Sindh Univ. Res. Jour. (Sci. Ser.) Vol. (45): 01-06 (2013)

SINDH UNIVERSITY RESEARCH JOURNAL (SCIENCE SERIES)

Architecture for Sensing Activity Context using Software Sensors in the Context-Aware Service-Based Environments

K.T. PATHAN++, N. CHANNA**, N.H. ARIJO*, S. MEMON***

Department of Computer Science, University of Leicester, Leicester, United Kingdom

Received 15th August 2012 and Revised 13th January 2013

Abstract: Expectations of the users from context-aware systems is to help them in their dynamic situations. Systems follow the context of the user either by hardware or software they use in their surroundings directly or indirectly. Hardware devices are already at common when activity context is the major concern. However, after the implementation of 'software as a service' the software can also be utilised; whenever we are dealing with software and exchange the data through SOAP messages (the less explored part of the technology which has a lot to offer). This paper presents a framework for the possibilities to exploit that data with number of ways to sense all the flavours of the context with an emphasize on activity context. If the activity context is gathered and utilised in a true sense, the promise of context-aware systems would be near to promising. This further states all the steps of the architecture starting from capturing data to reasoning and querying results of the structured knowledge in the web services environment for a scenario.

Keywords: Context-aware Systems, SOA, Activity Sensing, Software Sensors.

1. INTRODUCTION

Context in the physical world, which involves a number of important concerns related to the combination of data provided by sensors to the context-aware computing, that includes: what to sense for the particular type of context, how to acquire information needed and how to apply reasoning with that information provided to infer the context of a user. It is highly needed that programs and services should respond according to user's situational needs and behave the way they want these to be, i.e. dynamically. To know the context the information which we require can be captured through number of ways for example by user information, network (location, time, nearby objects), sensors (activity) and other sources.

In late 1990s some location-aware systems (Abowd, *et al.,* 1997) Baldauf, *et al.,* 2007 and Cheverst *et al.,* 2000) were made and still the most frequently concerned type of context is the location. Many of the context-aware systems are still far away to sense the user's activity (i.e. what user is doing?). Use of hardware sensors seem to be used mostly for activity context where computer vision techniques are used for recognition, reconstruction and reorganizations of the data taken from tracking devices.

The logical sensors (Watson Project Budzik, *et al.,* 2000) and the IntelliZap Project (Finkelstein, *et al.,* 2001): however provide related information by reading user's information of opened web pages and other documents. Gmail a service from Google (most popular search engine and giant of advertising industry) analyzes user's interactions with the services they offer for example if a user opens up an email which contains text of restaurants then it advertises the restaurants' link on that page, unaware of the fact that user is opening only the already advertised deal hence no concern with the user's context. Other reveal status of the online user (Truong, *et al.,* 2008) which might not be true at all times; if user forgets to update it, therefore lacks to capture the activity.

Hardware sensors may be more expensive and will be time taking to implement as a full to sense the activity. Service oriented architecture gives an idea of "software as a service" which is developed and updated at the single site and users are using these services to accomplish their day to day tasks, these software can become smart and used in addition to the hardware or on their own to sense the context, when employed properly.

To emphasis more on software sensors during the use of web services, the challenge is to what and from where the data to capture that reflects the user's context? We see different exchanges between user and services provide us with SOAP (Gudgin, *et al.,* 2004) messages and therefore the raw data. This data can lead us to user's activity. The next challenge is to make it structured and transform it into a knowledge which is required for the inference mechanism. Reasoning can not be done logically, if data is unstructured, therefore the utter need arises to structure the data and apply the reasoning onto it. Once knowledge is there the requirement might be to infer some more facts by applying rules onto the existing facts.

++Corresponding author K.T PATHAN email: ktp2@le.ac.uk Cell. No. +92-3352171212

*Institute of Information & Communication Technology, University of Sindh, Jamshoro.

**Institute of Business Administration, University of Sindh, Jamshoro.

***Centre for Electronic Systems Research, Brunel University London, United Kingdom.

In traditional web services SOAP messages carry data which is not processed but contains useful information regarding a user's interaction with that service and after processing can be useful in further ways. If a user is using a Service of Calendar and she is filling in the details (the data), for example she wants to add an event which requires details like Title of the Event, Time (From to To), Location, Description etc. While inputting these details user is actually passing her crucial data through SOAP messages and this raw data in SOAP messages carry information of the User's Activity but in a form which is not further processable by machine and hence become a source to update the Calendar Service only.

If system is based on software sensors structured knowledge is a need that is stored in a triple form so that automatic reasoning can be performed. For example if you have SOAP messages containing the event title, time and location- mentions the complete context, if the rules are applied properly. Weather conditions can further add semantics with that particular location to deduce with the reasoning rules. Category of Office Calendar and Work Calendar may strengthen the rules and hence sense the activity context with the combination of traditional services along with semantics.

To instantiate an Ontology after making one we are not only considering the methods which create the new file and not only inserting it without any logical way but we are also instantiating the existing knowledge and providing methodology which can further extend the Ontology.

This paper discusses all the steps starting from acquiring context to rules to infer to use software sensors for the use of recognizing activity context in the context-aware service-based environments. Next section presents the context-aware service-based architecture to support software sensors approach in the context-aware systems.

2. MATERIAL AND METHOD

The Framework

Context-Aware Service-Based Architecture For Sensing Activity Context Using Software Sensors

This paper presents the Architecture to sense the Activity Context of the user in the context-aware environment with the use of software sensors. Generic context model has strong need in the context-aware systems as well the architecture to understand the context in the situation of the user. This becomes even more crucial when considering input from the software sensors that senses out of web services with the abstraction of the context data (Hong *et al.,* 2004) A framework that supports the development of context-aware systems easier in the future to utilize the less explored side of sensors.

OASIS August 2012) defines service-oriented architecture as: "A paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains. It provides a uniform means to offer, discover, interact with and use capabilities to produce desired effects consistent with measurable preconditions and expectations". To design and develop 'software as a service', the service-oriented architecture (SOA) provides with the set of principles. SOA requires interoperability in between the different platforms using a communication protocol that depends on the messages. SOA has three main components. The first one is client (service requester), services itself, and service repositories. The communication between a service and a client is facilitated by the service repository. Services encapsulate the description of the software; Web Service Description Language (WSDL) (Christensen, *et al.,* 2001) is an XML format for describing services as a set of end points. To invoke the services, clients interact with the services in the form of messages that are travelled using SOAP (Mitra, *et al.,* 20007) protocol for exchanging structured information. This information is in the form of XML and carries the arguments client sends to perform its task on web services. This is where we can sense the user's context in the form of raw data only. Following sections will unleash to make this data usable in our proposed system.

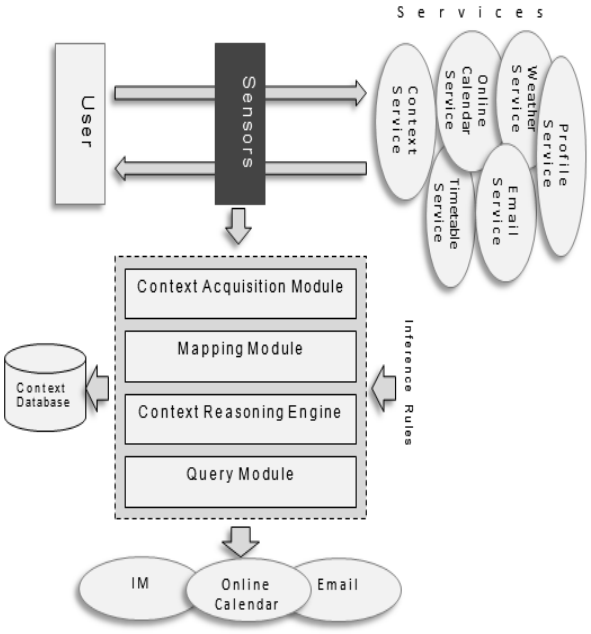

Fig. 1: Context-aware Service-based Architecture for Sensing Activity Context using Software Sensors

For a developer with the context platform can be facilitated in other applications of the context-awareness which are related with software sensors. The implementation of such systems usually consists of a client (a user or service invoking an interaction) and a service (providing a response) Loke: (2006) Exchanges from both ends are being monitored. As

this infrastructure deals with the activity of the user; all the available data is considered helpful to gain insight into said activities. Furthermore, our mapping methodology (Pathan, et al.,2011) is significant in providing meaning to the extracted data. The architecture is depicted in **(Fig. 1)** and we will discuss the components in more detail in the corresponding sections.

In this architecture context information is provided by various context sources, which include web Services. The data from the sources is either provided directly (the usual approach for hardware sensors) or through observation of message exchanges (the proposed approach for software sensors). This data forms input to reasoning which allows to determine a user's activity.

We have divided this architecture in three parts which further explains each module of the architecture depicted in the **(Fig. 2).**

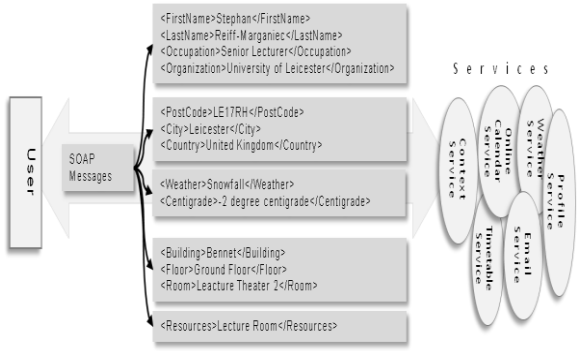

Fig. 2: An Example of Formatted SOAP Message Request of A Calendar Service Showing An Example of the Data that can be Achieved Through Exchanges

A. Sensing Context Information

During the exchanges occur between user and services data moves to and from the services in a SOAP envelope **(in Fig. 3)** and remains unused for further processing. (Fig. 1) combines traditional web services with semantic web technologies to not only make use of that data but also out of that data we can actually sense the activity of the user without a physical sensor which is a trend so far.

The user's context information is provided by the exchanges occurred between a user and the Services, whenever any user is using any kind of service, she is actually providing data in the form so that a service can respond in kind. This data is processed to achieve fruitful results. Here Sensors are capturing those exchanges and context acquisition module is taking that data to the system as in **(Fig. 2)**. This raw data is further processed by mapping module which is the key to this research and has been described in the next sections. By putting that data to the context model is the beauty of this architecture which combines web services with the semantic web technologies to sense the activity of the user, as we know that raw data makes no sense but when that data is mapped into the structured knowledge with all the reasoning rules makes sense and hence we can query and reason the knowledge base by applying inference rules the already available data along with the data which is coming from sensors in the form of user's context, logically.

1) Context Providers

Profile, Calendar, Timetable etc. Services are the web services that provide data when a user interacts with those while performing their tasks hence named as Context Providers.

A Profile Service provides the entity information which can help to collect and aggregate information according to user's own record. This can be the name, affiliations, education, etc. Calendar Services show the scheduled activity of the user and is further explained in the coming sections. Email Services can also be used to infer scheduled or deduced activities of the sender with the help of looking at sender, receiver and subject of the message only. A Timetable Service shows the scheduled activity of the user from the organization's side where s/he works. Weather Services can strengthen the deduced activity should the reasoning rules are applied on the collected context. The Context Service provides the actual context defined by users. If there is any conflict with users' provided context then heuristics can be used to resolve it.

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope" xmlns:ser="http://service.calendar.google.com">
  <soap:Header/>
  <soap:Body>
    <ser:getTitle>
      <!--Optional:-->
      <ser:title>Meeting</ser:title>
    </ser:getTitle>
                <ser:getFromDay>
                <!--Optional:-->
                <ser:fromDay>10/07/2012</ser:fromDay>
                </ser:getFromDay>
                <ser:getFromTime>
                <!--Optional:-->
                    <ser:fromTime>18:00:00</ser:fromTime>
                </ser:getFromTime>
                <ser:getToDay>
                <!--Optional:-->
                    <ser:toDay>10/07/2012</ser:toDay>
                </ser:getToDay>
                <ser:getToTime>
                <!--Optional:-->
                    <ser:toTime>19:00:00</ser:toTime>
                </ser:getToTime>
                <ser:getLocation>
                 <!--Optional:-->
                    <ser:location>Leicester</ser:location>
                </ser:getLocation>
                <ser:getDescription>
                <!--Optional:-->
                <ser:description>This is a meeting with a colleague</ser:description>
                </ser:getDescription>
  </soap:Body>
</soap:Envelope>
```

Fig. 3: An Example of SOAP Message Request (RAW) of a Calendar Service

2) Sensors

In this work a sensor means software sensor which is used to extract context information from different services and provide this to the Context Acquisition Module. We have already presented background information regarding architecture in section 1 that are placed in between the user and the

services. These sensors play an important role in extracting the data from the exchanges occurring between a user and the services. While using the services we actually send user data to a service to be responded to and this data is carried through the SOAP protocol. SOAP (simple object access protocol) is an XML-based communication protocol to make applications exchange structured information to the web services over HTTP. SOAP allows getting around the firewalls and is a W3C recommendation.

While running services the SOAP exchange information is forwarded to the sensors. The SOAP monitor module of the services is not enabled by default for security reasons because it exploits the data regarding that service and the user. There are two kinds of handlers to set one is in client side and the other one in the server side. Handlers allow you to intercept SOAP messages and can reside on both the client and the server side during a service invocation.

$$Client \rightarrow SOAP\ Envelope \rightarrow TCP\ Monitor$$

$$TCP\ Monitor \rightarrow SOAP\ Envelope \rightarrow Server$$

You can have a handler to capture a SOAP request/response message exactly before it goes from or returned to the client. The SOAP messages are redirected to or from the TCP Monitor. The client connects to the TCP Monitor and TCP Monitor is configured to connect to the server.

The same applies when the SOAP Envelope is sent to the Client. With reference to Figure 5, we show the interaction between a user and services (in **(Fig. 2 and 4)** highlighting the data that has been extracted from the soap request and response messages.

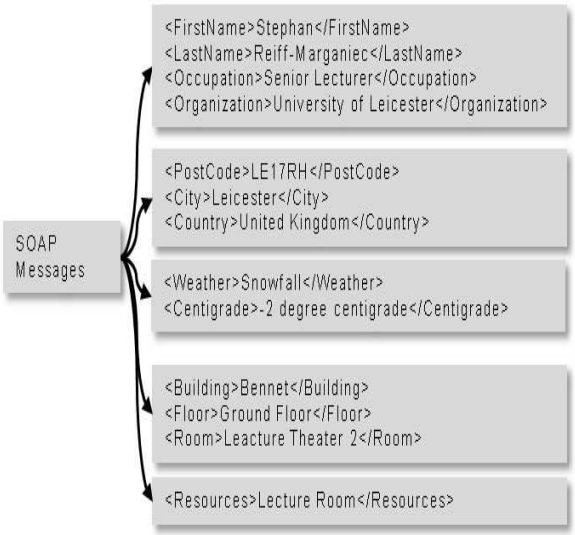

Fig. 4: Sample User Data inside SOAP Message

B. Linking Context and Services

There is still research going on to sense the activity of the user. Some of them only tell the online status which is either updated by the user or if forgotten then the only understood context no matter between that period of time the user has done number of other activities. Some of the Advertising services (e.g. Gmail (one of the applications of Google)) look at the description and subject of the email and advertise; no matter if that advertisement is necessary for the users at their situation or not and hence failed to respond according to context.

In traditional web services the SOAP messages carry data which is unprocessed for further use but carry a useful information regarding a user's interaction with that service and if processed can be useful in further ways. For example if a user is using a Calendar Service and she is putting data into the required form for example She wants to add an event which requires details like Title of the Event, Time (From to To), Location, Description etc. While inputting these details user is actually passing her crucial data through SOAP messages and this raw data in SOAP messages carry information of the User's Activity but in a form which is not further processable by machine and hence become a source to update the Calendar Service only. This part of the architecture (discussed in the next sections) explains how we make use of that data.

1) Context Acquisition Module

The Context Acquisition Module acquires user's context from different services with the help of sensors (discussed in previous section). These sensors capture the raw SOAP messages that are being used to invoke the web services.

For example while creating a new entry in the Calendar Service; a user inputs the data that travels from the client to the web service using the SOAP protocol. These SOAP messages carry the arguments a user passes to create an event in the calendar, such as Title, FromDay, FromTime, ToDay, ToTime, Location and Description in the request.

In the system we place sensors to sense this data. In this particular example the request is more important than the response because a response might only be the acknowledgement (SOAP response message) stating that the entry is added into the calendar.

Taking another example of Weather Service, the request and response both are important to process because the request reveals the city of which the weather is asked and the response shows the Fahrenheit/Celsius, hence affects the activity of the user if the weather is pleasant, normal or worst.

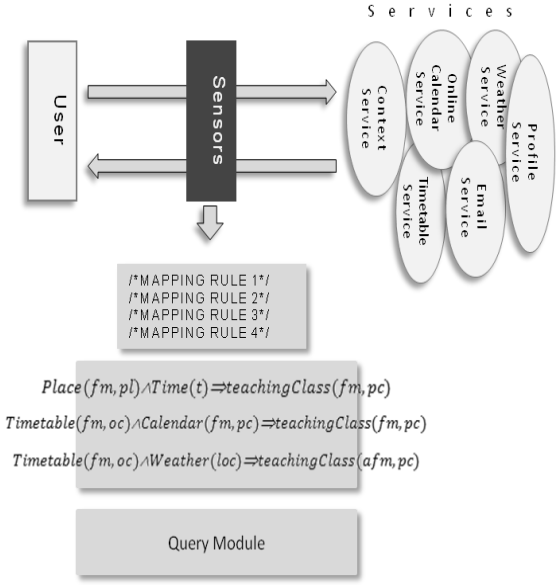

Fig. 5: An Example of The Architecture Applying Mapping Rules and Semantic Rules

2) Mapping Module

The Mapping Module maps the data captured from context providers into a semantic representation so that context can be further processed, shared and reused by other components. This approach addresses both the interoperability, extendibility and the reuse issues. As this approach is based on semantic technologies so that users can adapt the services according to their own needs.

3) Context Knowledge Base

The Context Knowledge Base stores this data into a triple format, making a statement i.e. subject, predicate and object and instance statements specific to the individuals. It can provide various services with the help of the Query Module for querying, notifying, adding, deleting or modifying context knowledge stored in the context database.

C. Retrieving Context Information

To sense the activity of the user using hardware sensors there might be the need of proper image processing etc. but for software sensors there is a dire need of structured knowledge stored in a triple form so that automatic reasoning can be performed. An example is shown in Figure 4. From different exchanges between user and services provide us with SOAP messages and therefore the raw data. The challenge is to make it structured and transform it into a knowledge which is required for the inference mechanism. Reasoning cannot be done logically, if data is unstructured, hence the utmost need arises to structure the data and apply the reasoning onto it. Once knowledge is there then the need is to infer some more facts by applying rules onto the existing facts and this is the important feature. For example if you have SOAP messages containing the event title, time and location mentions the complete context, if the rules are applied properly. And if further added weather conditions of that particular location can deduce with the reasoning rules. Office calendar and work calendar may further strengthen the rules and hence sense the activity context with the combination of web services and semantic web technologies.

To instantiate an ontology after making one; we are not only considering the method which create the new one like the available literature Bohring, *et al.*, 2005 and Ghawi *et al.*, 2009) and not only inserting it without any logical way like (Rodrigues, *et al.*, 2006) and Ferdinand, *et al.*, 2004) but we are also instantiating the existing knowledge and providing methodology which can further extend the ontology.

1) Context Reasoning Engine

After mapping data of user's context in the web ontology language with the help of mapping methodology the Context Reasoning Engine infers the scheduled and deduced context of the user by avoiding context conflicts into the knowledge base with the help of inference procedures. The semantic rules (Pathan, *et al.*, 2011) help to infer the data using Semantic Web Rule Language (SWRL) (Horrocks, *et al.*, 2004)

2) Context-aware Services

Once we have user's context then the Context-aware services can be utilized so that these can adapt according to user's situation in a context-aware environment.

Based on this architecture, we have implemented an activity-aware prototype for the scenario that senses the activity of the user in the context-aware environment using software sensors.

3. SCENARIO AND DISCUSSION

D. A Teacher in the class room

A faculty member (A Teacher) teaching a postgraduate class and trying to concentrate on a very core topic of the subject and trying to make it simple so every student in the class can easily understand it.
In this scenario Dr. Kim (a teacher) is teaching to the class of 70 students in lecture theatre. The topic is very new to most of the students and can be understood with the open discussions while explaining different situations. Dr. Kim is using the networked computer, the projector, white-board and any other visual aids to convey his message; on the other side students can have their notebooks or networked tablets with them for note taking. Dr. Kim can be approached by the outside entity using telephone (attached in the classroom), mobile (personal) or Email Service but usually mobile phones are kept off during the classes to avoid any disturbance.

While giving a lecture Dr. Kim saw a notification of a new email on the system and avoided

to open it in front of the seventy students. After finishing the lecture (that lasted for almost two hours), Dr. Kim saw that Mr. Jim wants to meet with him regarding a final version of the research paper. Dr. Kim asked him to meet with him at around 1pm the same day by replying with email because at 4pm he has another meeting pertinent to the research project.

When Mr. Jim found no reply from Dr. Kim after sending him a request for the urgent meeting, he deduced that Dr. Kim as holding a lot of responsibilities must have any important work to do and he could not manage to see my request. Mr. Jim decided to attend one of the two hours PhD seminars to be started at 1:20pm.

4. CONCLUSION AND FUTURE WORK

In this paper we have presented the architecture to sense activity context in the services-based context-aware environments. The framework supports the software sensors (unlike traditional tracking devices etc.) and adapts the literature to sense the activity of the user that interacts with the already available services to capture user records to facilitate the user itself.

In future SOAP messages of the other services could be explored to sense the other types of context i.e. Location and some other methods might be introduced to strengthen the confidence on the outcome.

REFERENCES:

Abowd, G. D., C. G. Atkeson, J. Hong, S. Long, R. Kooper, and M. Pinkerton (1997) Cyberguide: A mobile context-aware tour guide. *Wireless networks*, *3* **(5):** 421-433.

Baldauf, M., S. Dustdar, and F. Rosenberg, (2007) A survey on context-aware systems. *International Journal of Ad Hoc and Ubiquitous Computing*, *2* **(4):** 263-277.

Cheverst, K., N. Davies, K. Mitchell, A. Friday, and C. Efstratiou, (2000) Developing a context-aware electronic tourist guide: some issues and experiences. In *Proceedings of the SIGCHI conference on Human factors in computing systems* 17-24. ACM.

Budzik, J. and K. J. Hammond (2000) User interactions with everyday applications as context for just-in-time information access. In *Proceedings of the 5th international conference on Intelligent user interfaces* 44-51. ACM.

Bohring, H and S. Auer (2005). Mapping xml to owl ontologies. *Leipziger Informatik-Tage*, 72, 147-156.

Finkelstein, L., E. Gabrilovich, Y. Matias, E. Rivlin, Z. Solan, G. Wolfman, and E. Ruppin, (2001) Placing search in context: The concept revisited. In *Proceedings of the 10th international conference on World Wide Web* 406-414. ACM.

Christensen, E., F. G. Curbera, Meredith, and S. Weerawarana, (2001). Web services description language (WSDL).

Ferdinand, M., C. Zirpins, and D. Trastour, (2004) Lifting xml schema to owl. *Web Engineering*, 776-777.

Gmail. https://mail.google.com/. Last accessed on 7th August 2012.

Hadley, M., N. Mendelsohn, J. Moreau, H. Nielsen, and M. Gudgin, (2003) SOAP Version 1.2 Part 1: Messaging Framework. *W3C REC REC-soap12-part1-20030624, June*, 240-8491.

Ghawi, R. and N. Cullot, (2009) Building Ontologies from XML Data Sources. In *Database and Expert Systems Application, 2009. DEXA'09. 20th International Workshop on* 480-484. IEEE.

Horrocks, I., P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean, (2004) SWRL: A semantic web rule language combining OWL and RuleML. *W3C Member submission*, 21, 79Pp.

Hong, J. I. and J. A. Landay (2004) An architecture for privacy-sensitive ubiquitous computing. In *Proceedings of the 2nd international conference on Mobile systems, applications, and services* 177-189. ACM.

Loke, S. (2006). *Context-aware pervasive systems: architectures for a new breed of applications*. Auerbach Publications.

Mitra, N. and Y. Lafon, (2003) Soap version 1.2 part 0: Primer. *W3C recommendation*, 24, 12Pp.

Organization for the Advancement of Structured Information Standards. https://www.oasis-open.org/, last accessed on 07th August 2012.

Pathan, K. T., S. Reiff-Marganiec, and Y. Hong, (2011) Mapping for activity recognition in the context-aware systems using software sensors. In *Dependable, Autonomic and Secure Computing (DASC), 2011 IEEE Ninth International Conference on* 215-221. IEEE.

Pathan, K. T., S. Reiff-Marganiec, A. A. Shaikh, and N. Channa, (2011) Reaching Activities by Places in the Context-Aware Environments Using Software Sensors. *Journal of Emerging Trends in Computing and Information Sciences*, 212Pp.

Rodrigues, T., P. Rosa, and J. Cardoso, (2006) Mapping XML to Exiting OWL ontologies. In *International Conference WWW/Internet* 72-77.

Truong, H. L., C. Dorn, G. Casella, A. Polleres, S. Reiff-Marganiec, and S. Dustdar, (2008) inContext: On Coupling and Sharing Context for Collaborative Teams, 225-232.