data patterns extracted from HoC.

It should be pointed out that this component does not automatically monitor all types of context information for inclusion in the history database. Instead, the PSS owner has to explicitly register context types for HoC maintenance. Registration ensures that whenever these context types are updated in the (current) context database, the Context History Management component is notified and appropriately updates the context history database after processing and scoring the relative data items, evaluating the likelihood of their occurrence. Based on these updates, time-dependent attenuation can be applied to all past context values in order to extract context prediction rules.

5.2.4 Context DB Management

As already stated, the CM framework comprises two databases for storing context information, one for current values and the other for historic data. Hence the need for an intermediate layer between the context query interface provided by the Context Broker, and the underlying DBMS that actually controls the storage, management, and retrieval of data in these databases. The functionality of this layer is implemented by the Context DB Management component which is able to translate context queries submitted by PSS/3rd party context-aware services into standard SQL queries that can be executed in the framework's DBMS.

Another responsibility of the Context DB Management component is to provide the Context Source Manager with an interface to manage the QoC meta-data associated with context information, i.e. to assign and update the relative QoC attributes.

5.2.5 Context Source Management

The Context Source Manager component manages communication with diverse context sources. More specifically, it is responsible for discovering, (de-)registering, configuring and collecting context information from available sources. In the presence of multiple sources for the same piece of context information, this component is able to select the appropriate source based on the QoC requirements of context-aware services. Re-configuration of registered context sources may also be invoked based on such requirements. For example, if a context-aware service requires more frequent updates on location information, the Context Source Manager can dynamically re-adjust the sample rate of the location sensor. It should be noted that the overhead in resource consumption of both the sensor and the PSS device is taken into account for this process.

5.3 Distributed context management

This subsection deals with context management across the device nodes of personal smart spaces.More specifically, it describes a scalable scheme for distributing context data among multiple devices forming a single PSS. The following distribution approaches have been considered:

1. _Centralised context management_: The context database is hosted by one device per PSS. All context queries must be forwarded to that particular device for processing.
2. _Fully distributed context management_: Each device in a PSS hosts a copy of the context database. Context information is replicated in all devices, thus, context queries can be handled by any of them.
3. _Hybrid distributed-centralised context management_: Each device in a PSS hosts a context database; yet, context information is not replicated in all of them. There is, however, one device whose database contains every piece of context information and is able to handle all context queries. Regarding the other devices in the PSS, context queries cannot always he handled locally and must thus be forwarded to the appropriate device for further processing.

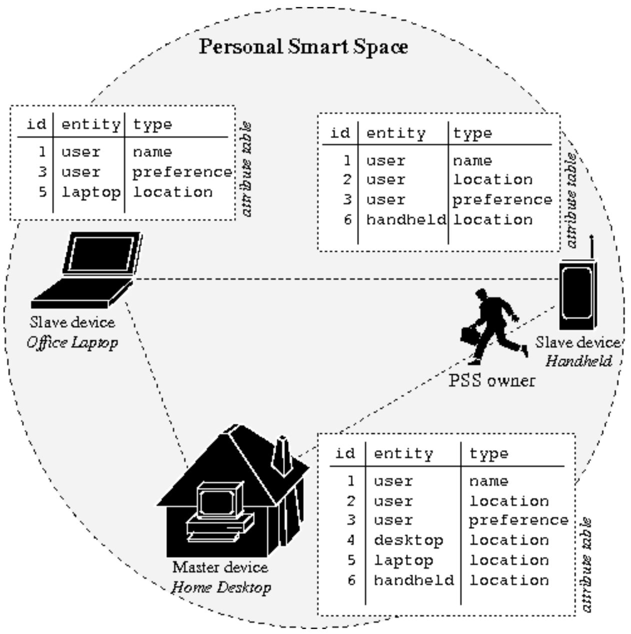
The first two approaches would be sufficient for fixed PSSs, however, scalability, in terms of processing, storage, remote communication and, power consumption, would be an issue for PSSs where both fixed and mobile node devices are involved. Therefore, we consider the hybrid distributed-centralised approach as more appropriate for Persist.

In the hybrid approach, all devices within a PSS contribute to collecting context information and are assigned different roles based on their capabilities. The following roles have been distinguished:

- _Master device_: A single device within a PSS, usually one with server capabilities, i.e. high processing power and storage capacity, is elected as the master device. It aggregates context information from all devices of the PSS and is responsible for maintaining HoC data in order to support inference of current and future context. Being the core of the CM framework, this device is intended to be always on.
- _Slave device_: Apart from the master device, every PSS may have one or more slave devices which contribute to the collection of contextual data. Slave devices periodically send collected data to the master one. If the master device becomes unavailable, a slave one may be elected in replacement.
- _Home device_: A device is considered home for a particular piece of context information if the latter is relevant to this device only.

For example, Attributes which are derived from context sources attached to a PSS device, have that particular device as their home one.

Nevertheless, it should be highlighted that static context information, as defined in 4, is fully distributed in the hybrid approach too.



id	entity	type
1	user	name
3	user	preference
5	laptop	location

id	entity	type
1	user	name
2	user	location
3	user	preference
6	handheld	location

id	entity	type
1	user	name
2	user	location
3	user	preference
4	desktop	location
5	laptop	location
6	handheld	location

Figure 3. Example context data distribution within a PSS

Replicating this type of context information in all devices of a PSS ensures that it will be available even when the master device is currently unreachable. Furthermore, any of the aforementioned device roles may additionally be flagged as active to indicate that this is the device the PSS owner is currently interacting with. As the active device is more likely to interact with other personal smart spaces, dynamic context information can be cached on that particular device in order to improve the responsiveness of the CM framework.

Figure 2, illustrates a rather simple example of context data distribution across the device nodes of a PSS under the hybrid distributed/centralised context management approach.

In this example, the user's desktop computer has been elected as the master device. As the user is currently away from both his home and the office, his handheld computer is considered the active device.

Thus, apart from his static context information (user preferences) and the device-specific data (handheld location), the CM of this device additionally stores dynamic context information (user location). The master device, however, aggregates context information from all device nodes within this PSS.

6 CONCLUSIONS

The scenarios discussed show how useful personalised services can be in a Personal Smart Space (PSS). The delivery of personalised services relies on the implementation of a context management subsystem so that services can utilise context data to interact with the user at the appropriate time and in the appropriate way, personalised to the user's requirements. The context management subsystem must gather and store this context data and must also maintain a context history in order to provide data to other components that provide learning and reasoning functionality.

A key research focus of the Persist project is the elimination of islands of pervasiveness where a user has no access to personalised context information while disconnected from a fixed smart space so it is necessary to store context information on each device in a user's PSS. Current device limitations prevent the replication of all context information across all devices and it was with this limitation in mind that a hybrid approach was chosen in which each device stores context information that is relevant to that device as well as static context information, while a master device stores context information from all devices in addition to context history. The complete set of context information is available on the master device to support algorithms that perform learning and reasoning for the PSS and a limited set of context is available on slave devices to allow services to access context information even when the device is separated from the rest of the PSS. When context information is not directly available on the device or in other devices in the PSS, it can be inferred.

The combination of distributed context information with personalised context aware services allows the realisation of a truly useful personal smart space that accompanies a person as they move between smart locations.

ACKNOWLEDGEMENTS

The research leading to these results has received funding from the European Community's Seventh Framework Programme [FP7/2007-2013] under grant agreement n° 215098 of the Persist (*PERsonal Self-Improving SmarT spaces*) Collaborative Project.

REFERENCES

[1] M. Satyanarayanan, "Pervasive Computing: Vision and Challenges", IEEE Personal Communications, Vol. 8, No. 4, pp. 10-17 (2001).

[2] P. Remagnino, G.L. Foresti, "Ambient Intelligence: A New Multidisciplinary Paradigm", IEEE Transactions on Systems, Man and Cybernetics, Part A, Vol. 35, No. 1, pp. 1-6 (2005).

[3] R. Singh, P. Bhargava, S. Kain, "State of the art smart spaces: application models and software infrastructure", ACM Ubiquity, Vol. 7, No. 37, pp. 2-9 (2006).

[4] I. Roussaki et al., "Persist Deliverable D2.4: Initial architecture design", Technical report, PERSIST (FP7-ICT-2007-1) (2008).

[5] S. Loke, "Context-Aware Pervasive Systems: Architectures for a New Breed of Applications", Auerbach Publications, 1st edition (2006).

[6] A. Dey, "Understanding and Using Context", Personal and Ubiquitous Computing, Vol. 5, No. 1, pp. 4-7 (2001).

[7] S. McBurney, N. Taylor, H. Williams, "Giving the User Explicit Control over Implicit Personalisation", in Procs. of Workshop on Intelligent Pervasive Environments (under AISB'09), Edinburgh, Scotland (2009).

[8] K. Frank, P. Robertson, S. McBurney, N. Kalatzis, I. Roussaki, M. Marengo, "A Hybrid Preference Learning and Context Refinement Architecture", in Procs. of Workshop on Intelligent Pervasive Environments (under AISB'09), Edinburgh, Scotland (2009).

[9] T. Buchholz, A. Kupper, and M. Schiffers, "Quality of context: What is it and why we need it", in Procs. of Workshop of the HP OpenView University Association (HPOVUA'03), Geneva, Switzerland (2003).

[10] C. Pils, I. Roussaki, T. Pfeifer, N. Liampotis, N. Kalatzis, "Federation and Sharing in the Context Marketplace", in Procs. of Workshop on Location- and Context-Awareness (LoCA'07), Lecture Notes in Computer Science, Vol. 4718, pp. 121-138, Springer Berlin / Heidelberg (2007).

[11] K. Frank, M. Rockl, P. Robertson, 'The Bayeslet Concept for Modular Context Inference', in Procs. of 2nd IEEE International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies (UBICOMM'08), pp. 96-101, Washington, DC, USA (2008).

[12] N. Kalatzis, I Roussaki, N. Liampotis, M. Strimpakou, C. Pils, "User-centric inference based on history of context data in pervasive environments", in Procs. of 3rd ACM International Workshop on Services Integration in Pervasive Environments (SIPE'08), pp. 25-30, New York, USA (2008).