MPSG: A Generic Context Management Framework in Mobile Spaces

Penghe Chen, Shubhabrata Sen, Hung Keng Pung, Wai Choong Wong
National University of Singapore
21 Lower Kent Ridge Road
Singapore 119077
g0901858, g0701139, dcsphk, elewwcl@nus.edu.sg

ABSTRACT
Recent advances of body networks enable the acquisition of personal context from various kinds of body sensors. As a result, how to build a platform to properly manage personal context becomes a problem. In order to address this issue, we design and implement a mobile physical space gateway (MPSG) which is a generic context management framework in mobile spaces. This framework includes three layers – service management, data management and network communication. A system toolkit has been developed and experimental tests have been conducted to examine its performance.

Keywords
Framework, Context Data Management, Mobile Space, Mobile Device, Service Management.

1. INTRODUCTION
Context is usually defined as information that can be used to characterize a user's situation.[2]. The recent advances in the field of body network enables various data can be monitored and used as context data. Additionally, developments of mobile technologies have made mobile devices as an integral of people's lives. As a result, how to utilize mobile device to manage and utilize personal context becomes a challenging issue.

Different frameworks have been developed to manage various data of people. Aware[1] is an Android based framework that focuses on instrumenting, inferring, logging and sharing mobile context information. Contory[8] is a middleware specifically designed for context provisioning on mobile devices. MDi [6] designed as part of the proposed agent-based framework for ambient intelligence utilizes a master agent DB module to store user data. A mobile framework is illustrated in [5] that handles context acquisition, unifying, modeling and management for mobile devices. MobiCon [4] is a mobile context-monitoring framework that aims to support context aware applications by addressing issues of resource limitations in sensor-rich mobile environments. Orchestrator [3] utilizes an active resource usage orchestration method to manage system resources to support concurrent context aware services.

However, most of these frameworks do not provide a proper integration of local personal context data with external context entities. Also, these frameworks do not provide any concrete solution for the problem of generating high level context data through the process of reasoning.

In this paper, we propose and design a mobile physical space gateway to represent and manage context data in a mobile context space as an extension to our previous work Coalition [7]. Physical space gateway (PSG) is a software module managing the context data from the context sources inside a physical space [7]. The contributions of this gateway can be outlined as follows. Firstly, it provides a way to manage the various types of services. Also, it defines a way to model and manage the context data of a mobile entity. Finally, it defines a way to manage network communications with other components and the network connection controls.

An illustration of the framework architectural design is shown in Figure 1 and we classify those modules into three categories based on their functionalities: Service Manager, Context Data Manager and Network Communication Manager. Details of these categories are illustrated in following sections.

The rest of the paper is organized as follows. Section 2, 3 and 4 illustrates service management, context data management and network communication management respectively. Performance evaluation is demonstrated in section 5 and the whole paper is concluded in Section 6.

2. SERVICE MANAGEMENT
Service Manager manages the various services in each mobile device. These services can include system services, such as context data services among others. Furthermore, inter-relationships between different services are also handled by this manager to improve reusability and cooperation.

2.1 Context Data Services
Context data services handle the tasks of context management and provision, which are designed for the owner/administrator to acquire and modify context data. Some of the context data services are as follows:

Context Querying Service: This service is used to access context information through SQL-based queries. It can provide context information to the requester accordingly by communicating with the modules in the underlying context data manager.

Context Attribute Adding/Removing Service: The MPSG uses an attribute-value based context data model. This service also lets the user add/remove the local context database with attributes.

Local Context Lookup Service: This MPSG framework is designed as a platform of managing context data for the mobile space. This context data can either exist locally or be sourced from other context sources. This service handles the queries concerning local context data only.

BODYNETS 2013, September 30-October 02, Boston, United States

DOI 10.4108/icst.bodynets.2013.253579

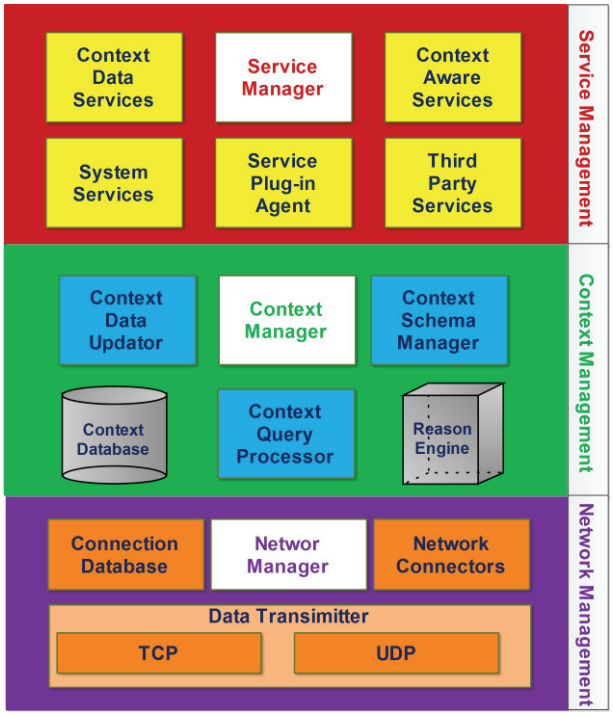

Figure 1Framework Architecture of MPSG

Context Data Updating Service: This service is intended to help users manually update the user defined context information, such as user preference and profile.

2.2 System Services

Another group of services is system services which realize functionalities that are related to the Coalition middleware such as joining/leaving, mobility management and other context related services. Some of the important system services are as follows:

MPSG De/Registration Service: This service helps the user de/register the MPSG with Coalition. Registration enables personal context can be accessed by others through Coalition, while deregistration remove person appearance from the Coalition system.

Mobility Updating Service: MPSGs may experience network information changes or losses caused by movement. In order to solve this problem, this service lets MPSGs update their new network information and trigger the callback lookup operation in Coalition.

Callback Issuing Service: In order to resume the communication caused by movements, this service lets callers issue a special request to Coalition so that notification will be given when the specific MPSG reappears later.

2.3 Context Aware Services

Another important group of services are the context aware services that utilize the context information provided by the underlying data management modules to realize context awareness. These services can be orchestrated to build the context aware application. There are a number of context aware services that are used frequently such as *finding friends nearby to you*, and *finding restaurants around you.* We create a repository of these commonly used services which allows developers to design applications without worrying about the underlying data management operations .The services stored in this repository can be viewed as plugins and are made available to developers using an agent module. This agent manages the local plugins and also imports new plugins from the repository as and when dictated by the user's context. This on-demand approach for installing the plugins leads to better resource utilization.

2.4 Service Invocation

Different service invocation methods have been designed for services. By considering the nature of each group of services, they have different preferences in selecting the methods. Context aware services that mostly utilize the underlying context data management modules are invoked directly using the relevant API. On the other hand, system services pertaining to Coalition are invoked through the RPC method. Finally, independently developed third party services can be discovered and invoked using the LASOD platform [9].

3. CONTEXT DATA MANAGEMENT

Context Data Manager handles the task of modeling the various context data in each mobile space and how to manage those context data. Additionally, it also takes care of the processing of context data to derive high level context information. It can also provide context information through a SQL-based context query language.

3.1 Context Data Modeling

The main aim of context modeling is to represent context data using different technologies. By considering the simplicity and extendibility, this MPSG framework utilizes the attribute value pair based method to represent and model context data. Each attribute represents a specific piece of context, such as location, activity, etc. A real time value will be updated to each attribute based on which context aware services can derive situations of users. The context schema is another important concept defined in MSPG and consists of the entire list of context attributes of each MPSG. Each MPSG submits its context schema to Coalition during registration. Accordingly, Coalition chooses the context space and semantic cluster for the MPSG. A list of neighbors will be replied to the MPSG. MPSG keeps all the context values and only submits its attribute list to Coalition during registration. As a result users have full control over the context data which makes security and privacy management easier.

3.2 Context Query Language

The *Context Query Language (CQL)* defines the syntax to formulate a request to query the context information. This MPSG framework uses an SQL-based context query language to represent context queries. The basic structure of context query is as follows:

MODE LOCAL | GLOBAL
SELECT <context information>, […]
FROM <context domain>, […]
WHERE <predicate>, […]
[ON INTERVAL <value> LIFETIME <value>]

The MODE clause defines the type of query, and LOCAL and GLOBAL indicate the local and global context data respectively. The SELECT clause indicates what type of context information is needed. The information can either be a context attribute or processed context attribute information applied certain operations. The FROM clause indicates the corresponding context domain from where the information is requested. The WHERE clause

specifies the constraints to be placed while retrieving the context information. The ON clause is optional and specifies LIFETIME and INTERVAL for continuous context retrieval.

3.3 Context Query Processing

As shown in the proposed CQL, MPSG can handle two types of queries: local queries issued by native applications which concern local context data and global queries issued by other MSPG/PSGs of Coalition which concern context data over all the possible context sources.

The process of handling global queries includes three parts. First, the query is circulated in the P2P network of the semantic cluster [7] by forwarding the queries to its neighbors. Second, the query constraints are checked with its context data to see if the query requirements are met. Third, the context data is reported to the query processor if it fulfills the requirements.

The process of handling local context queries is relatively simple. Whenever a query comes in, it is parsed and the relevant context data is reported to the application.

3.4 Context Data Updating

Context data updating is important for context data management because a stale context value may easily cause mistakes or errors to the applications. In order to let the MPSG obtain the latest data value, two different ways have been designed for this MPSG framework to update context data.

One is to update context data in an automatic manner. Context data is obtained directly from sensors and third part services, such as acceleration, GPS coordinate, weather conditions, and semantic location names and so on, changes frequently and need be updated frequently.

The other is to update context data manually. There are some kinds of context that cannot be obtained through sensors or third part services but have to be retrieved directly from users, such as profile, preference, hobbies and so on. Such kinds of context data have to be updated relying on the users. In the design, MPSG provides an interface over which users can retrieve and check the fresh value of each registered context attribute, and then users can manually update context data by inputting a new value.

3.5 Context Reasoning

Context is usually divided into two categories: primary context data representing raw context data and derived context information which are information generated by applying certain functions on primary context data. Higher level context information is more meaningful to users and how to reason the derived context information is important for context data management. In order to improve the framework intelligence, MPSG is designed with a local reasoning engine that can derive higher context information based on local context data information.

4. NETWORK COMMUNICATION MANAGEMENT

This manager takes charge of managing the network connections of MPSG. Since Coalition leverages on P2P networks to connect the MPSGs belonging to the same context domain together, an MPSG needs to maintain P2P connections to its neighbors belonging to the same semantic cluster. Additionally, as each MPSG has different context attributes corresponding to different semantic clusters, each MPSG will have a different set of neighbors per semantic cluster. Furthermore, MPSGs also need to connect to other components of Coalition such as context space gateway, semantic cluster and query processors. All of these connections must be properly managed. By designing this module that specifically handles the network communications between the MPSG with other components, the upper layer service manager and context data manager are completely decoupled from the underlying network communication details. The communication manager is designed to support network communication using both the TCP and UDP protocols. This network communication manager comprises of three sub-components: connection manager, network connector and connection database.

4.1 Connection Manager

This component provides a list of API functions that receive upper layer service instructions like registration/deregistration, context query. It checks the instruction details and identifies the corresponding components in the other end and gets the network information from the *connection database*. Additionally, it also identifies and invokes the corresponding *network connector* that should be used according the nature of function.

4.2 Network Connector

Coalition leverages on JAVA reflection to realize RPC between different components so that network level function callings can be realized. As different functions/services will have different parameters each function needs an individual function to do the remote function call. We call such individual functions as connectors. Some of the connectors are:

registerMPSG: takes care of joining the MPSG to Coalition by connecting the MPSG to the corresponding semantic clusters.

removeMPSG: takes care of disjoining MPSGs from Coalition system by disconnecting the MPSG from all related semantic clusters.

registerNeighbor: builds connections with neighbors in the P2P network by passing its own network information.

queryMCS: issues queries to the Coalition server in order to retrieve context information from various context sources.

queryP2P: forwards incoming queries to its neighbors so that the query can be flooded to all MPSGs in the same P2P network.

4.3 Connection Database

The MPSG does not keep persistent network connections with other components but maintains the network information of other components and initiates connections on demand at runtime. The connection database records the network information of these components which may be contacted by the MPSG later.

On one hand, MPSGs need to communicate with the Coalition server as well as context spaces and semantic clusters to join and leave the system. On the other hand, MPSGs need to communicate with the query processor as well as their P2P neighbors to route context queries and collect context data. Network information of these components is stored in the database and connections can be established with them whenever necessary.

5. PERFORMANCE EVALUATION

We have developed an Android version of MPSG framework based on above mentioned concepts and ideas and tested with Coalition to examine the performance. We first examine the

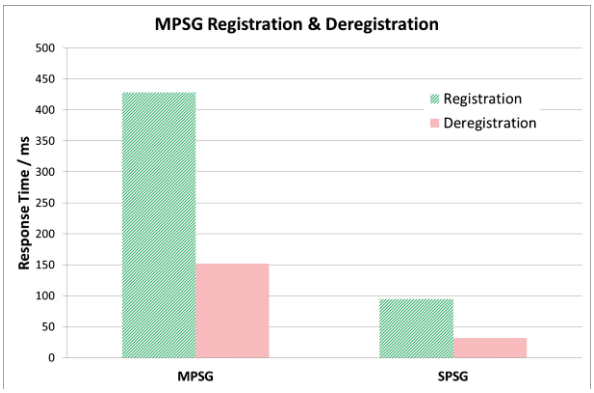

Figure 2 MPSG registration and deregistration

response time of MPSG registration and deregistration services. The results are illustrated in Figure 2. From the diagram, we can see that, compared to SPSG, MPSG needs more time to join and leave Coalition system. This is expected as SPSGs are normal PCs and connect to Coalition server through a wired network, while MPSGs are mobile devices and connect to Coalition server with a wireless network usually with more overheads.

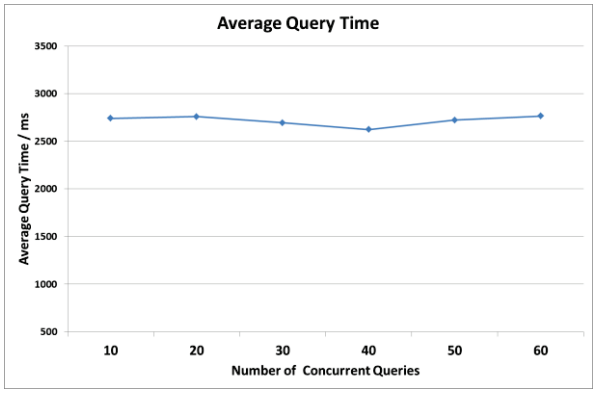

Figure 3 Average Query Time

We next examine the average response time of context queries in MPSG. In our experiments, we test the performance by issuing a number of queries simultaneously to MPSGs and other SPSGs. The results are shown in Figure 3. The chart indicates that the increase in the average response time is quite smooth with the increase in the number of concurrent queries. This indicates that our MPSG framework can support context query processing in a scalable manner.

Table 1. Query Time Breakdown (/ms)

	Global Query	Local Query
Pre-Process	1006	8
Process	631	5
Post-Process	651	12
Total	2288	25

We also examine the time breakdown of the query response time. The query pre-processing event refers to the operations of query parsing, query analyzing and query flooding to corresponding MPSGs. The query process refers to each MPSG processing the query and reporting it to the query processor. The context post-processing operation refers to the task of consolidating the query replies. From table 1, we can observe that the response time is evenly distributed and pre-processing takes more time in global queries, but post-processing takes more time in local queries. This is expected as the global queries need to distribute queries to the MPSG network in the pre-processing part, while local queries do not need this.

6. CONCLUSION

We have presented the framework of a MPSG in this paper. This framework includes three layers. The service management layer includes different types of services. Applications can utilize these services to realize context awareness. The data management layer models and manages the context data of the mobile space. As a result, context information can be obtained through SQL based queries. The network communication layers takes charge of various network communication for the framework. A system toolkit has been developed and experiments have been conducted to examine the performance.

7. ACKNOWLEDGMENTS

This research was carried out at the SeSaMe Centre. It is supported by the Singapore NRF under its IRC@SG Funding Initiative and administered by the IDMPO

8. REFERENCES

[1] http://www.awareframework.com/home/

[2] Dey, A. K., Abowd, G. D., & Salber, D. (2001). *A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications*. Human-Computer Interaction, 16(2-4), 97-166.

[3] Kang, S., Lee, Y., Min, C., Ju, Y., Park, T., Lee, J.,. Song, J. (2010). *Orchestrator: An active resource orchestration framework for mobile context monitoring in sensor-rich mobile environments.* 2010 IEEE International Conference on Pervasive Computing and Communications (PerCom),.

[4] Lee, Y., Iyengar, S., Min, C., Ju, Y., Kang, S., Park, T., Song, J. (2012). *Mobicon: a mobile context-monitoring platform*. Communications of the ACM, 55(3), 54-65.

[5] Martín, D., Lamsfus, C., & Alzua, A. (2011). *Mobile context data management framework*. 5th FTRA International Conference on Multimedia and Ubiquitous Engineering (MUE), 2011.

[6] Nulu, S. (2009). *A Generic Mobile Agent Framework for Ambient Intelligence*. 2008 ACM symposium on Applied computing SAC '08

[7] Pung, H. K., Gu, T., Xue, W., Palmes, P. P., Zhu, J., Ng, W. L., Chung, N. H. (2009). *Context-aware middleware for pervasive elderly homecare*. IEEE Journal on Selected Areas in Communications, 27(4), 510-524.

[8] Riva, O., & di Flora, C. (2006). *Contory: A smart phone middleware supporting multiple context provisioning strategies*. 26th IEEE International Conference on Distributed Computing Systems Workshops, ICDCS Workshops 2006.

[9] Zhu, J., Oliya, M., Pung, H. K., & Wong, W. C. (2010). *LASPD: A Framework for Location-Aware Service Provision and Discovery in Mobile Environments.* IEEE Asia-Pacific Services Computing Conference (APSCC), 2010.