ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

COLLEGE OF NATURAL SCIENCES

DEPARTMENT OF COMPUTER SCIENCE

Architecture for Context Management and Context-Aware Service Delivery

Yonas Tadesse Woldegebriel

A THESIS SUBMITTED TO

THE SCHOOL OF GRADUATE STUDIES OF ADDIS ABABA UNIVERSITY IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE OF MASTER OF SCIENCE IN COMPUTER SCIENCE

30 May, 2014

ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

COLLEGE OF NATURAL SCIENCES

DEPARTMENT OF COMPUTER SCIENCE

Architecture for Context Management and Context-Aware Service Delivery

Yonas Tadesse Woldegebriel

Advisor: Dejene Ejigu (PhD)

Signature of the Board of Examiners for Approval:

Name	**Signature**
1. Dejene Ejigu (PhD)	_____
2. _____	_____
3. _____	_____

30 May, 2014

DEDICATION

To my parents, for teaching me what is important in life

ACKNOWLEDGEMENTS

First and foremost, I thank God who endowed me with the strength to tackle problems in life and also to finish my work for his gift, care and protection. I must also thank Saint Virgin Mary, Holy Mother of God, for Her intercession and all the Saints in the Kingdom of God for their blessings.

I offer my sincere gratitude to my supervisor, Dr. Dejene Ejigu, who has supported me throughout my thesis with his patience, motivation, enthusiasm and immense knowledge whilst allowing me the room to work in my own way. His guidance, encouragement and advice helped me through the learning process of the thesis work.

My utmost gratitude goes to my mom, Genet Yihdego, and my dad, Tadesse Woldegebriel. I am grateful for the love, care and lessons they gave me in life. Without their cooperation, patience, understanding, sacrifice, unremitting support and encouragement, none of these would have happened. I will be grateful for their love forever.

It gives me immense pleasure to also thank my brothers Dr. Zekarias Tadesse, Dr. Fisihatsion Tadesse, Zelalem Belay and Yosef Belay for the wonderful experience we have had when we grew up together. Special thanks go to Dr. Zekarias and Dr. Fisihatsion for being an inspiration in the academics and for their invaluable and continual support over the course of my academic life. Also with Dr. Fisihatsion, I would like to thank Dr. Wudase Akalewold and my sister, Alem Meaza, for being there for me at difficult times during the study of this research.

Finally, I would like to thank my colleagues, families, teachers, friends and others whom in any way possible had contributed to the successful accomplishment of this thesis work.

Table of Contents

List of Figures

List of Algorithms

ACRONYMS

ABox	Access Box
CASS	Context-Awareness Sub-Structure
CMF	Context Management Framework
CoBrA	Context Broker Architecture
CoBrA-Ont	CoBrA-Ontology
CORTEX	CO-operating Real-Time sentient Objects: architecture and EXperimental Evaluation
DAML+OIL	Combination of DAML and OIL DAML - DARPA (Defense Advanced Research Projects Agency) Agent Markup Language OIL - Ontology Inference Layer
DL	Description Logic
EHRAM	Entity, Hierarchy, Relation, Axiom and Metadata based context representation model
Enhanced CoCA	Enhanced Collaborative Context-Aware Service Platform
HCOM	Hybrid Context Management model
HF	High Frequency
HTML	Hyper Text Mark-up Language
IC	Integrated Circuit
ID	Identification
IEEE	Institute of Electrical and Electronics Engineers
KR	Knowledge Representation
LAN	Local Area Network
LF	Low Frequency

OML	Ontology Markup Language
OWL	Web Ontology Language
R-CCANS	RFID based Campus Context-Aware Notification System
RDF	Resource Description Framework
RDFS	Schema for RDF
RF	Radio Frequency
RFID	Radio Frequency Identification
SBox	Semantic Box
SDA	Semantic Data Access
SHOE	Simple HTML Ontology Extensions
SIG	Bluetooth Special Interest Group
SOCAM	Service-Oriented Context-Aware Middleware
UCSD ActiveCampus	University of California – San Diego Active Campus project
UHF	Ultra-High Frequency
UML	Unified Modeling Language
W3C	World Wide Web Consortium
WLAN	Wireless LAN
WPA	Wi-Fi Protected Acess
WSN	Wireless Sensor Network
XML	eXtensible Markup Language
XOL	Ontology Exchange Language

ABSTRACT

The emergence of various technological resources in the computer industry has shown its advancement in recent years. It provides computers the advantage of high processing capability with reduced physical size options and small size wearable appliances along with wireless communication technologies have emerged. This, in turn, led to a new computing paradigm called pervasive or ubiquitous computing. Pervasive computing aims at embedding such resources into the environment to ease service delivery while reducing the amount of user effort and attention. It helps to minimize user distraction providing context aware services. Thus, it takes context information and context management at its forefront to achieve its goal, by analyzing and processing any relevant data about a given entity to arrive at a decision.

In this study, generic context aware service management architecture is proposed to support and enhance the context management starting from context acquisition to its final delivery of services. A way of resource handling, context acquisition, context management, service delivery and security are provided in the architecture. The context management module processes the context reasoning and context decision considering rules and historical context information. A way of context representation and model using ontological modeling approach is provided to enrich context management.

To evaluate usability, applicability, reusability and extensibility of the system, the study considered scenarios in a smart campus which require a one card identification system. With the proposed architecture at hand, a one card system in a university setting is modeled using the proposed context modeling approach to extend its functionality beyond its traditional counterpart. Unlike the traditional way, which uses a barcode reader to handle individual cards, it considers context information to perform identification. Functionalities of components in the proposed architecture are also evaluated using implementation of scenarios like notification services for service areas in the domain. Technologies like RFID and Bluetooth are exploited in the implementation to support context acquisition and resource management.

The study enhances mechanisms for resource discovery, context management and pervasive service deliveries. It proves the use of the proposed architecture for application in university setting and other related domains while retaining necessary functionalities. Also, security using log management method is offered as a means of security in pervasive computing.

Key words:

Pervasive/Ubiquitous Computing, Context, Context awareness, Ontology, Smart Campus, RFID

CHAPTER ONE

INTRODUCTION

1.1 Overview

Recent developments in hardware technology have led to the creation of more portable devices and the growth of computing. The invention of handheld and wearable computers, wireless communication and devices that sense, identify and control appliances has enabled computing to highly involve in the environment and impact social life. As the trends show, people nowadays want to access information anytime and anywhere using their personal devices that they carry all the time, but with much less effort and distraction than before. However, it's beyond the traditional distributed system capability to handle such extremely mobile scenarios. This has, therefore, contributed to the emergence of pervasive computing so that people's needs are met.

Pervasive computing envisions [1, 2] at a model of human-computer interaction that information and computational capability present throughout the physical environment integrates into a broad range of everyday object and/or action without being visible to users. Its goal is to deliver context aware services using computing systems with a lessor user effort and attention. By embedding different computing devices in to everyday items, they can be augmented with self-awareness and awareness of their surroundings in order to provide new functionality and patterns.

People in different areas use access to technology and computational capabilities to facilitate their tasks. The advent of pervasive computing has also contributed in bringing a way of providing context aware services with minimal user distraction. It provides advantage to application areas in automating their service deliveries. For instance, university environment is among the environments that require a proper study in delivering such services and can benefit from its application. There are many reasons that campus environments are ideal for the development, deployment and testing of ubiquitous computing technologies [3]. Mobile access to information and services in a university setting is one central reason for the use of ubiquitous computing model in a campus. The mobility of technology along with the role of students in the university, their relationship to each other and other social groups, the need for accessing

services in the university, academic expectations placed up on them, and other things that students routinely encounter and navigate are examples of what is required in universities with regard to computation capability. Another reason for a campus environment to be convenient is that many technologies are developed in university research. Various technologies which are highly networked exist in its premises with strong infrastructural support and a large number of potential users have access to these infrastructures. In addition, services that require no distraction to contexts such as location, time, identity and other situations exist in universities.

Hence, this study aims at providing generic architecture of context management and context aware service delivery. Context aware services are modeled, designed and developed with service management focusing on incorporating resource discovery, context acquisition, context management, service delivery and security mechanisms. The study also exploits different technologies and tools for identification and other context aware services providing the support for smart campus implementation.

1.2 Motivation

The motivation for this study is to apply ubiquitous computing technology in application areas like university settings. The advent in pervasive technology which is regarded as the next wave in the evolution of computing [2], the growth of tools for semantic use of context information, and the rising use of technologies like RFID, WSN and others in pervasive computing are also among the key motivating parameters for this work.

Another motivating factor is application advantage of context awareness in pervasive environments in creating smart spaces. The motivation behind selecting a university campus setting for implementation is the effectiveness of using one card system for services like entrance monitoring, cafeteria, book issue and return, notification and other related services for students and other university community members in such an environment. This one card system is aimed at improving and replacing the current manual delivery of services in universities that involve different cards to handle different services with only one card which can handle all the services.

1.3 Statement of the problem

Context aware service deliveries in pervasive environments involve a set of functionalities that require attention. These include:

The need for handling ubiquitous technological resources: The existence of various resources `distributed in the environment can be considered as an advantage by treating them as potential sources of context. Discovery of resources and techniques to handle identified resources is required to enable access to context data. It is necessary to provide approaches to tackle problems associated with discovery and handling of these resources.

The need for approaches in collecting context from sources of context information: There are a variety of significant sources of context information. Identification of such sources and a way of collecting raw context data are necessary before further processing takes place. Context acquisition from the identified sources of context should consider assimilation and preprocessing requirements apart from collecting them. Use of historical context information as one basic source of context is important. Architectural support in providing approaches to such requirements is, therefore, a center of attention in context awareness.

The need for management of context data: Context aware computing requires processing of raw context data collected to arrive at a final context decision to enable context aware service delivery. Issues in context management include context abstraction, assimilation of related context, change in context, context reasoning, a higher level description of context and context decision. An approach to handle the dynamic nature of context data is also required. An enhanced model towards addressing these and other related issues should be included in context aware architectures.

The need for delivering pervasive service according to the context information: Automations of services in pervasive environment require its delivery to consider context information. Service delivery should be based on processed context. It should also consider to process services on behalf of the user when necessary. Hence, analysis of service delivery is a crucial component required in pervasive frameworks.

The need for integration of security mechanisms to prevent and monitor threats: The deployment of pervasive computing systems inevitably raises security concerns. The ability of such systems to synthesize partial information and react to situations without explicit user interaction, confirmation or consent means that issues of trust, safety and privacy become exceptionally important. Dynamic nature of context also raises another concern for security mechanism to react to change of context. Prevention of unauthorized access to context information and identification of other security issues should be made in early stages of pervasive computing framework designs.

The design of architecture considering the required functionality should help pervasive environments to provide smart spaces. For instance, a university environment comprises many studies [4,5,6,7] focused on the process of providing pervasive teaching and/or learning only. With functionality in architectural support it can be possible to embed other service deliveries in the environment to create smart campus that is aware of its context.

Hence, to provide these and related context aware services and to automate them, architectural support is the crucial step in its development and in bringing better context aware systems. Although there are research efforts to provide such functionalities, architectural design is still in its early stage to answer requirements of pervasive computing.

1.4 Objectives

1.4.1 General objective

The general objective of this research work is to develop generic context aware service management architecture.

1.4.2 Specific objectives

The specific objectives devised to accomplish the general objective are:

- To identify potential sources of context and model relevant context information
- To provide resource handling and context data handling techniques
- To exploit different technological resources for the purpose of context acquisition

- To enrich context representation and acquisition techniques
- To propose a model for context management for delivery of context aware services
- To provide service delivery mechanism that can react towards sensed and processed context
- To integrate security mechanisms to prevent and monitor threats in pervasive service deliveries
- To develop a prototype system
- To test and evaluate the system

1.5 Scope and limitation of the study

The main scope of this work is to propose a semantic context management model, design the architecture and implement, test and evaluate a prototype of pervasive context aware system based on the designed architecture. This study does not focus on improving the way of communication between pervasive devices. Enhancing communication protocols and communication security between such devices are not the concerns of this study. Its focus is to study, design and model context aware service management architecture in pervasive environments.

1.6 Methodology

Different methodologies are employed in this research in order to accomplish the general and specific objectives of this study. First is to conduct an extensive review of relevant literatures to acquire a deeper understanding of the research area and its problem domains. Previous researches in the areas of pervasive computing, context awareness and related issues are investigated to visualize their importance towards this research. Existing works related to this research work are also assessed to identify and point direction in providing solution to identified problems.

Using the inputs from conducted review of literatures, useful models and design requirements are then identified to arrive at sensible solutions. Based on these requirements, a model of context and design of architecture is proposed which can guide implementation of certain applications. Tools and techniques that can fulfill the design of the architecture are then labeled for the

implementation of the prototype system. The prototype considers the necessary identified services to prove the significance of the proposed solution so that it should carefully follow the specification provided in the model and architecture. To validate this prototype and the significance of the current work, testing and evaluation techniques are identified and used.

1.7 Application of the study

The significance of this work is to provide a well-studied model and architecture in order to create a smart space with context aware services. The proposed architecture for pervasive services can also be an input for further works in different domains that requires such service deliveries. It can also be provided as a pre-step in other related studies. Moreover, it can be an integral part of studies in human-computer interaction or artificial intelligence.

For instance, the result of the context aware system that is modeled and developed will also contribute in developing applications in a university. It is to strengthen service delivery in campus environments and other domains with a similar setting. Among which are:

- *Notification system*: provides notification, if any, checking user's context. It automatically delivers the notification whenever it is required and available.
- *Café System:* a tag can be embedded into students ID card and by the time they approach to get the services, detailed information about a specific student (including a photo of the student to confirm on identification) can be retrieved and displayed on monitor of the control computer, and checks whether the student has already received the services or not. This has an advantage over the one card system in which it is tiresome to check every student's barcode using the barcode reader or manual checkups.
- *Student attendance monitoring*: student information can be sensed or read by technologies like RFID with a tag embedded on the ID card, or with a detection mechanism using WSNs.
- *Entrance control* in the gates or other places which require monitoring
- *Automating library services* like book issue and return control, providing notification and detail information on new arrivals and other related services.

Other domains can also use the generic architecture advantage in developing applications, for instance, in public transportation systems, hotels and supermarkets to check and record user identity and provide relevant services based on read context.

1.8 Thesis outline

The remaining of the thesis is structured in the following way. Chapter 2 starts with review of literature on pervasive computing and the basics of its working environment. Then concepts of context, context awareness and context-aware computing are provided. The ways to acquire context data with support of various existing context acquisition technologies are next discussed. Modeling approaches and techniques to model and represent the acquired context information are also presented in detail.

Review of related works is presented in Chapter 3. Works that have significant relation with this thesis are assessed. Generic architectures, and works proposed in other domains but are related to the thesis are taken into consideration. Moreover, other related works in pervasive computing in the area of smart campus are discussed.

Chapter 4 concentrates on providing design of generic architecture. Issues relevant for context capturing, context representation and context management are addressed; model of context management for supporting design and implementation of context services is provided; and, context representation for the design of identification in smart campus is considered.

Chapter 5 presents an implementation and discussion of a prototype system based on the architecture proposed. Different on ground scenarios in a university setting are presented for experimentation.

The conclusion in Chapter 6 summarizes the contributions made in the thesis, and critically assesses the shortcomings detected in the course of this research. Furthermore, new issues that have been surfacing while working on the thesis are suggested as future works.

CHAPTER TWO

LITERATURE REVIEW

This Chapter presents a review of literatures conducted in order to understand the problem associated with the realm of the objectives of this thesis and also to identify appropriate direction. An extensive review is made on areas of pervasive computing, context awareness, context modeling approaches, and technologies that support the effectiveness of context awareness.

The review discusses main concepts in pervasive computing, the requirements to fulfill its objectives and major practical areas of application. Context awareness takes the major share in pervasive computing and is also discussed in this Chapter, followed by its various modeling approaches and context input sources. Among the context acquisition tools, RFID technology, Wireless Sensor Network (WSN) and wireless connectivity such as Bluetooth and Wi-Fi are presented in detail.

2.1 Pervasive Computing

A team of visionaries led by Mark Weiser at Xerox Palo Alto Research Center first envisioned the concept of pervasive/ubiquitous computing in late 80's **[2].**

Mark Weiser states –*The most profound technologies are those that disappear*" proving the merging and interpretation of information as a key power to drive pervasive computing with the help of the capability of pervasive devices in capturing that information. He also envisions advantages of its application in areas like campuses [8].

Pervasive computing represents a major revolutionary step among which distributed systems and mobile computing contribute the major share [9]. Remote communication, fault tolerance, high availability, remote access, security, mobile networking, mobile access and adaptability are among the key features that are contributed to its evolution.

It proposes a digital future in which component technologies are embedded seamlessly into the fabric of the world around us [10]. It refers to a seamless and invisible computing environment, integrated with existing and future pervasive devices. It provides dynamic, proactive and

context-aware services to the user by acquiring context knowledge from the environment and composing available services. Hence, a pervasive computing system that strives to be minimally intrusive has to be context-aware [9].

2.1.1 Pervasive Computing Environment

Pervasive computing can support work, recreational or any other environment to facilitate activities, especially service deliveries, minimizing users' effort. Pervasive computing environment consists of a large number of independent entities that help transform such areas into computationally active and intelligent spaces.

Pervasive computing environment can then be characterized as one that saturated with computing and communication capability, yet so gracefully integrated with users that it becomes a ―technology that disappears" [9].

Smart Spaces

A space may be an enclosed area such as a meeting room or a corridor, or it may be a well-defined open area such as a courtyard or a quadrangle. A smart space is ubiquitous computing environment that encompasses such physical spaces. It can bring together two worlds that are disjoint by embedding computing infrastructure to other existing infrastructures, like building infrastructure [11]. The fusion of these worlds enables sensing and control of one world by the other.

Different projects have been suggested in regard to smart spaces which provide the user with smart environment such as home, hospital, classroom, etc. Smart Home [12,13,14], Smart Hospital [15,16,17,18], and Smart Classroom [4,5,6,7] are among the proposed works providing intelligent support for home, hospital, and classroom environments, respectively. Furthermore, other spaces can also get similar intelligent support with the help of ubiquitous computing to facilitate activities in those environments easing user effort. For instance, University campus setting, which is also the focus of this wok, can gain computing advantage by being smart campus. It is a suitable environment holding different ubiquitous computing technologies with the existence of required day to day interaction [3]. The interaction includes institutional activities of students and other communities of the university with the existing technologies.

2.2 Context, Context Awareness and Context Aware Computing

Context and context awareness have been central issues in ubiquitous computing since its very beginning. Context is very important since it provides information about the present status of people, places, things, and devices in the environment [19,20]. As context becomes basic element in pervasive computing to provide context aware services, researchers try to formally define context from different perspectives and come up with some relatively accurate definitions.

In the early development of context awareness, Schilit and Theimer [21] first described context as location, identities of nearby people, objects and changes to those objects. Schilit et al. [22] observe that context encompasses more than just the user's location, because other things of interest are also mobile and changing. They consider lighting, noise level, network connectivity, communication costs, communication bandwidth and even the social situation as context.

Brown in [23] defines context to be the elements of the user's environment which the computer knows about. In one of the more extensive investigations of context-based computing, Ryan et al. [24] define context as "location, identity, environment, and time." Hull et al. [25] consider situation to define context as the aspects of the current situations. Schmidt et al. [26] define context as "knowledge about the user's and IT device's state, including surroundings, situation, and to a less extent, location." Also, from applications perspective context is defined in [27] as the set of environmental states and settings that either determines an application's behavior or in which an application event occurs and is interesting to the users.

Context in [28] is also, defined, in terms of hierarchical description of historic dependency, as a set of changing relationships but based itself on historic relationships and hence the historic relationships are guides to shape the current relationship.

A widely used definition, which is given by Dey and Abowd [29], is adopted in this study. They defined context as: "*Context is any information that can be used to characterize the situation of an entity. An entity is a person, place or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves.*" Dey [30] also considers context as the user's emotional state, focus of attention, location and orientation, date and time, as well as objects and people in the user's environment.

The three entities that can be distinguished are places (rooms, buildings etc.), people (individuals, groups) and things (physical objects, computer components etc.). Each of these may be described by various attributes that fall into four categories which are identity (a unique identifier for each entity), location (an entity's position, co-location, proximity etc.), status (intrinsic properties of an entity) and time (to accurately define situation, ordering events etc.).

The description of context led researchers to a better understanding of context awareness and context aware computing. Context awareness means that one is able to use context information. To provide adequate service for the users, applications and services should be aware of their contexts and automatically adapt to their changing contexts – known as context-awareness [31, 32].

The term context-aware computing is commonly understood by those working in context-aware, where it is felt that context is a key in their efforts to disperse and transparently weave computer technology into our lives [33]. This gives context-aware systems a goal to acquire and utilize information on the context of a device in order to provide services that are appropriate to the particular people, place, time, event, etc. It aims to provide context-aware access to information, communication and computation.

With the emergence of context awareness, projects like active badge location system [34], location-aware tour guides [35, 36, 37] and cricket compass [38] were introduced. These projects focused on providing information according to users' location context.

Various considerations have been given on criteria to whether a given system is context aware or not. Byun and Cheverst [39] consider a system as context-aware if it can extract, interpret and use context information and adapt its functionality to the current context of use. Adaptation is also considered in [40] which characterized context-aware systems to be able to adapt their operations to the current context without explicit user intervention. It targets at increasing usability and effectiveness by taking environmental context into account. Moreover, Dey and Abowd [29] set situation abstraction as a requirement in order for systems to be considered context-aware. Situation abstraction is a description of the state of relevant entities. The requirement accepts a given system as context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task.

Context aware system, therefore, should most consider designing context information carefully for its better use in the computing world. Henricksen et al. [41] characterize context information as that exhibits a range of temporal characteristics which is highly interrelated, imperfect and can be represented in many ways. Additionally, when the computing, user and physical contexts are recorded across a time span, we obtain a context history, which could also be useful for certain applications [27]. Context-aware systems should deal with such characteristics by representing aspects of context information such as how the information representing the context is obtained; how it is represented; and, for what objectives and purposes the context information is used for [42]. This is where context acquisition techniques, context model and its modeling approaches should come in. The reason for modeling context is to better understand users' activity, to find out their needs related with the respective activity and to respond to them with the necessary information required [28].

2.3 Context Acquisition

Context data acquisition is an important step when designing context aware services. It is a prerequisite that can be preserved with the help of different physical sources providing the necessary inputs by capturing relevant physical context data. Applying sensors, network information, device status, and browsing user profiles are among the variety ways of retrieving context information. RFID and WSN are the two main components in pervasive computing capable of sensing context. Other wireless connectivity, such as Bluetooth and Wi-Fi, are also among the major providers from various sources of context information. In addition to physical sources, virtual and logical sources can as well contribute as sources of context information [40].

2.3.1 RFID Technology

RFID is a generic term for technologies employing radio waves for detecting objects [43] and is a major and widely used component in pervasive computing environment [44]. It is an automatic identification technology that contributes to the area by identifying entities along with location information. It serves as a context input source easing user effort in initiating and further delivering pervasive services. It makes it easier to gather information about physical objects by providing a wireless non-contact reading system assigning them unique serial identification.

Pervasive computing applications having RFID can embed capabilities of computation and communication into a variety of objects, humans and other living species so as to identify them as well as to detect locations, media content and data services [45]. Such application environments can incorporate tremendous numbers of tagged entities to automatically identify, interact and collect information in a transparent manner without requiring manual intervention. Hence, RFID technology plays a vital role in potential pervasive application areas and also believed to contribute a major role in future computing. RFID technology with the use of radar is claimed to be used in World War II to identify between army and allied planes [46] and has been used in various other application areas to help strengthen service deliveries. Asset tracking, real time supply chain management, transportation, building access, healthcare, schools, finance, location identification, human identification, smart spaces and telemetry based remote monitoring are among the vast application areas which made use of RFID technology [47,48,49].

RFID systems consist of main components: reader, tag and host system [50,51,52]. The RFID reader, consisting of transmitting and receiving sections, is able to read and/or write data to tags via wireless transmissions [52, 53]. It transmits a carrier signal, receives the backscattered signal from the tag and performs data processing. It is also responsible for details of communication in establishing it, preventing collision, authentication and for the communication to an external host system [53]. It consists of one or more integrated or external antennas; radio interface for modulation, demodulation, transmission and reception; and, a control system to direct communication with the tag and interacts with applications [45]. It operates by broadcasting to all tags within range and receives responses to construct an inventory, select a particular tag and exchange information. Once the tag is identified, a reader may read data from or write to tag memory, depending on the permissions granted by the tag.

The technology is dependent on communication between RFID tags and RFID readers. RFID tag is a simple, small device consisting of IC and an antenna [43, 50, 51, 52]. The IC is used for modulating/demodulating radio frequency and processing information, while the antenna is responsible for receiving and transmitting radio signals. Tags can be attached or embedded into any device, object or implanted into living beings for tracking or identification and announcing its presence to a reader. A tag has a unique identification (ID) number encrypted and a memory

that stores additional data such as manufacturer, product type, and environmental details such as temperature, humidity, etc.

RFID tag classification is based on different requirements [49, 50]. The tags can be read-only or read-write. Read-only memory is factory programmed which cannot be modified after manufacture and hence stores a very limited quantity of static data. Read-write tags on the other hand can be read from as well as written into and can store a larger amount of data which can be dynamically altered.

RFID tag can also be classified into categories as passive, semi-passive (or semi-active) or active tags based on its energy source. The passive tag, also known as pure passive, obtains its operating power from the RFID reader in which the reader sends electromagnetic waves that induce current in the tag's antenna. The tag then reflects the RF signal transmitted and adds information by modulating the reflected signal. Semi-passive follows the same communication method but uses a battery to maintain memory in the tag or power the electronics that enable the tag to modulate the reflected signal. In comparison to these two types, active tag supports a longer read range, is more expensive and powered by its own internal battery to run the microchip's circuitry and to broadcast a signal to the reader. Active tags usually are read-write tags.

The range of the reader is dependent upon its operational frequency. RFID tags can operate at three frequency ranges [54]: LF, HF and UHF. LF tags are less affected by the presence of fluids or metals compared to the higher frequency tags. They are fast enough for most application, and are also cheaper than any of the higher frequency tags. However, low frequency tags have shorter reading ranges. HF tags have higher transmission rates and ranges but are also more expensive than LF tags. UHF tags are more expensive than any other tag and are severely affected by fluids and metals.

The communication between the tag and reader uses induction or propagation method of wireless signal. Induction operates using close proximity electromagnetic or inductive coupling and use LF and HF frequency bands while the later operates on far field by propagating electromagnetic waves in the UHF and microwaves frequency bands.

RFID communication is fast, convenient, and its application can substantially save time, improve services, reduce labor cost, thwart product counterfeiting and theft, increase productivity gains, and maintain quality standards [48]. It also provides advantages of tags in reacting adverse conditions, to store large amount of data and additional information, support longer read range and above all an easier implementation of entity identification [45]. However, there are also issues of concern in RFID technology's standardization, cost, signal collision, frequency, manufacture problems, security and privacy issues [49].

2.3.2 Wireless Sensor Network

WSNs are networks of small, cost-effective devices that can cooperate to gather and provide information by sensing and monitoring physical, chemical and biological environments through sensing of temperature, light, humidity, pressure, vibration, and sound via simple integrated sensors [48,50,52,55]. It is a system which is capable of self-configuring, self-networking, self-diagnosing and self-healing [56].

The sensor network is composed of a large number of tiny low-powered, low-memory and low-range wireless computing sensor nodes with extremely restricted computational, communication, sensing (measuring) and battery capabilities [51, 53]. Wireless sensor node consists of sensor, microcontroller, radio frequency transceiver, and power source. The microcontroller manages the procedures that make the sensor node collaborate with the other nodes. The process of collaboration between the sensor node and other nodes is managed by the microcontroller while a transceiver unit connects the node to the network. Power source is also an important component of a sensor since battery management is an important issue [53].

The communication to gather required information is done usually in a manner sensor nodes send their data to a specific sink node or monitoring station for collection. If all the nodes communicated directly with the monitoring station, the communication load would quickly drain the network's power resources. Therefore, with power management under consideration, the sensors operate in a self-organized, decentralized manner that maintains connectivity for as long as possible and let message communication via multi-hop spreading [56].

WSN is a very attractive solution for a wide range of applications and gain popularity for its social implications. WSNs provide cost-effective monitoring of critical applications including industrial control, border monitoring, environmental monitoring, military, home applications, intelligent transportation, control and health care applications [48, 51, 56]. Such application areas use sensor networks to gather users' context, which are circumstances or conditions that surround users' behavior and affect their actions. Services are then offered to the context source, which often involve automatic actions that the user may desire in a particular situation [55].

2.3.3 Integration of RFID and WSN

Another imminent step that will lead context acquisition to a high level of synergy and more technological advance is the integration of RFID and WSN. It will maximize effectiveness to broad range of useful applications providing extended capabilities, scalability, and portability with reduced costs [46]. It can incorporate the advantage of both devices to give an intelligent decision, since integration with sensors enables RFID to get more environmental information operating in multi-hop way. It gives the advantage not only to reveal an item's identity and location but also its current state, and any required and sensed context information. However, more research effort is necessary and important requirements [54, 57] such as accurate and reliable communication, energy efficiency, network survivability, should be taken into consideration.

2.3.4 Wireless Connectivity

Short-range wireless technologies form the basis of ubiquitous computing communication platforms. Communication platforms for ubiquitous computing are required to be flexible, self-organizing, highly scalable, and energy efficient. This is due to the reason in which a large number of autonomous entities communicate in potentially unpredictable ways in the envisioned scenarios [58]. Connectivity should also consider mobility which is a key to bridge the gap between computing and communications for a wide range of computing devices to enable sharing of resources. Bluetooth and Wi-Fi are two of the most widely used wireless technologies in consumer electronic devices. These technologies are among the short-range wireless

technologies that can provide the support of ubiquitous computing communication fulfilling its requirements.

Bluetooth

Bluetooth is one of wireless technologies for ubiquitous computing which is a low cost, low power radio frequency technology for short-range communications.

It is named after Herald Blatand, "Bluetooth", a Viking 10th century king [59]. The concept behind Bluetooth had its origin in 1994 when Ericsson Mobile Communications initiated a study to investigate the feasibility of a low power, low cost radio interface between mobile phones and accessories. It aimed at replacing numerous kinds of special cables used for interconnection. The name Bluetooth was officially adopted in 1998 and owned by the Bluetooth Special Interest Group which is formed by five companies: Intel, IBM, Toshiba, Ericsson and Nokia. The Bluetooth Special Interest Group[1] (SIG) is a group of companies that cooperate to define Bluetooth standards and qualify Bluetooth® products in which its membership surpasses 20,000 companies in 2013.

It provides ubiquitous connectivity and offers the ability to provide seamless voice and data connections. Other key benefits of this technology are its low power, long battery life, low cost, low complexity, and wireless connectivity.

Wi-Fi

Wi-Fi is the name of a wireless networking technology that uses radio waves to provide wireless high-speed Internet and network connections. Wi-Fi® is owned by the Wi-Fi Alliance[2] and refers to any wireless local area network (WLAN) that is based on the IEEE 802.11 standards. It is much more popular for WLAN environment being a "wireless Ethernet". It offers good radio coverage and data rate, and also provide WPA security mechanism. Hence, it is among the technologies that can be able to integrate context-aware systems in an existing Wi-Fi infrastructure.

[1] SIG – The Bluetooth Special Interest Group (https://www.bluetooth.org)
[2] Wi-Fi Alliance - is a trade association that promotes Wi-Fi technology and certifies Wi-Fi products (http://www.wi-fi.org)

Bluetooth and Wi-Fi both use second and third generation chips, with each communication method having reached a high level of maturity regarding design efficiency. However, due to the physical design of dies, Bluetooth is more efficient in terms of power consumption. In terms of data rate, Wi-Fi overcomes Bluetooth and supports better transfer of large amounts of information. But, when Wi-Fi link goes further from the access point, the data rate decreases. Although the maximum communication speeds are less than Wi-Fi, Bluetooth's throughput can also satisfy context-aware system requirements.

Since Wi-Fi and Bluetooth are complementary technologies, they often are used in close proximity to one another. Neither Wi-Fi nor Bluetooth was originally designed with mechanisms in place to deal with the interference that each creates for the other. Early indications are that both technologies can coexist in harmony, even without coexistence mechanism in place. This is largely due to early usage models that do not have both devices being used simultaneously, and to the fact that even when interference does occur, the consequences are not severe. Interference problems form collision and causes dropped packets. Coexistence is, therefore, a priority when both technologies are placed in close physical proximity. However, with the latest advances in technology innovation, particularly, with the new Bluetooth "adaptive frequency hopping" technology, both can operate without significant interferences in the same environment. It is even possible to co-locate them on one IC, residing in the same device, thereby reducing cost, size and time-to-market.

Wireless connectivity such as Bluetooth and Wi-Fi can both contribute to the needs of ubiquitous computing. It allows context capturing providing advantage of easy communication between pervasive devices.

2.3.5 Virtual and Logical Context Sources

Some contexts may not exist physically as such but made by software to appear to do so or even delivered using reasonable rational thought expected under some circumstances. Mathias et al. [40] describe these as virtual and logical sensors. Virtual sensors consider software applications or services as their source of context data while logical sensors combine physical and virtual sensors with support of additional information from various sources like database. The virtual sensors which base themselves on systems can capture context data from sources like electronic

calendar, e-mail and other related systems that are capable of holding context data. Users' activity in such cases can also be monitored from mouse movement, keyboard input and related activities.

2.4 Context Aware Modeling Approach

Since context information is proven to be basic in pervasive computing and to provide necessary context aware services, a way to represent and manage captured context is a relevant task to get the required computing advantages. A model of context, therefore, is necessary in order to go through the process of providing its service. A well designed model is a key in bringing good performance in any context aware systems. It can give representation on how to get hold of and process context data having the purpose of enabling context aware computing with re-use of its architecture [28]. It can also provide a support for context aware systems to reason about contexts obtained.

Context Reasoning

Among key issues that context model should support is reasoning about context. A context aware system should be able to identify and use more complicated and higher level context to support more specific pervasive services. In order to achieve this, a system must have a thorough knowledge and understanding of its environment with the help of context reasoning. Context reasoning is about deducing new and relevant information to the use of application(s) and user(s) from the various sources of context data [60]. It should also make use of historical context information since past context, that characterize a past situation, and valid present context can be used to predict future context information either automatically or manually.

The need for reasoning in context aware systems derives from imperfection, untimely and uncertainty characteristics of context data which are unknown, ambiguous, imprecise and/or erroneous. Moreover, applications should be able to use the context in an intelligent way by using a capability of mapping of lower level context into higher level. It is, therefore, essential for the raw context data obtained to be transformed into meaningful information. It can later be used by applications for decision making mechanism either when there is a presence of certain

changes detected in the context or not. Taking this into consideration, support for context reasoning should be among the issues modeling approaches should consider.

Context Aware Models

Context aware modeling approach is required to design model of context to support context reasoning and other requirements of context data manipulation. Modeling approaches to model context are presented in different researches [40, 61, 62] based on the data structures used for representing and exchanging contextual information in the respective system. Strang and Linnhoff-Popien in [40] present relevant context modeling approaches and evaluated the approaches with requirements. The requirements include distributed composition, partial validation, richness and quality of information, incompleteness and ambiguity, level of formality, and applicability to existing environments. The context modeling approaches investigated are discussed below in this section.

2.4.1 Key-Value Models

Key-Value models are frequently used and represent the most simple data structure for context modeling which employ matching algorithms using key-value pairs for service discovery. Key-value pairs are easy to manage but lack capabilities for sophisticated structuring for enabling efficient context retrieval algorithms.

The applicability to existing ubiquitous computing environments is the strength of key-value modeling approaches. Its simplicity is an advance from the management and error risk perspective. However, it is weak on other requirements and has drawbacks if quality meta-information or ambiguity shall be considered. Distributed composition and the handling of incompleteness are possible only at instance level. Partial validation is also difficult since there is no scheme or at least range definitions available.

2.4.2 Markup Scheme Models

Markup scheme models use a hierarchical data structure consisting of markup tags with attributes and content which is usually recursively defined by other markup tags.

The strengths of markup scheme context modeling approach are partial validation and applicability to existing markup-centric infrastructures of ubiquitous computing environment. Scheme definition and a set of validation tools exists which enable type checking for complex types and range checking to some degree for numerical values. However, it doesn't satisfy the requirements such as incompleteness and ambiguity, and quality of information while distributed composition is addressed to some level. A comprehensive scheme definition is a step towards a high level formality and thus may be used to determine interoperability.

2.4.3 Graphical Models

Graphical models use a modeling instrument called Unified Modeling Language (UML) which has a strong graphical component to model context since it is appropriate due to its generic structure.

Applicability requirement is addressed well in graphical models due to its strength on the structure level which used to describe structure of context knowledge and to drive a model. Partial validation and distributed composition are possible with the latter one having some constraints at structure level where as level of computer evaluable formality is relatively low.

2.4.4 Object Oriented Models

Object oriented models use object oriented techniques. It incorporates advantage of object orientation comprising encapsulation, re-usability and inheritance to cover parts of the problems arising from the dynamics of the context in ubiquitous environments. The details of context processing is encapsulated on an object level and hence hidden to other components. Access to contextual information is provided through specified interfaces only.

Object oriented context modeling approach is strong regarding the distributed composition requirement. New types of contextual information as well as new or updated instance can be handled in systems in a distributed fashion. Partial validation, and handling incompleteness and ambiguity are possible. A higher level of formality can be reached but with few application drawbacks. Applicability to existing object-oriented ubiquitous computing runtime environment

is given with some requirements which often cannot be fulfilled in ubiquitous computing systems.

2.4.5 Logic Based Models

Logic based models use facts, expressions and rules in a high degree formality to define a context model. A formal system is applied to describe the conditions in a set of rules. Inference (also called reasoning) process can be used to derive new facts based on existing rules in the systems. Usually contextual information is added to, updated in and deleted from a logic based system in terms of facts or inferred rules in the system.

Logic based context models have distributed composition as their strength, but partial validation is difficult to maintain. This makes the specification of contextual knowledge very error-prone though the level of formality is high. Quality of information, and incompleteness and ambiguity are not addressed while applicability to existing ubiquitous computing environments seems to remain as a major issue.

2.4.6 Ontology Based Models

Ontology based models represent contextual information using description of concepts and relationships with high and formal expressiveness and reasoning techniques. Concepts and their interrelations can be defined using ontologies. It is suitable to project parts of the information describing our daily life onto a data structure utilizable by computers.

Ontology based context modeling approaches are strong regarding the distributed composition requirement. Incompleteness and ambiguity are also addressed. Partial validation is possible, and a comprehensive set of validation tools do exist. All ontology based context models inherit the strengths in the field of normalization and formality from ontologies.

Compared to other related models, ontology based models provide the most promising assets for context models with regard to the requirements set. It satisfies the requirements better than the other context modeling approaches.

Fact based models, such as object-role and spatial context models, in relation to ontological modeling approach and hybrid approach are discussed in [61]. Timeliness and requirements to model contextual information used in [62] are also taken into consideration which appear different from terms in [61] to evaluate these modeling approaches. Also discussed is object-role approach that originated to fulfill requirements not satisfied under attribute-value pairs when context information is more sophisticated. These are to support query processing, and also to provide modeling constructs suitable for use in software engineering tasks such as analysis and design.

The other types of models are spatial context models that organize their context information by physical location of real-world entities. These models include location of the sensor that measures the context information, or, for non-physical context information, an associated location as metaphor. Spatial models provide efficient procedures for the execution of typical spatial queries. Uncertainty of actual location readings is not always achieved while interoperability can be obtained only by adopting expressive languages if more complex spatial domains are to be modeled.

The studies discussed have shown that ontological models have clear advantages regarding support for interoperability, heterogeneity, and representation of complex relationships and dependencies among context data. It's also claimed that expressiveness and complexity can be achieved in a better direction if ontological models can try to integrate different models and different types of reasoning in order to obtain more flexible and general systems. It seems to suggest that different models and reasoning tools need to be integrated with each other to come up with a better result. However, it still needs a proper analysis on whether this fulfills the different requirements of context models and a further test is required on existing and new projects to check on possible advantages and shortcomings.

The survey made on modeling approaches discussed in [40,61,62] with regard to requirements of contextual information confirm that ontology based models fulfill important requirements better than the other listed approaches to provide a well-designed context model.

2.5 Ontology

The term ―ontology" comes from the field of philosophy that is concerned with the study of being or existence. It has been adopted by early AI researchers to refer to both a theory of a modeled world and a component of knowledge systems.

Gruber [63] defines ontology as an ―explicit specification of a conceptualization," which is, in turn, ―the objects, concepts, and other entities that are presumed to exist in some area of interest and the relationships that hold among them." Ontology defines (specifies) the concepts, relationships, and other distinctions that are relevant for modeling a domain. The specification takes the form of the definitions of representational vocabulary, which provide meanings for the vocabulary and formal constraints on its coherent use. It is also used to reason about the properties of that domain and may be used to define the domain itself. For this reason, it is a formal representation of set of concepts within a domain and relationships between those concepts. Common components of ontologies include individuals, classes, attributes, relations, function terms, restrictions, rules axioms and events.

Ontology can be used in almost every discipline and is also part of the W3C standards stack for the semantic web. It is used to specify standard conceptual vocabularies to exchange data among systems, provide services for answering quires, publish reusable knowledge bases, and offer services to facilitate interoperability across multiple, heterogeneous systems and databases.

2.5.1 Ontology Language

An ontology language is a formal language used to encode the ontology. There are a number of such languages which are developed earlier such as CycL[3], DOGMA[4], Gellish[5], IDEF5[6], KIF[7], RIF[8], LOOM[9], OCML[10], and FLogic[11]. The success of the Internet, afterwards, led to the

[3] CycL - http://www.cyc.com
[4] DOGMA - http://www.starlab.vub.ac.be/website
[5] Gellish - http://ieeexplore.ieee.org
[6] IDE5 - http://www.idef.com/IDEF5.html
[7] KIF - http://www.ksl.stanford.edu/knowledge-sharing/kif
[8] RIF - (http://www.w3.org/2005/rules/wiki/Overview). It's an XML language for expressing rules
[9] LOOM - http://www.isi.edu/isd/LOOM/
[10] OCML - http://technologies.kmi.open.ac.uk/ocml//
[11] FLogic - provides a logical foundation for frame-based and object-oriented languages for data and knowledge representation.

creation of ontology languages that exploited the characteristics of the web and are usually referred to as web based ontology languages. SHOE was built as an extension of HTML before XML was created and widely adopted as a standard language for exchanging information on the web. SHOE, XOL and OML are created based on XML syntax whereas RDF and RDF Schema are languages created by W3C working groups contributing to the foundation of semantic web.

XML provides a surface syntax for structured documents, but imposes no semantic constraints on the meaning of these documents. XML Schema extends XML with data types and restricts the structure of XML documents. RDF, a semantic network based language to describe resources, is a data model for objects and relations which provides a simple semantics for the data model. RDF Schema, extension of RDF with frame-based primitives, is a vocabulary for describing properties and classes of RDF resources, embedding semantics for generalization-hierarchies for such properties and classes. The combination of both RDF and RDF Schema, also known as RDF(S), is not very expressive, just allowing the representation of concepts, concept taxonomies and binary relations. Three more languages are also developed by W3C working groups as extensions to RDF(s) which are OIL, DAML+OIL and OWL. OIL adds frame-based KR primitives to RDF(S), and its formal semantics is based on DLs. OWL is a revision of the DAML+OIL web ontology language.

OWL is a semantic web language designed to represent rich and complex knowledge about things, groups of things, and relations between things. It is a computational logic based language such that knowledge expressed in OWL can be exploited by computer programs. It is intended to be used when the information contained in documents needs to be processed by applications. It can be used to explicitly represent the meaning of terms in vocabularies and relationships between those terms. It becomes widely used ontology language becoming W3C's recommendation, because of its better facilities for expressing meaning and semantics than XML, RDF, and RDF(S), and due to its ability to represent machine interpretable content on the web. It has three increasingly expressive sublanguages: OWL Lite, OWL DL and OWL Full.

OWL Lite is for classification hierarchy and simple constraints providing a quick migration path for thesauri and other taxonomies, whereas OWL-DL provides maximum expressiveness retaining computational completeness and decidability. It is easier to provide tool support for OWL Lite which has lower formal complexity than OWL-DL. Unlike OWL Lite, OWL DL

includes all OWL language constructs but used only under certain restrictions. OWL Full, on the other hand, provides maximum expressiveness along with the syntactic freedom of RDF with no computational guarantees. The choice between OWL DL and OWL Full is then mainly depends on the extent to which users require the meta-modeling facilities of RDF Schema. Reasoning support is also less predictable when using OWL FULL than OWL DL.

2.5.2 Ontology Development Tool

Ontology development is a complex and largely domain oriented process that can benefit from tool support. The Ontolingua Server[12], Ontosaurus[13] and WebOnto[14] were among the pioneer ontology tools, but they were designed for Ontolingua, LOOM and OCML specific languages, respectively, and didn't provide facilities for extensibility.

Other ontology development tools such as Protègè[15], WebODE[16], OntoEdit[17], Top Braid Composer[18], OilED[19], Internet Business Logic[20], OntoTrack[21], and IHMC Cmap Ontology Editor[22] have been created to integrate ontology technology in actual information systems and are labeled in [64] as the most used models. They have extensible, component based architectures, where new modules can easily be added to provide more functionality to the environment. The survey also suggests Protègè as the widely used tool to model ontologies. It is an open source, standalone application with an extensible architecture. The core of this environment is the ontology editor and its capability to hold a library of plugins that add more functionality to the environment.

[12] Ontolingua Server - http://www.ksl.stanford.edu/software/ontolingua/
[13] Ontosaurus - http://www.isi.edu/isd/ontosaurus.html
[14] WebOnto - http://kmi.open.ac.uk/projects/webonto/
[15] Protégé - http://protege.stanford.edu/8
[16] WebODE - http://delicias.dia.fi.upm.es/webODE/
[17] OntoEdit - http://www.ontoprise.com
[18] Top Braid Composer - http://www.topbraidcomposer.com
[19] OiLED - http://oiled.man.ac.uk/
[20] Internet Business Logic - http://www.semanticweb.org/wiki/Internet_Business_Logic
[21] OntoTrack - http://www.informatik.uni-ulm.de/ki/OntoTrack/
[22] IHMC Cmap Ontology Editor - http://www.ihmc.us

CHAPTER THREE

RELATED WORK

In this Chapter, previously conducted researches in areas related to providing architectural level support for building context-aware computing are presented and analysis on various proposed architectures is made. Also, domain specific works in university settings and other domains which are related and relevant in terms of their objectives are discussed.

3.1 Context Aware Architectures

Context aware architectures have evolved over the years from location aware systems to present context aware systems. Many works have been suggested since their early development, which based their focus mainly on location aware services. Among the good efforts are Active Badge [34], ParcTab [65], Stick-e-notes [66], Cyberguide [67] and Guide [68], which are specific to localization systems and cannot be easily used for other kinds of context-aware systems. They also could not satisfy the required level of context abstraction, extensibility and reasoning support [69] in context management.

Later systems tend to provide architectures that can support implementing a wider range of context aware applications in many domains. One of early attempts of framework to support various domains in building context aware applications and aims to provide reusable software component is Context Toolkit [70] proposed by Dey and Abowd. Context Toolkit, shown in Fig. 3.1, consists of context widgets, interpreters and aggregators which are used to encapsulate the complexity of actual sensors from applications forming reusable software component, to abstract low level sensed information into high level information and to compose context information, respectively. Context Toolkit provides a basic access control for privacy protection of users by providing context ownership concept. However, it has limited extensibility when the number of devices increases.

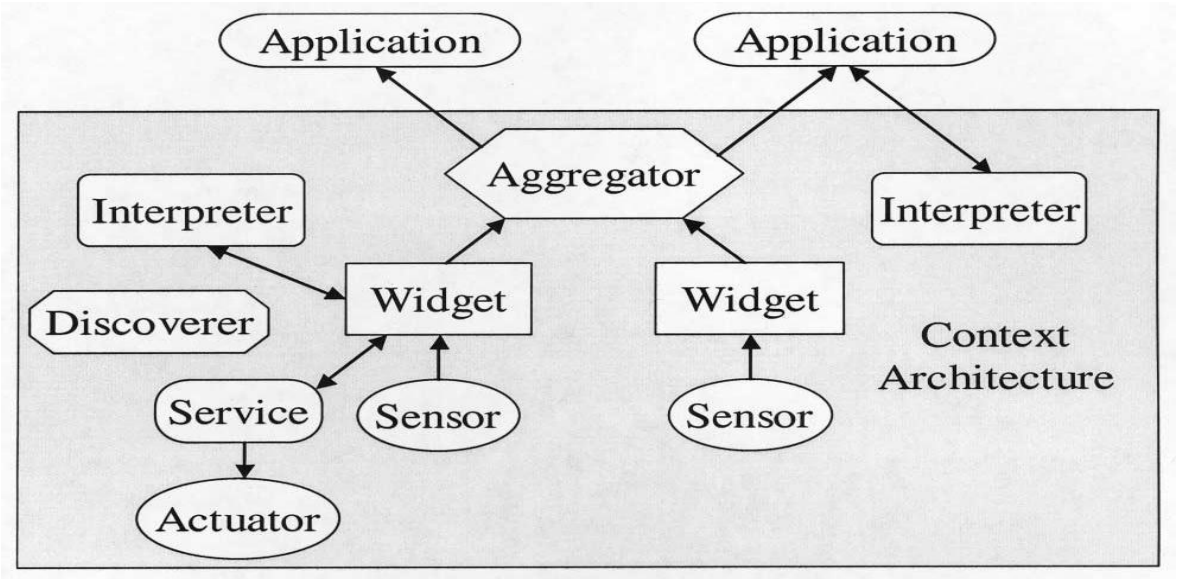


Fig. 3.1: Example Configuration of Context Toolkit Components

Similarly, Hoffer proposed another early open and extensible architecture and framework called Hydrogen [71] to support various types of context information. The architecture, as shown in Fig. 3.2, has adaptor, management and application layers. The context server in the management layer contains all the sensed information perceived by the sensors of the adaptor layer and provides context to the application layer of the attached device. It differentiates between a remote and local context and allows context sharing among devices but does not support resource discovery and persistent storage functionality.

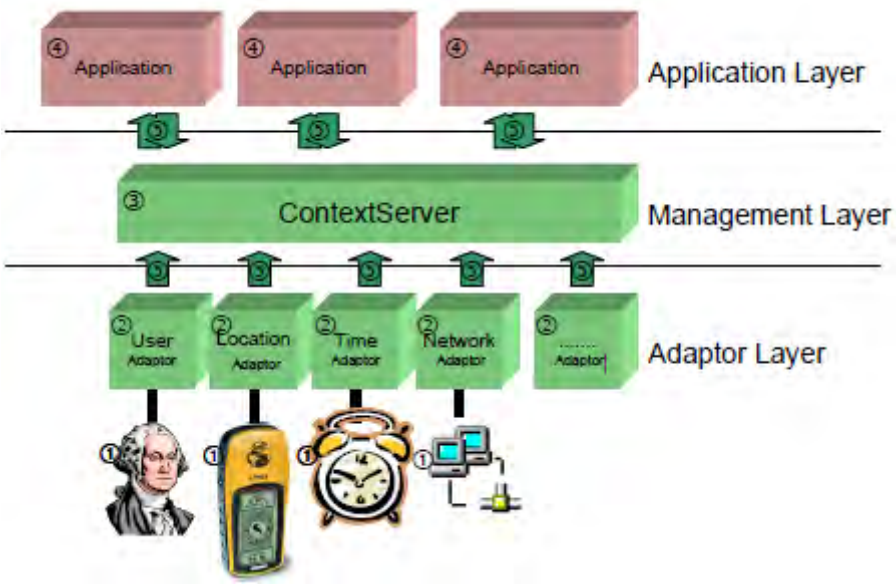


Fig. 3.2: Hydrogen Architecture

Both Context Toolkit and Hydrogen are distributed architectures that aim at providing reusable software component; however, the former uses a simple attribute-value tuple as context model while the later uses object oriented context data model. As a result, both architectures can be easily implemented but do not provide good reasoning module on context.

Gaia [72], CASS [73] and CORTEX [74] are also other suggested works for context management. Gaia is similar to traditional operating system providing several kernel services in the system to support context aware applications in active spaces. It is an agent based ubiquitous software platform project which aims at providing user-centric, resource-aware, multi-device, context-sensitive mobile services. It consists of core services and a framework for building context-aware applications and represents context data using 4-ray predicates in DAML+OIL as its context model. However, it does not focus on scalability and privacy protection. CASS, shown in Fig. 3.3, is another middleware system for supporting the development of context-aware applications. It aims to provide context data abstraction and separate context based inferences and application behaviors. It uses an object oriented model for context description and database table as its context model. The architecture is based on a server containing a database of contextual information and a knowledge base with an inference engine to infer other contextual information using a back chaining mechanism. Its good context abstraction level, extensibility and reasoning mechanism makes it more proactive. However, its centralized architecture holding the central server to manage the functionality of the interpreter, context retriever, rule engine, and sensor listener is its weakness.

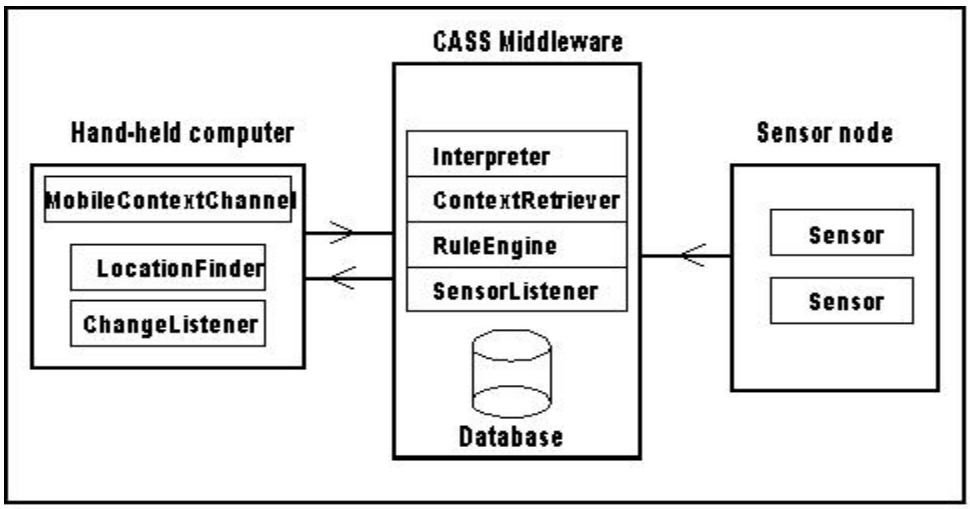


Fig. 3.3: CASS Architecture

CORTEX is also another framework to ease the development of context aware mobile applications. As shown in Fig. 3.4, it is based on sentient object with its core architecture composed of modules for fusion and interpretation of context, hierarchical representation of context and an inference engine. The communication between sentient objects, sensors and actuators that compose the system uses a mechanism based on events which are established dynamically during the system operation. However, inadequate contextual information, lack of resource discovery mechanism to measure extensibility and its less usability due to its inference engine language are considered as its weaknesses.

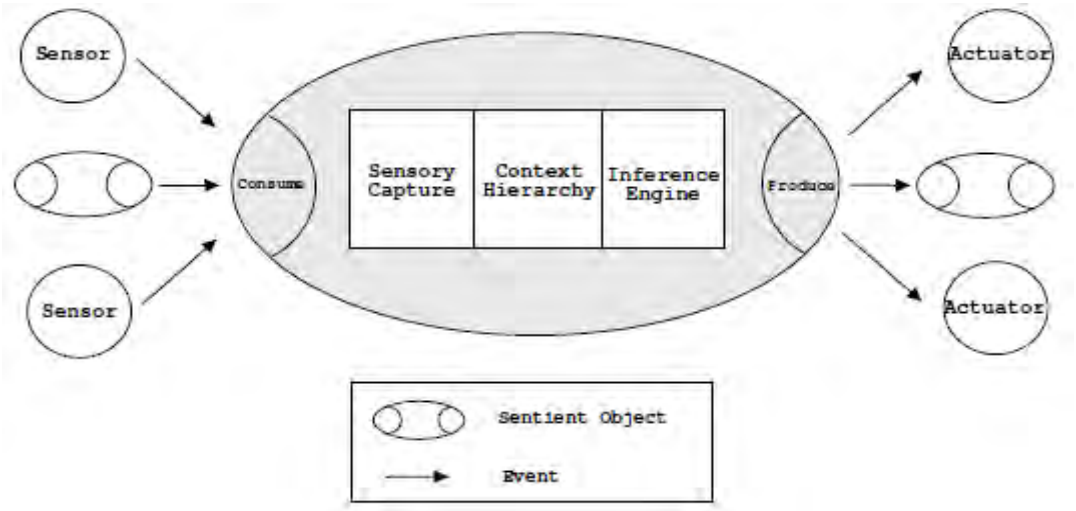


Fig. 3.4: Sentient Object Model

Ontological context model has then become popular since it is suitable for expressing concepts in a more meaningful way. CMF [75] is a client-server architecture which uses

ontology as its context model and delivers contextual information to applications by using a communication model based on events. As shown in Fig. 3.5, it is composed of context manager for storage and delivery of context, resource server for acquisition of contextual information and change detection service for detection of change in context and service. It enhances reusability and provides context abstraction and security mechanism. However, it does not offer a reasoning module though it allows semantic context reasoning. Context Fabric [76] proposed by Hong is another context aware system architecture which considers ontology as its context model and mainly focused on providing protection of privacy sensitive context aware applications. It provides privacy support by letting users define privacy tags for each context tuple.

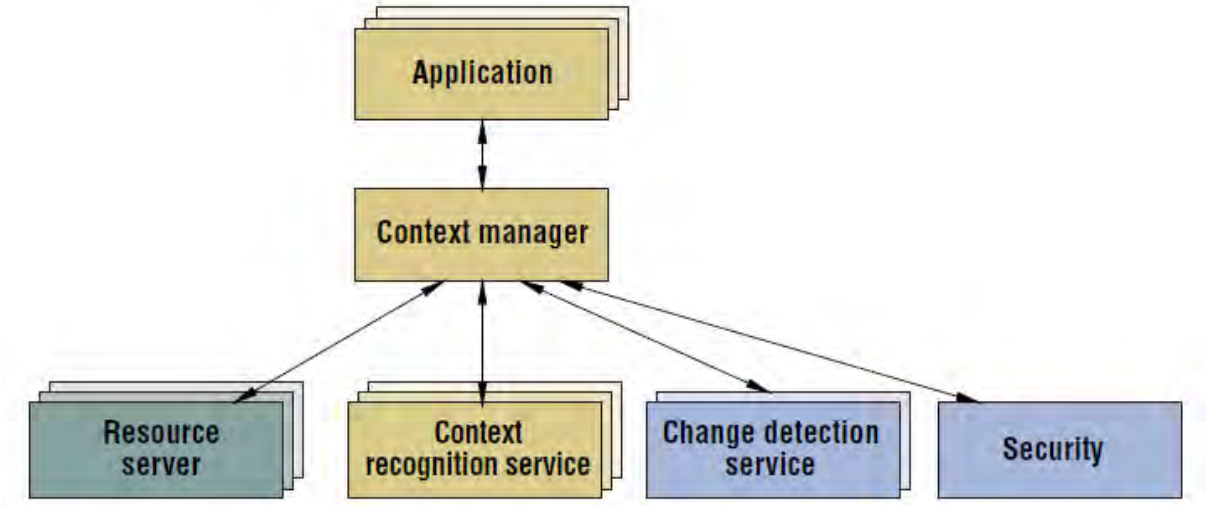


Fig. 3.5: Context Management Framework

Following the announcement of OWL by W3C, many more context aware systems adopted ontology as their context model due to its flexibility, expressiveness and suitability for knowledge sharing. Among promising architectural works to take these advantages are CoBrA [77], SOCAM [78] and enhanced CoCA [79].

CoBrA shown in Fig. 3.6 is an agent based context architecture to support context aware applications in intelligent spaces and uses OWL to support context modeling and knowledge sharing. It has main components: context base, context inference engine, context acquisition module and privacy management module. It uses a context knowledge base and its own OWL-based ontology (CoBrA-Ont) to manage ontological context information of entities. Context acquisition module acquires context and its reasoning is handled in the inference

engine, while privacy issues considered in privacy management module. Its centralized resource rich agent called context broker manages and maintains a shared model of context and controls context aware communication between agents and resources. It is up to the context broker to acquire and reason about contextual information, detect and resolve inconsistent context knowledge, and protect user's privacy. For privacy protection, CoBrA provides a policy language that allows users to define their policies in order for the context broker to control the sharing and use of their contextual information. To resolve congestion and other bottlenecks associated with a single centralized server, it uses a hierarchical approach providing a broker associated with a given space, which can also be subdivided into small granularities with individual brokers. However, it does not consider a resource discovery mechanism and depends on the assumption that there exists a context-broker server that is known by all the participants.

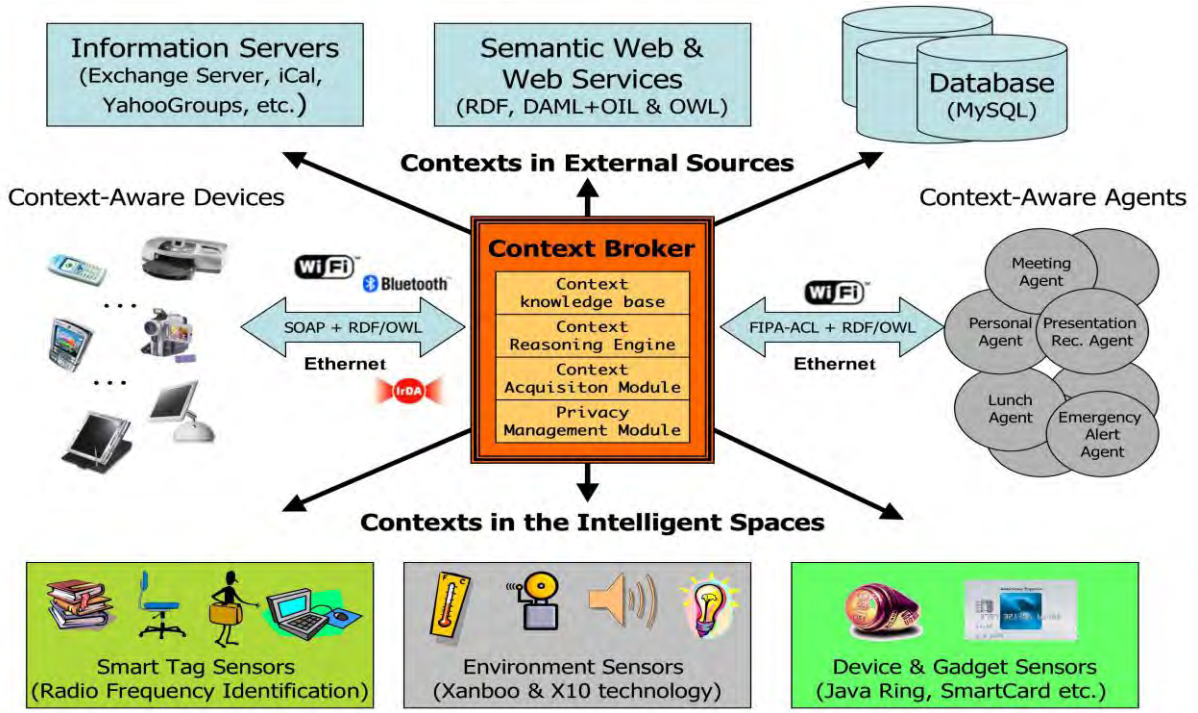


Fig. 3.6: CoBrA Architecture

Likewise, SOCAM aims to build a middleware for context aware services and uses OWL to model its context composed of generic and domain-specific ontologies. SOCAM, presented in Fig. 3.7, has a centralized architecture for building and rapid prototyping of context aware services. The architecture is composed of context provider, context interpreter, context knowledge base, context reasoner, context aware service, service locating service and context

database. Context providers acquire context information from various internal and external sources and convert them into ontological representation. Context reasoner in the context interpreter performs reasoning to preprocess context information and maintain its consistency. So, the context interpreter collects and reason about context information obtained from context providers (internal or external) and context database, and provides them to context aware service and service locating service modules. Service locating service is a discovery mechanism where context providers and the context interpreter can advertise their presence.

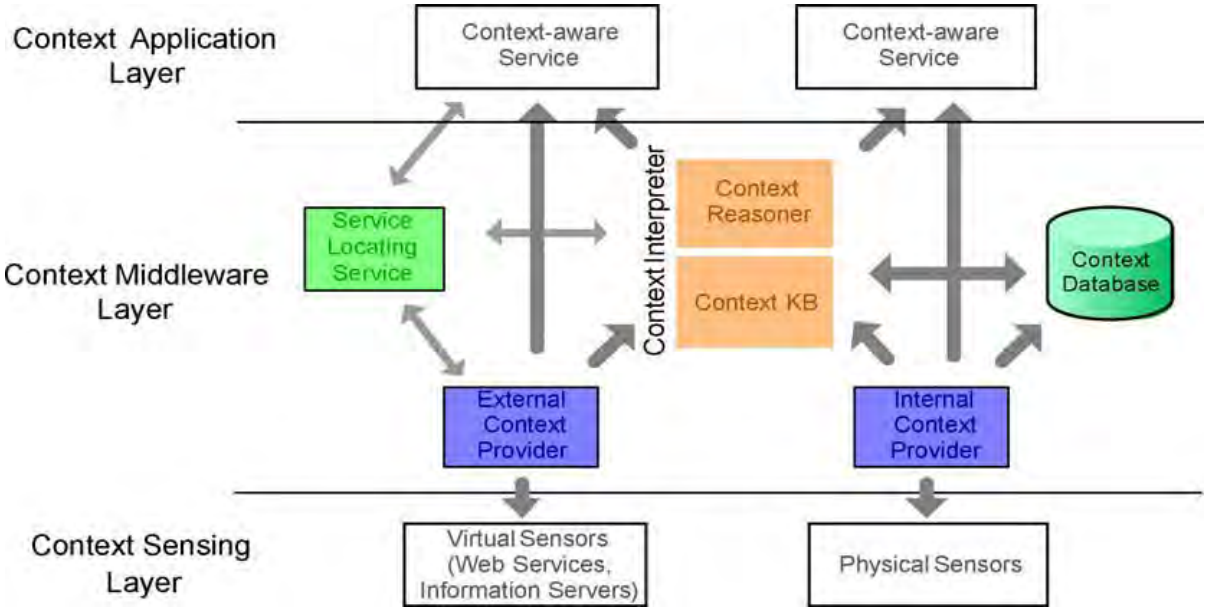


Fig. 3.7: Overview of the SOCAM Architecture

Enhanced CoCA differs from other works in that it is based on hybrid context management model that combines ontology and relational database approaches to achieve acceptable performance of the system. It incorporates modules, as shown in Fig. 3.8, to represent context data and to perform reasoning, aggregation, and interpretation on context data. It triggers actions and decisions based on the context data and supports neighborhood collaboration for sharing computational resources among devices to perform expensive processes. To overcome problem of processing computationally intensive high-level inferences, enhanced CoCA provides a way of representing context using EHRAM (Entity, Hierarchy, Relation, Axiom, Metadata) model and uses hybrid context management (HCoM) to manage context data. It uses semantic ontology approach to manage context semantics and relational database approach to manage context data, and its processing takes place separately. The results are then combined to enhance reasoning and decision support.

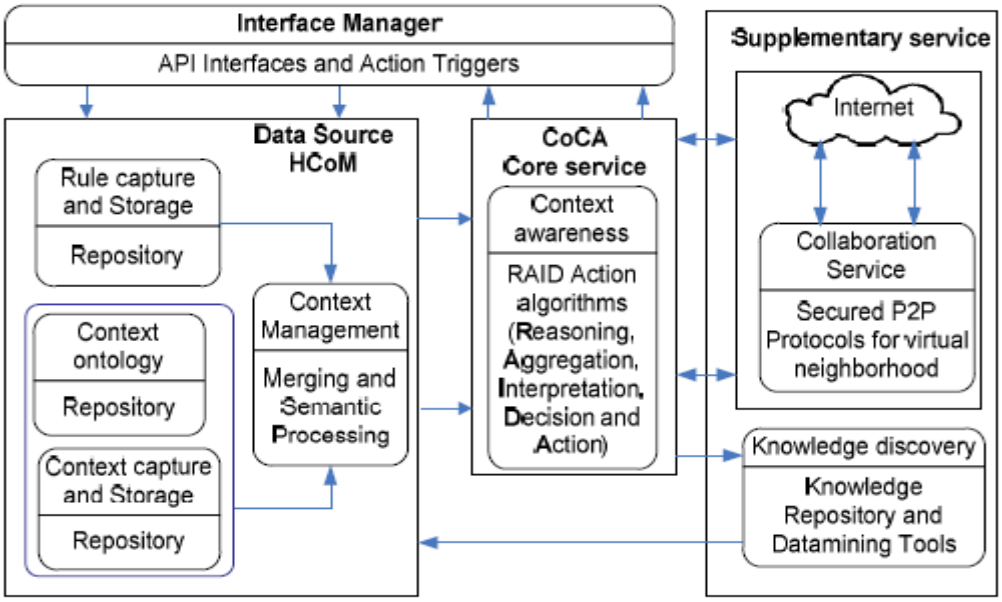


Fig. 3.8: Architecture for the CoCA Platform

3.2 Works related to university campus environment

The UCSD ActiveCampus project [80] is one of the efforts in a university setting which is an exploration of wireless location-aware computing in the university setting. It aims to fulfill campus needs by employing context aware ubiquitous computing to application running on existing infrastructure and handheld devices. It provides two applications to offer services related to classroom activities and location aware learning mechanism in the process of communication between students outside classroom. ActiveClass is a simple client-server application designed to encourage student participation in large classroom settings. It considers help of personal, mobile wireless computing devices for interaction between students and teachers. ActiveCampus Explorer is another application designed to help students and faculty members take advantage of unseen opportunities for learning and interaction on campus. This includes nearby activities or meeting with a colleague by detecting location using handheld devices. However, it doesn't focus on various services offered in university setting and a proper mechanism to handle contextual information around the campus and to distinguish between low level context data from its high level interpretation were not addressed.

A research on Semantic Data Access (SDA) [81] at Bell Labs also considers university setting on providing application to facilitate the working process in the domain. Its focus is mainly on implementing a Smart Campus application prototype that integrates heterogeneous

data using semantic technologies. It is based on a lightweight semantic architecture that facilitates semantic data access and integration using semantic languages and tools. Its architecture provides Semantic Box (SBox), Access Box (ABox) and External Interface layers. SBox layer is for management, query and reasoning functionalities while ABox layer is for conversion of raw data to RDF. External Interface layer facilitates communication between the other layers and support for programming functionality. Acquisition of context, context abstraction and aggregation and reasoning services are not well addressed to the required level of importance.

Another considerably related work done in terms of its application area in a university setting is RFID-based Campus Context-Aware Notification System (R-CCANS) [82]. It represents the design and development of a context aware notification system using RFID technology motivated by observation of common existing ways of campus notification channels, E-learning and bulletin board. The system uses RFID technology to trigger and acquire context information and uses available screen monitors at the various locations in the campus to display the notifications. However, it provides a domain specific platform, as shown in Fig. 3.9, which does not consider any of the proposed generic architectures and context modeling approaches. It uses database to model and store context information and its context model tries to consider location, identity and time contexts but lacks proper handling of context abstraction and reasoning.

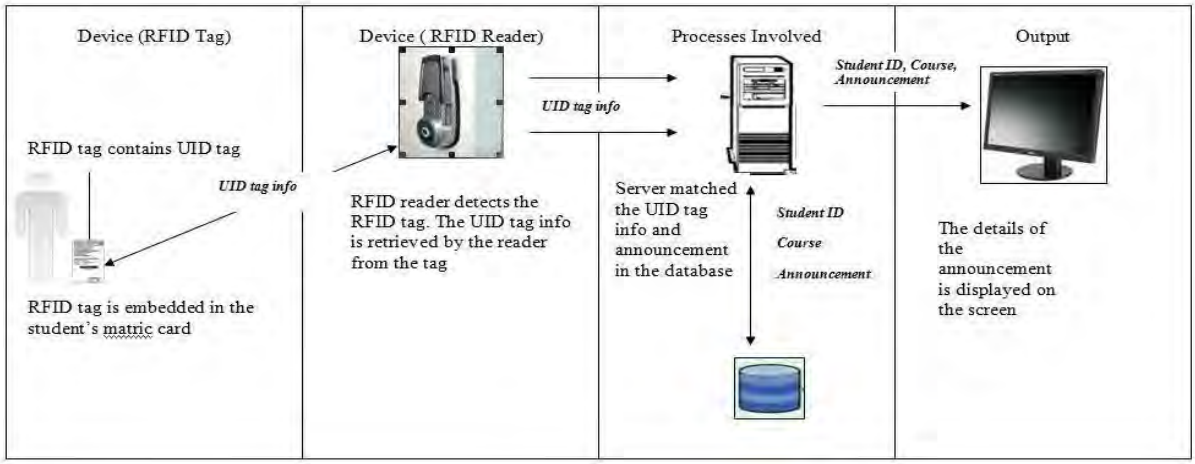


Fig. 3.9: R-CCANS Architecture Platform

3.3 Related works in other domains

With the widespread of applications in pervasive computing, various domains are automating their services with its support of context awareness. A number of potential projects have been applying its concept in different application areas. However, in this section, works that are related in terms of context management to assess and take advantage of contextual information handling and manipulation are selected and discussed despite a concern of reusability in other domains.

The first domain that is investigated in this study is a work related in terms of context management which is done in the area of robotics. An ontology based context aware system to control the robot manipulator in delivery logistic industry aiming at providing an automatic smart robot manipulators for generating information to control the movement of a robot as shown in Fig. 3.10 is discussed in [83]. It defines domain ontology and constructs an ontology based context model. It also provides domain specific architecture for context management comprising context acquisition module, knowledge base, rule manager and context manager. The context acquisition module collects context from various sensors and stores in the context information database after a process of abstraction and generalization is done in the context collector. The context knowledge base provides rule database to store rules provided by rule manager to perform reasoning. The context manager analyzes the context by managing and processing the rules thus obtained. It then recognizes the context by relational database ontology to deliver context information and to store in the knowledge base.

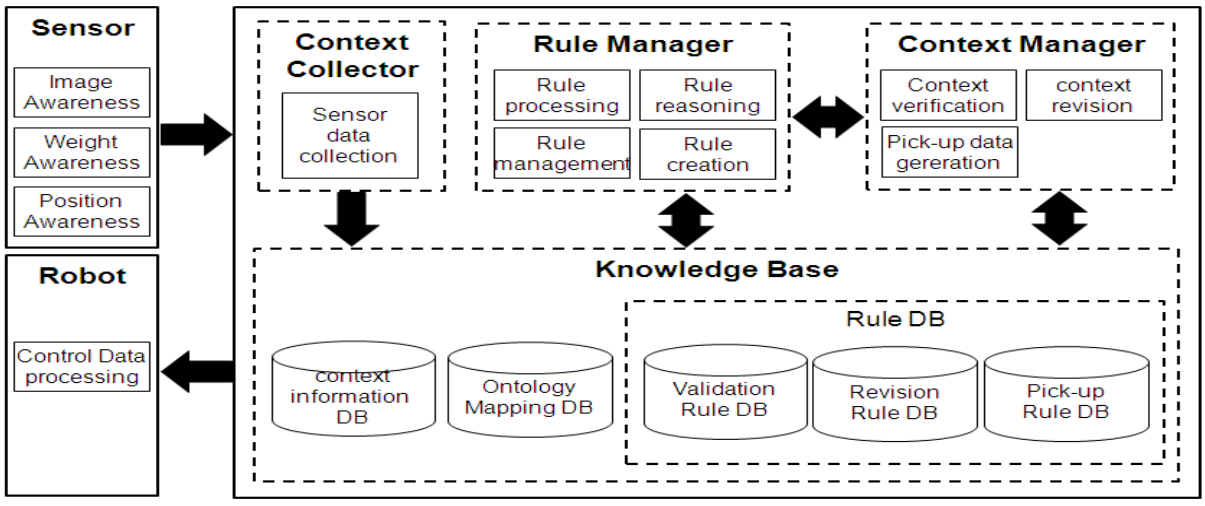


Fig. 3.10: Architecture for context management in controlling robot manipulation

The other context management domain selected to assess its delivery is an intelligent transportation system which focuses on providing context aware vehicle service. Ontology based context modeling for vehicle context aware services focusing on building an ontology modeling approach for context management in intelligent pervasive middleware for context aware vehicle services is given in [84]. Its architecture, as shown in Fig. 3.11, provides modules for context acquisition and fusion, context assimilation, context interpretation, inferring rules, and storing contexts. Context acquisition and fusion module is used to acquire contexts from sensors, abstract them and convert them into OWL representation. Context information is further organized in the context assimilation module to make them available for interpretation. The pre-history context is stored in context database and checked in relation to rules provided in the inference rule module. The reasoning engine and context knowledge base in the context interpretation module are responsible for interpretation of context information. The processed context information is then finally made available for access to domain specific services.

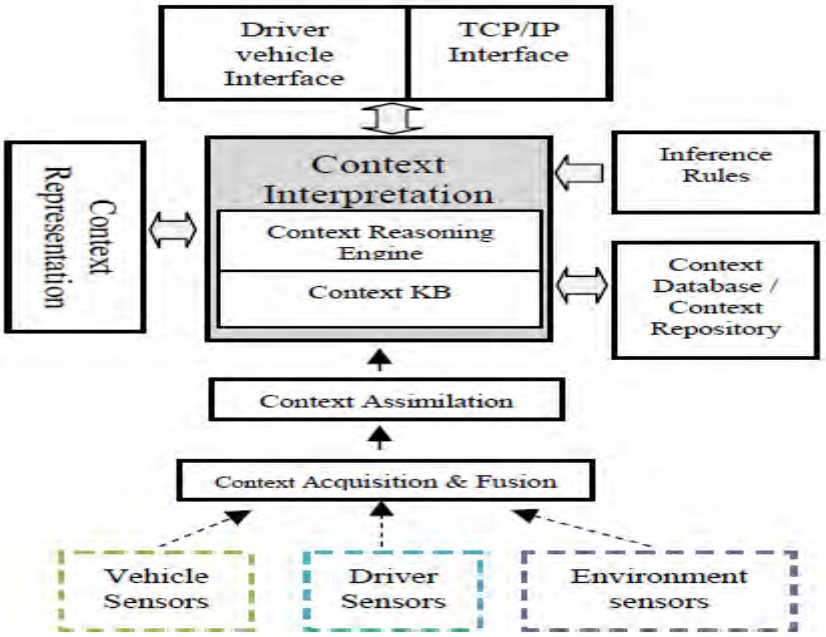


Fig. 3.11: Architecture for context aware vehicle service

3.4 Summary

The works provided in review of context-aware architectures in section 3.1 above are benchmarks in the process of building architectural support in context aware systems. Earlier works presented, however, lack on providing semantically rich and expressive models and a

support for a higher degree of reasoning. Efforts were made in order to bring re-usable platforms to support applicability in other domains but were limited in their component functionalities and representation models.

Most of later proposed architectures make distinction between context sensing processes from its use thus permitting an abstraction of low level sensing details and increases the extensibility and reusability of architecture components. Performance, scalability, delivery of quality service and robustness in addition to standard, reusability and context management are becoming issues of context aware systems as their development and use progress. CoBrA, SOCAM and enhanced CoCA use ontological context model to take advantage of ontology reasoning engines. Although they are similar in using ontological context model, they all differ in the approaches they use. CoBrA defines its own general ontology while SOCAM separates its context ontology into common upper ontology layer to represent general concepts and domain-specific ontology layer to represent sub-domain concepts. Enhanced CoCA on the other hand considers relational database in addition to ontology for context data management. Hence, SOCAM and enhanced CoCA deliver a mechanism for enhancing system performance and to support building context aware services to some level. However, architectural support still requires service centric context aware system management considering resource management along with enhanced security mechanisms. Consideration for historical context information is required to avoid redundant processing of context data. An enhanced context management is also required to cope with the dynamically changing nature of context data.

The domain specific proposed works in robotics and transportation system also deliver a way of context management. They are aimed at providing pervasive knowledge base systems to embed semantically rich and easily accessible information into the physical world. They provide context abstraction and interpretation with a support of context reasoning using rules defined in them. However, besides their domain-specific nature that render them unsuited for reusability and standard, they lack in utilizing virtual context information and other resources rich of context, issue of performance, and privacy.

In spite of the aforementioned promising efforts, there is still a lot to do to ensure necessary requirements of pervasive system that considers existence of various entities, services and their collaboration with the frequent change of context in pervasive environment. This work therefore, focuses on providing a proper way of resource discovery, context management,

security and context-aware service delivery. Also considered are handling of context data sources, historical context information with additional learning mechanism, and handling of context change preserving usability, re-usability, performance, scalability and security.

Moreover, works presented in terms of applicability in university setting were not addressed with the required level of importance in pervasive computing. Ways of handling contextual information around the campus with respect to service deliveries were also not considered. Previous works discussed under section 3.2 didn't consider context acquisition, aggregation, abstraction and reasoning advantages. Therefore, this work also focuses on embedding such functionalities to service deliveries using the components of the generic architecture to be analyzed.

CHAPTER FOUR

DESIGN OF CONTEXT AWARE SERVICE MANAGEMENT ARCHITECTURE

This chapter presents the design of context-aware service management. It discusses a proposed model and architecture of context management for the delivery of services that are aware of their environment. It also shows how model of context in smart campus can be represented using the proposed context management model for services that require a proper identification mechanism.

4.1 Overview of Context-Aware Service Management Architecture

Pervasive computing offers a way of providing services that enables users to access and use information. It hides process details in delivering those services to the user with minimal distraction and, when necessary, relies completely on itself to act on behalf of the user to perform tasks. The crucial information which enables systems perform accordingly is obtained from context. To achieve this, context is necessary to provide the crucial information in which systems could act up on to perform accordingly. Context awareness plays a vital role for acquiring the goal of pervasive computing by identifying any relevant information on entities in their environment and relationships among them.

Context-aware systems require a proper acquisition, analysis and management of context to ensure delivery of services with the maximum required intelligence. Context sensing, context management, decision and analysis, service delivery and security can be seen as the overall components in the proposed architecture. It collects raw context data from different sources, converts into logical context information, and provides relevant service according to the given context. Sensing can be done by context data sources comprised of various devices and software related sources. Producing logical context information not only helps systems to simply provide their services to the intended user, but also enables them to give intelligent decision when necessary before delivering those services.

An overview of the proposed architecture is shown in Fig. 4.1. The Context-Aware Service Management Architecture (CASMA) is comprised of components such as context source, resource manager, rule manager, context repository, context manager and context-aware

service management modules. The inputs are collected and managed in the resource manager according to the context. The resource manager identifies resources with context data, assimilates the acquired input data, and sends it to context acquisition module for further processing. Various context manipulations exist in context management module in relation to rules, historical context, any change in context and other context information in repositories to come up with a final context decision. Context-aware service management module makes use of this information to identify required services and provide applications to process the necessary service. Context information identified can be investigated by log analyzer for security purposes. The details of the components of CASMA are provided in section 4.2.

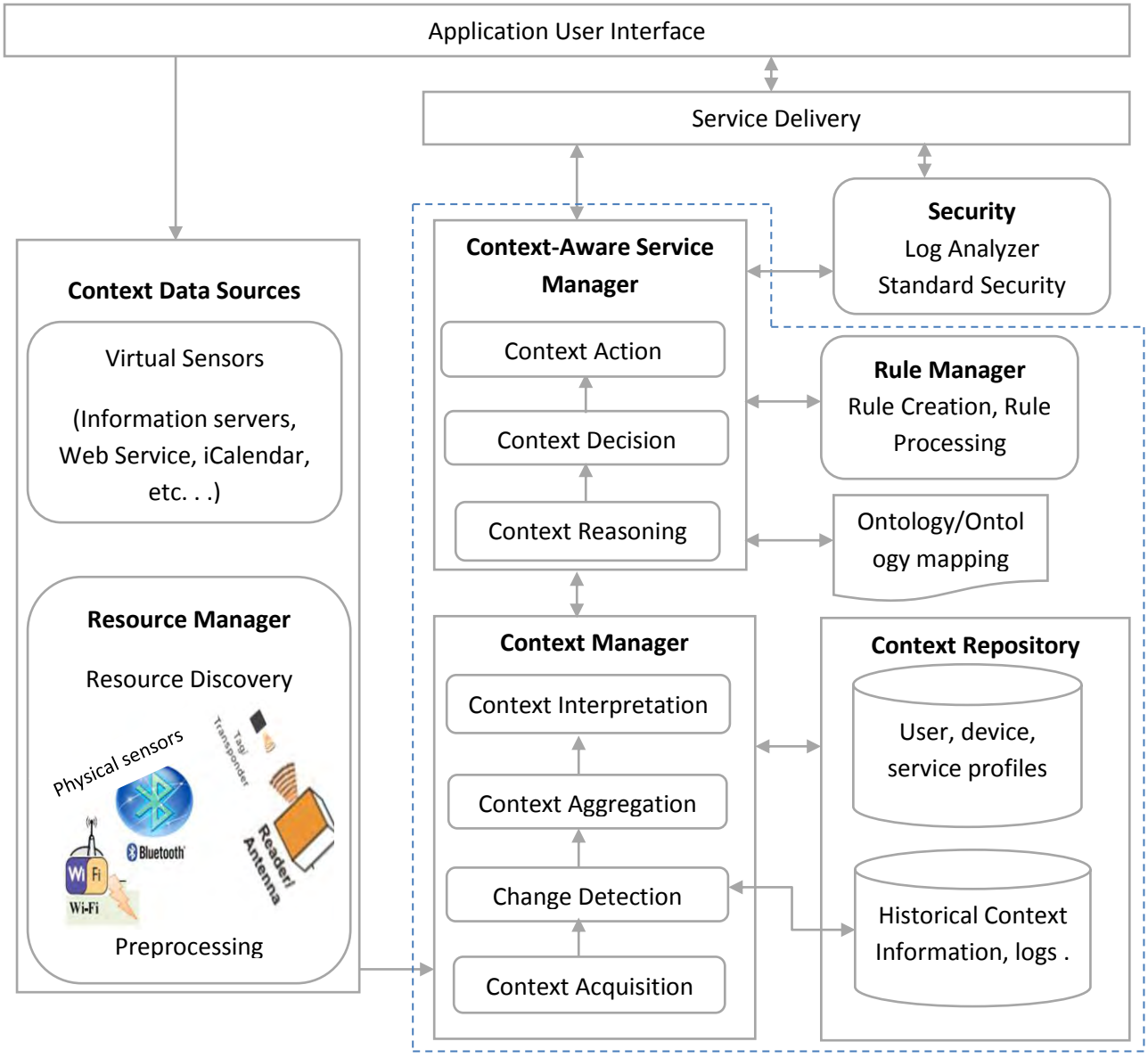


Fig. 4.1: Overview of CASMA

4.2 Major Components of CASMA

The overall process of delivering services in CASMA is provided in this section. A process or action can be triggered by the availability of resources from virtual context and/or inputs arising from users according to their preference. The responsibilities of each individual component of CASMA in relation to other components are also discussed in detail in this section.

4.2.1 Context Sources

CASMA considers any source of context that could deliver relevant information to the process of context-aware services. Logical context sources, virtual sensors and physical devices are regarded as potential context sources. Users can also provide relevant context data when available to enhance context analysis for a more-inclusive decision making. Likewise, virtual sensors such as information servers, web services, electronic calendars, etc., are believed to be other potential sources. Moreover, physical devices distributed over a given environment contribute in creating smart spaces by bringing context data with quantity and quality. Technologies like RFID, WSN and other devices with support of wireless connectivity such as Bluetooth and Wi-Fi are among rich context sources. Their existence in CASMA by itself is a context data and can trigger action. For instance, a tag detected by RFID reader can at least give location, identity and time contexts. Virtual sensors and other related context sources can similarly provide such context data for further analysis.

Resource Manager

With the availability of different resources to capture context, a resource manager is needed to identify those resources along with the context data they deliver. A proper utilization of resources can help capture an adequate context data. Also, the context data acquired should be checked before it is processed in the acquisition module to simplify its manipulation in the reasoning service. Algorithm 4.1 can be applied to process resource discovery, gather context data, preprocess and present to context acquisition module.

- **Resource discovery**

The resource manager is responsible for listening to available devices and discovering them to acquire context data. In addition, some types of devices can advertise their existence to the resource manager.

- **Preprocessing**

After identifying the resources along with the context data, the resource manager checks for the correctness of the acquired data. It checks for erroneous, null and other data related problems detected that could harm next operations in context management. This in turn helps to ease the preprocessing operation that take place in context reasoning module.

```
//Resource Manager

1. BEGIN
2. REPEAT
      a. Listen for resources
      b. IF resource found THEN
           i. Parse context input
          ii. Preprocess
         iii. Return context data
      c. Else
           i. Do nothing
          ii. Wait
      d. END IF
      e. Trigger context-acquisition module in context
         manager
         //see algorithm 4.2
3. UNTIL User suspend process
4. END
```

Algorithm 4.1: Context Resource Manager

4.2.2 Context Manager

Context manager is responsible for context data manipulation before it is presented to context reasoning module in context aware service manager. This involves the collection, representation, combination, interpretation and presentation of context data to context-aware service management operations. It comprises context acquisition, context aggregation and context interpretation modules. Algorithm 4.2 provides the details in processing functionalities of context manager.

Context Acquisition

A context data acquired and preprocessed in the resource manager along with context data from other sources gets its representation in context acquisition module. Context acquisition module deals with assimilation, filtering and representation of the context data thus captured.

Context Change Detection

Context change detection module listens to context data acquired in context acquisition module. It checks for a change in context by relating new context data to a historical context information repository. It helps to simplify context decision by providing the previously analyzed context information if there is no change in context. For any detected context change, it sends the collected context data for further manipulation.

Context Aggregation and Interpretation

Context aggregation module is responsible for amassing context data from context acquisition module. It checks for related context data to process aggregation. It integrates numerous contexts into one to help enable context interpretation. Context interpretation provides a higher level context abstraction from the aggregated raw data. It gives concepts, relationships and properties a higher level of interpretation.

```
//Context Manager

1. Input:
     Context data from logical source
     Context data from resource manager //from
     algorithm 4.1
2. Collect data  //Context acquisition
3. Assimilate data
     a. Represent context
4. Query for historical context information //for
   change detection
     a. IF found equals true THEN
          i. Compare
         ii. IF context information similar THEN
             Trigger Context Action
        iii. Else
             Trigger Context Aggregation
         iv. END IF
     b. END IF
5. Check relationship between collected data
6. Look for Context repository for related context data
7. Aggregate related context data
8. Interpret aggregated raw data
9. Send to Context Reasoning
10. END
```

Algorithm 4.2: Context Manager

4.2.3 Context-Aware Service Manager

Context-Aware service manager is responsible for higher level management of context and mapping of context to service delivery, as depicted in Algorithm 4.3. It performs reasoning on context information to provide context decision and to trigger an action by relating to service. Context reasoning engine provides a reasoning service on the interpreted context

information. It deduces new facts based on context information held in repositories, rules provided and ontology stored. Context decision is made based on combined and reasoned about context. It provides higher level context information that can enable context action to be performed. Context action then is provided to relate context decision with the service to provide and present for context-aware service delivery.

```
//Context-Aware Service Manager

1. INPUT:
          Interpreted context data //from Algorithm 4.2
          Rules
          Ontology
2. Check for new rules
     a. IF new rules THEN
          i. Look for existing rules
         ii. Integrate with existing rules
        iii. Store in rule file
     b. END IF
3. Initiate Context reasoning module
     a. Check consistency in asserted context data
          i. EXCEPTION
             WHEN inconsistency occurs
             Notify
     b. Perform reasoning based on input data
     c. Infer new concept relationship
     d. Return inferred context information
4. Perform context decision
5. Store context decision information
6. Identify service
7. Integrate context-aware service
8. Send action for service delivery
9. END
```

Algorithm 4.3: Context-Aware Service Manager

4.2.4 Service Delivery

The goal of context-aware system is to finally deliver the required service to the intended user with possible ease according to the context analysis and decision made in the context-aware service manager. Applications can have different services included in which each can have its own functionality and require some information to trigger its process.

The context-aware service delivery module in CASMA is thus responsible to identify the services to be delivered. It maps context decision made to retrieve relevant service related data. It checks for availability of necessary services, compare them with the context provided and deliver to the intended user. It can also process some information by itself whenever the context information shows the user is unavailable or not required for the process of the service.

4.2.5 Security

CASMA provides a security module that enables security mechanisms in context aware service delivery. It provides standard security mechanisms, such as authentication and access control, to control how users and systems communicate and interact with other systems and resources. It gives the ability to control, restrict, monitor and protect resource availability, integrity and confidentiality. It also uses log analyzer to prevent and monitor security threats.

Log Analyzer

A log is a record of events occurring within a system. Logs are composed of log entries in which each entry contains information related to a specific event that has occurred. Originally, logs were used primarily for troubleshooting problems, but logs now serve many functions such as optimizing system performance, recording the actions of users, and providing data useful for investigating malicious activity. Logs can be used to prevent security attacks that are in progress or to detect and monitor attacks that are already occurred. Intrusion detection and intrusion prevention can be integrated with the current concept in log management security mechanisms. Log analysis can also be integrated with alarm and related warning to prevent malicious activity in progress.

CASMA provides log analyzer component for security purpose in pervasive access to services. Log analyzer in CASMA is responsible for security log management, which is the process for generating, transmitting, storing, analyzing and disposing security log data. It

receives context decision information, analyzes required log information and stores to log file for later monitoring. For instance, a device tag used for user identification can be represented along with service access information. Algorithm 4.4 shows processes for such related passive log management cases. The log analyzer is also responsible to respond to unorthodox behavioral patterns to prevent malicious activity in progress. It also uses hashing to secure the log file from unauthorized access and unknown modification.

```
//Log Analyzer

1. INPUT:
          Context decision information
          Service access parameters
2. Open the log file
3. Open file (filename.filetype)
4. Write file
     a. IF userTagID value used and accessparameters is
        enabled THEN
        //or any certain security condition provided
          i. Write access information to file
        //includes timestamp, user information and service type
         ii. Sort //based on user preference
     b. END IF
5. Read file (filename, buffer) // to read log file
   information
     a. Read line by line
     b. IF userTagID match accessparameter THEN
          i. retrieve log information
     c. END IF
6. Hash the log file
7. END
```

Algorithm 4.4: Log Analyzer

4.3 Context Representation in CASMA

Context involves situations of entity in a given physical environment as shown in Fig. 4.2. It considers an entity existed in the physical space and a relation with other entities. It checks context information that the existed entities could share to process required functionality. A user in a university setting, for instance, can be considered as an entity and daily routine activities in relation to other entities in the space can be considered as inputs for context data. Another scenario to show such an interaction is a student with an ID who wants to access services which are located in a given small space with the help of some resources in the area. Such activities can provide location, time, identity and other related context information of every entity presented in the process.

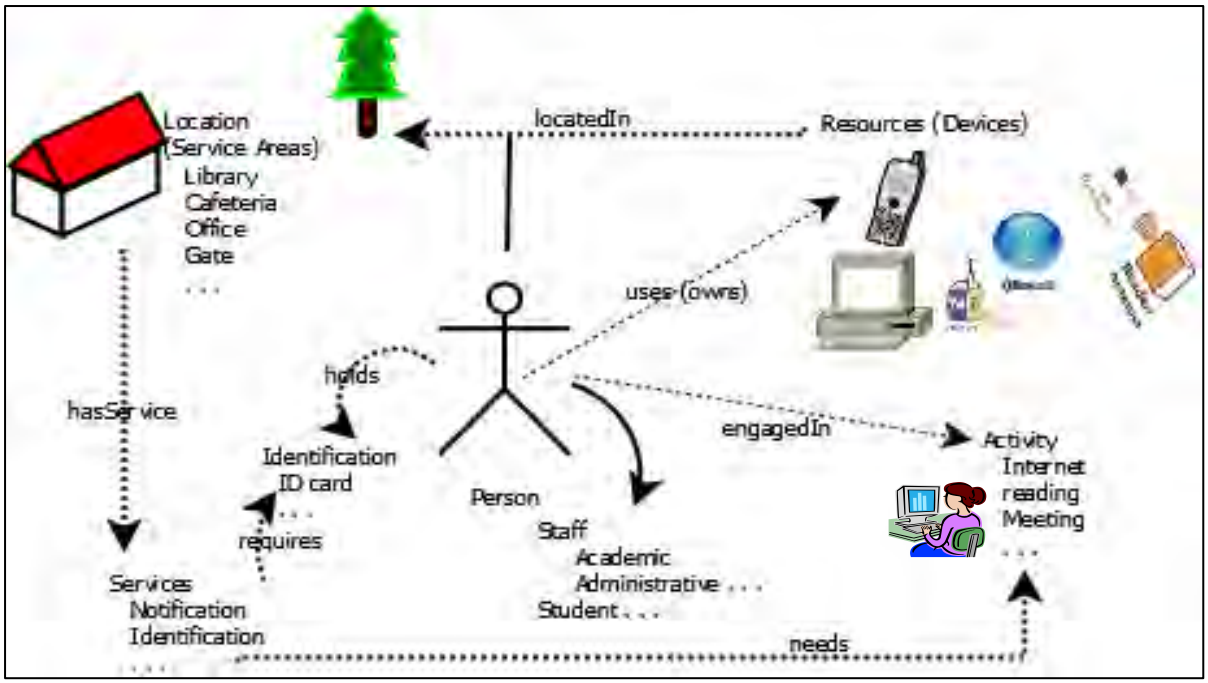


Fig. 4.2: Sample entity activities and relations in university setting

Such context distribution should be represented in a meaningful way to simplify and properly present the possible contextual information captured. Context representation in the proposed architecture provides a combination of domain and generic contextual information that characterize a given entity. The generic context provides context information that does not directly represent the context of a given domain but in combination with the domain context. Context representation of such a model is provided in Fig. 4.3 considering the university setting.

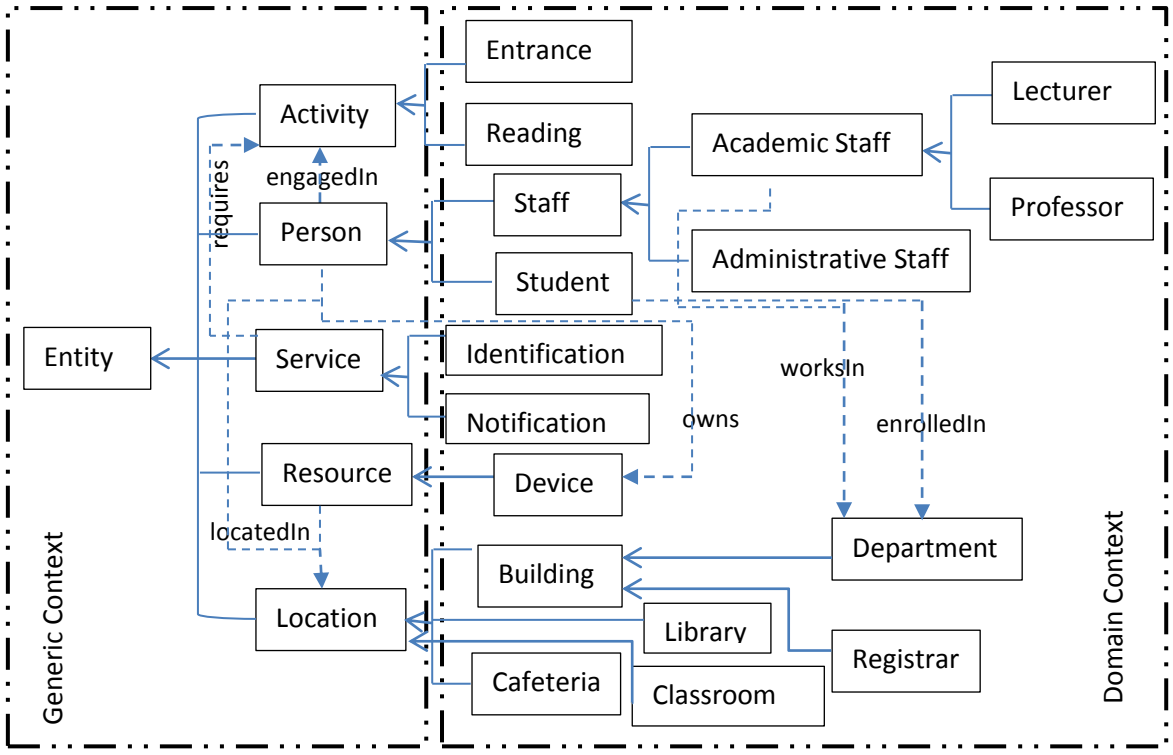


Fig. 4.3: Context representation model in university setting

The context model is based on ontological modeling approach and uses ontologies to represent the context. Ontology is a description of concepts and relationships and allows a programmer to specify the concepts and relationships in an open meaningful way that collectively characterizes a given physical environment. It may be considered similar to a well-defined database schema in which it is an explicit, first-class description. Its model has multi-purpose in which having been developed for one purpose, it can also be published and reused for other purposes.

Different ways of representing ontology are suggested (See section 2.5) and one of the most widely known alternatives is OWL. It is designed for use by applications which process content of information and has greater machine interpretability of contents. Evolved out of DAML and OIL, it is built on top of RDF/RDFS and provides a way of describing resources. It is more expressive in which it supports ontology importing, different types of properties, cardinalities, characteristics of properties, value restrictions, equivalence between classes and individual and set operation.

A sample representation of the proposed context representation model is shown in Fig. 4.4. It presents selected class hierarchy and individual members, and shows the model classification

of generic and domain ontology to represent generic and domain context, respectively, along with their relationship.

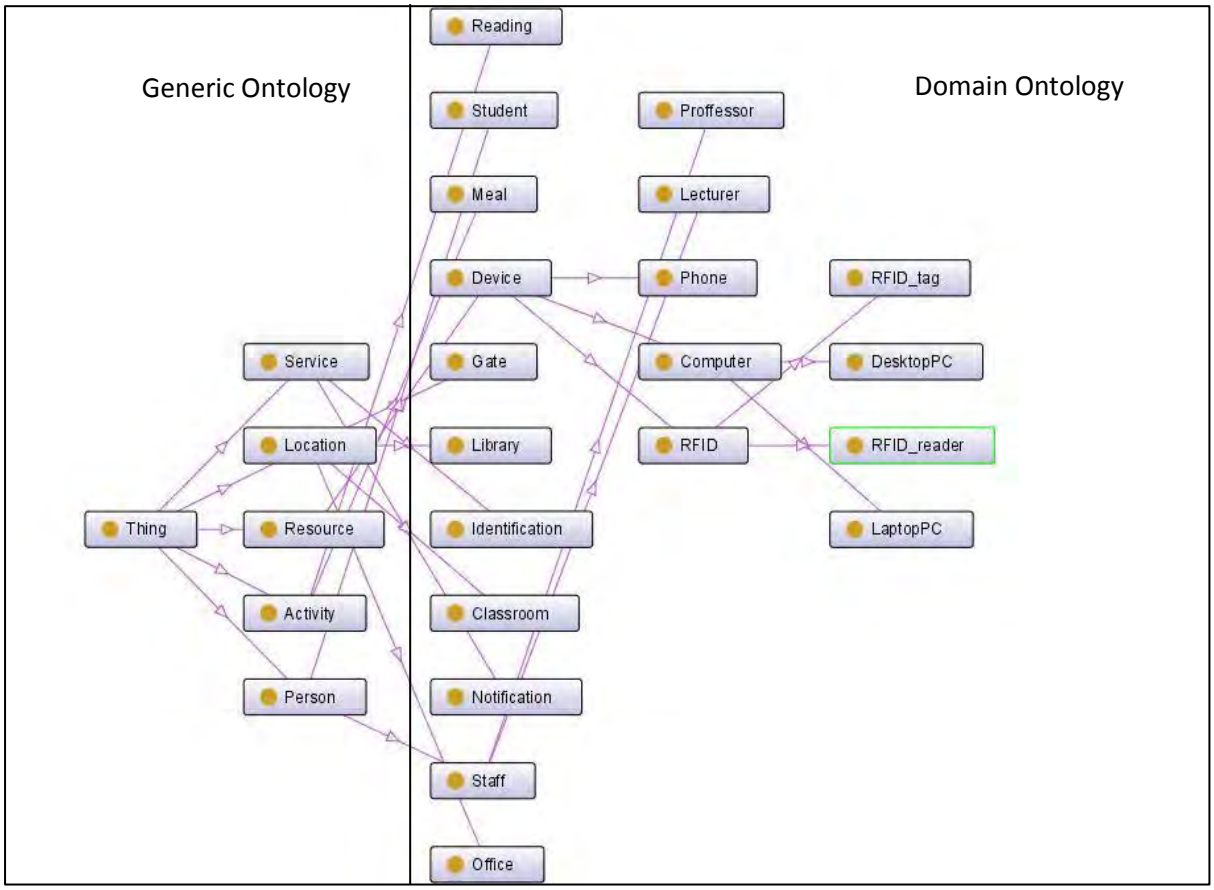


Fig. 4.4: Simplified ontology class representation graph for the context representation model

4.4 Context Management in CASMA

Context management model, as depicted in Fig. 4.5, provides a context analysis and decision relevant for context-aware service delivery. Algorithm 4.2 and Algorithm 4.3 show the details presented in the context management model of CASMA.

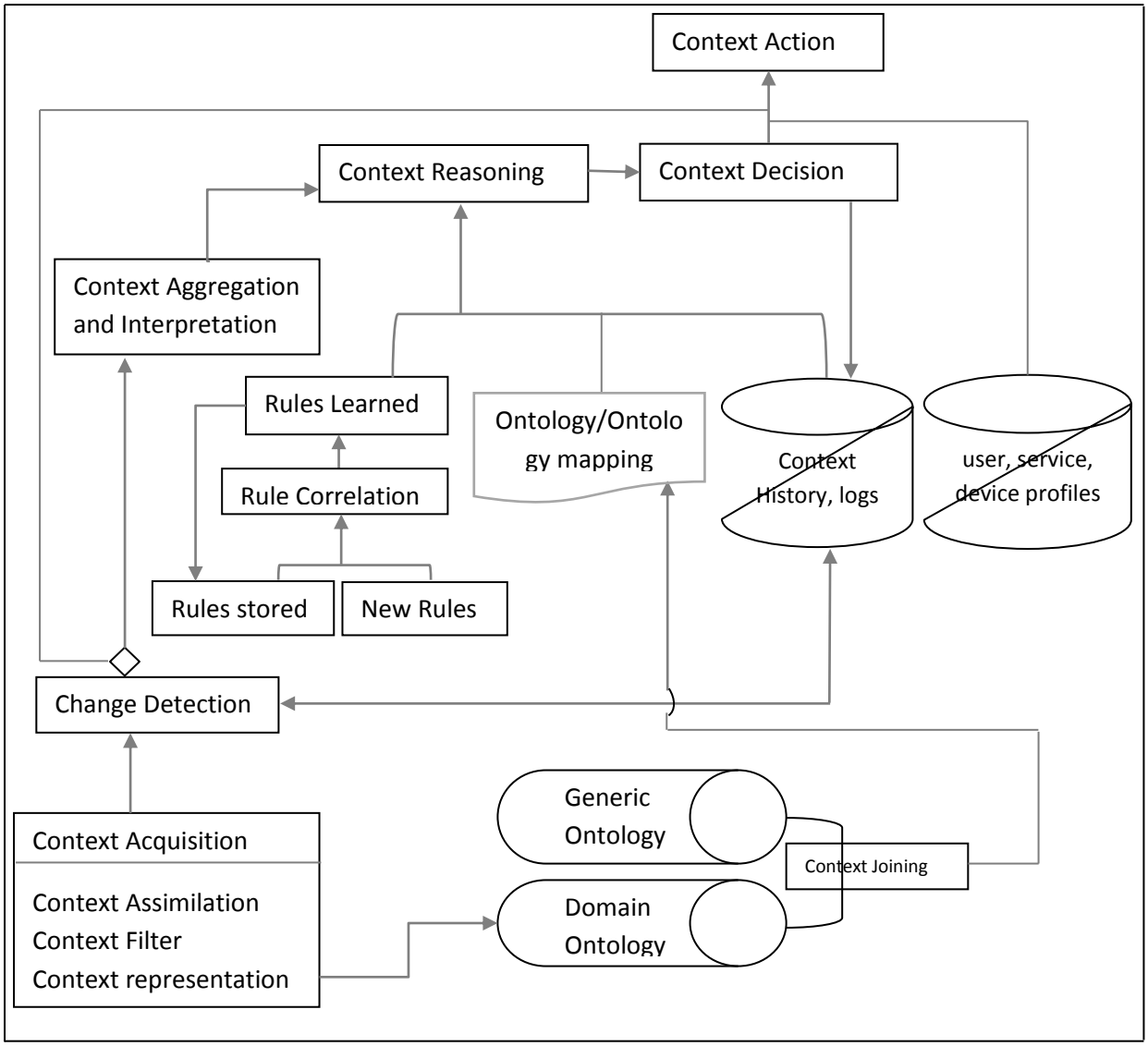


Fig. 4.5: Context management model

Context organized in acquisition module can get its representation as domain ontology and join with related generic ontology. Context history repository also stores previous context decisions to provide relevant context comparison with the new context to be analyzed. Rules on the other hand are captured, aggregated and compared to already defined and stored rules to learn and predict the rules necessary for processing. Change detection service checks for existence of any context change in new and historical context repository maintaining the rules that are learned. It then sends change in context information, if any, to context aggregation module. Interpretation of the aggregated context data is provided in context interpretation module ready for reasoning service. Reasoning is made based on information provided from the new context ontology, historical context information, rules and context changes detected. Context decision based on higher level context information is made to trigger context action.

Static context data about devices, users or services can also be used from relational database to get immediate action, or associate with the context decision made to get context action ready for service delivery.

The context management model provides the necessary context decision after analysis is made on the captured low level context data. The process starts from low level context data representation through context operations to come up with the required final contextual information.

CHAPTER FIVE

IMPLEMENTATION AND DISCUSSION

In this Chapter the implementation details of CASMA is presented. Use case scenarios in a university setting are provided to assess the usability and applicability of the proposed generic architecture. Services with various contexts and that require pervasive identification for their delivery in smart campus are taken into consideration during the demonstration of the use case scenarios. Context capturing tools are used to sense context data. Experiment using numerous RDF-triples is made on the scenarios presented.

5.1 Implementation

The implementation plan follows the process flow of CASMA for providing its demonstration. It tends to provide both the proof of concepts in CASMA and its use in smart campus, mainly for identification related services. The flow starts from its initial resource discovery mechanism and context data capturing techniques to the final delivery of service. Resources consist of physical devices distributed over a given space. Context capturing considers a collection of such resources along with software tools to capture, preprocess and transfer context data and supporting rules to the next component for representation and management processes. Context representation is provided with a collection of tools with support for ontological modeling structure. Context management, with support of data management tools, is also performed on context data and rules captured and represented. It performs semantic reasoning and aggregation on the raw context data captured. It interacts to change monitoring section and repositories that store ontology, rules and other supporting information to retrieve relevant context and make decision about it. The process continues to the part where context decision link with service type to provide or perform operation of relevant services. Standard security, such as authentication and access control, and log analyzer are also tested to provide security mechanism.

Several tools and technologies were utilized for the purpose of developing the prototype implementation. The following is list of programming, communication, database management, context representation, context reasoning and operating environment which are used in the prototype implementation.

- Java programming language is used to write the implementation of the prototype. The Java™ Programming Language is a general-purpose, concurrent, strongly typed, class-based object-oriented language. It is normally compiled to the bytecode instruction set and binary format defined in the Java Virtual Machine Specification. It is designed to be platform independent with additional key principles such as usability, reliability and security.
- NetBeans Integrated Development Environment (IDE) version 8 is used for developing the prototype with Java the selected programming language. This version of NetBeans support Java Platform Standard Edition (Java SE) specification version 8 with Java Development Kit (JDK) 8 and Java SE Runtime Environment (JRE) 8.
- MySQL database is used to store service related data in the prototype development. It is a widely used open source relational database management system.
- Protégé is used to construct the ontologies required in the prototype. It is a free, open-source platform that provides a suite of tools to construct domain models and knowledge-based applications with ontologies. It has ontology editing environment with support for the OWL 2 Web Ontology Language, and direct in-memory connections to description logic reasoners like pellet.
- Jena is used to access and manipulate ontology concepts modeled through java programming environment. It is a free and open source Java framework for building semantic web and Linked Data applications. The framework is composed of different APIs interacting together to process RDF data. It includes RDF API, Ontology API and support for SPARQL.
- SPARQL is used to query the information held in the ontology modeled for the prototype. It is a data-oriented query language and a protocol for accessing RDF.
- SWRL is used to provide rules to the ontology concepts modeled. It is a Semantic Web Rule Language that can be used to express rules as well as logic combining OWL DL or OWL Lite with a subset of the Rule Markup Language.
- Additional to Jena Inference API, Pellet Reasoner is used for reasoning the ontology according to concepts, properties and rules provided. It is an open source OWL-DL reasoner with support for reasoning with individuals, user-defined data types, and debugging support for ontologies. It also provides support for SWRL rules.

With Java is the selected language due to its portability to multiplatform environment and its support to include open source Java development API tools, the classes are structured accordingly taking CASMA in the forefront of organizing the concepts.

5.2 Use Case Scenarios

Pervasive computing enhances service deliveries in various application areas. A university setting, which is also the selected test environment in this study, is one area of application that could take this computing advantage. Educational technology in such settings is constantly evolving and growing in which it progresses to continually offer new and interesting advances. The instigation of ubiquitous media of instruction for delivery of education is a straightforward mechanism in lifting educational benefits to a higher level. However, activities in universities are more than the necessary delivery of instructions or teaching in a classroom. Apart from classroom activity, there are a number of services in campus grounds that involve students and/or staffs. Such activities include using library services, cafeteria and recreational services, notification services, registrations and other official services.

A use of one card system offers its own advantages in facilitating the process of routine activities in a university. However, its level of effectiveness is still in doubt as users' effort is not yet minimized to a higher degree. This is due to concerns of manipulation of services using barcode technology which still involves significant manual work. It is also down to lack of consideration for user situation and other supporting contextual information in such systems.

In addition to finding a way of delivering the services, a better way of identification mechanism is required to process those services and hence scenarios on bringing pervasive identification and delivery of context aware services in such settings are considered to provide smart campus. A smart campus is a campus aware of its contexts that are relevant and use them to deliver services to its users.

Two simple scenarios showing the significance of pervasive computing and context awareness in campus setting are presented here. The scenarios incorporate the use of RFID technology and one card identification mechanism in the domain. The scenarios assume RFID readers are installed at strategic locations in the campus and a tag is embedded to user identification card.

- *Fisihatsion is a professor at a University. He wants to check the attendance of his students while they are working on an activity. Without interrupting their work, he can do so from his computer using the information recorded from a tag in the students' ID cards using a tag reader installed in the room*
- *Alem needs to check a notification, if any, automatically when she enters her office (notification area, if student). The reader detects her ID card and displays the notification on her computer display (personal device). She also wants the convenience of using the same ID card to get in to the campus premises and use a cafeteria. This is possible using a tag in her ID. Her identification can be retrieved from the tag in her ID and used for monitoring.*

A demonstration version of CASMA is implemented and provided in this section. The scenarios are used to discuss the function of CASMA in smart campus domain by experimenting on the prototype of its concepts. Two general scenarios are selected and presented with regard to the internal components of the proposed architecture.

5.2.1 One Card System Scenario

The scenarios presented require a concept of more general way of representation to handle identification to deliver a given service. A one card system is the selected scenario to show what is required towards the implementation. For delivery of various services requires a one card, which can be an input to provide necessary context information that in turn could help identify the respective user. It also helps to identify the necessary service along with other contextual information.

Several questions can be raised regarding this scenario. The first question that can be considered is what physical resources are available to enable the identification process? Which resources belong to the user? Who is the user? Where and when the identification process is taken place? What services are available and related to the user?

As it is discussed in the proposed architecture, ontology is used to represent and have semantics about the concepts related to the components in the scenario. In a campus setting, a person whom is a student, professor, or administrative worker, can be located in various places to get access from a given service. For instance, a student can be located at any of the gates of the university to get to the premise of the campus. The identification service should act automatically towards the change of user to provide the new details. Furthermore, a

student can be engaged in an activity related to some locations in the campus such as library, cafeteria, classrooms, lounges, internet zones and others. He/she usually holds personal physical resources like RFID tag embedded identification card, Bluetooth-enabled devices, and other personal devices. There is also a possibility that the user is surrounded by other physical resources in those locations.

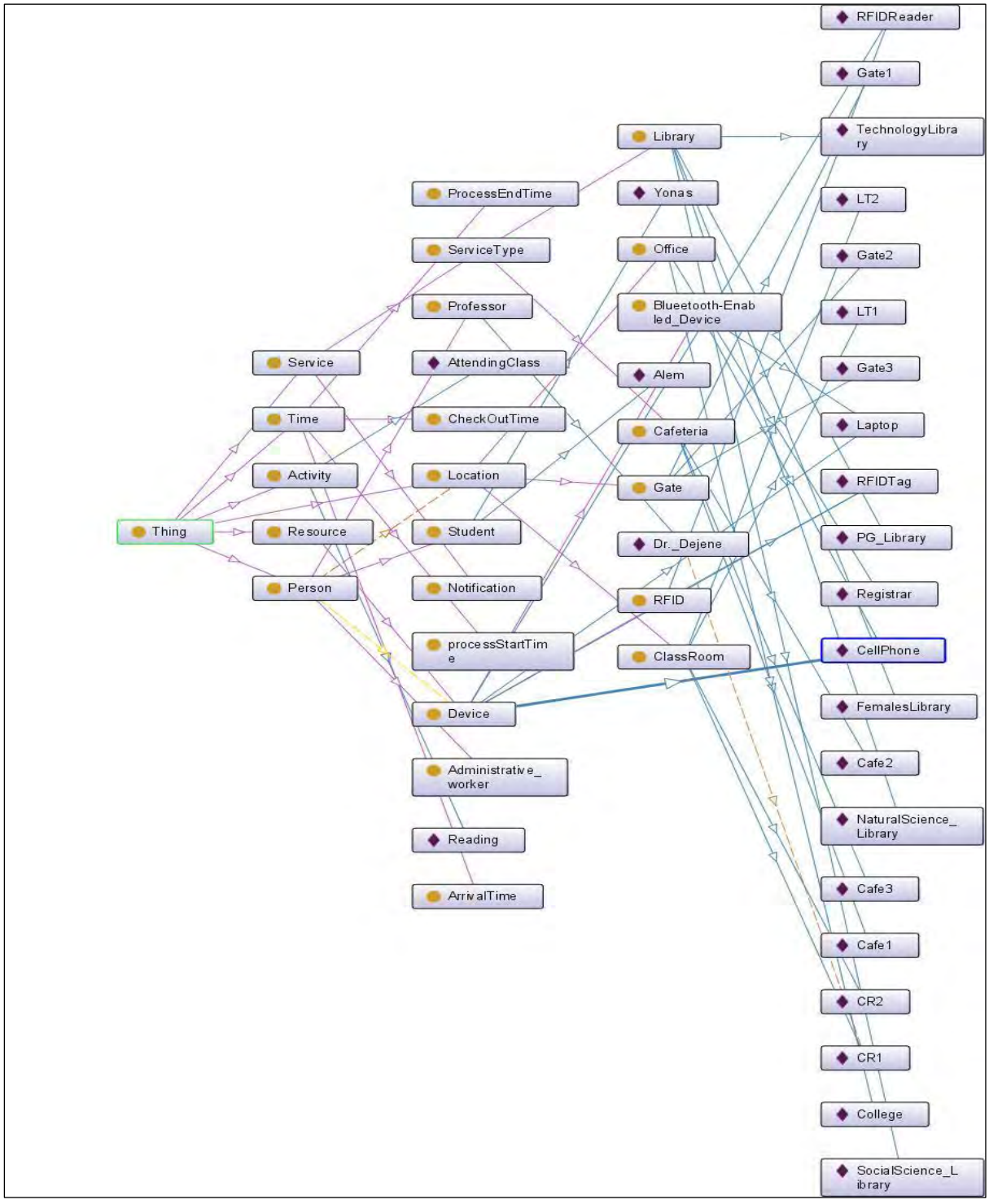


Fig. 5.1: sample ontology graphical representation from one card system scenario

Ontology is provided to represent, model and relate the context information. Graphical representation of part of the ontology is given in Fig. 5.1. The context information is represented in OWL RDF/XML format. Fig. 5.2 shows the representation of sample context information in the one card scenario. Among other possible syntax representation of OWL are OWL/XML, OWL Functional, Mancherster and Turtle. Such representation can be stored in an owl file extensions. Jena can access from or store to only suitable text file formats or Jena compatible databases.

```
<?xml version="1.0"?> . . .
    <rdf:RDF . . .
<owl:Ontology rdf:about="http://www.OneCardOntology.com/Onto1.owl"/>
    <owl:ObjectProperty rdf:hasProgramType">
        <rdfs:range rdf:Program"/>
        <rdfs:domain rdf:Student"/>
    </owl:ObjectProperty>
    <owl:ObjectProperty rdf:isStudentOf">
        <rdfs:range rdf:Proffessor"/>
        <rdfs:domain rdf:Student"/>
    </owl:ObjectProperty>
    <owl:ObjectProperty rdf:isTeacherOf">
        <owl:inverseOf rdf:isStudentOf"/>
    </owl:ObjectProperty> . . .
    <owl:DatatypeProperty rdf:hasEmail"/>
    <owl:DatatypeProperty rdf:hasID"/>
    <owl:DatatypeProperty rdf:teachesCourse">
        <rdfs:domain rdf:Proffessor"/>
        <rdfs:range rdf:string"/>
    </owl:DatatypeProperty> . . .
    <owl:Class rdf:Person"/>
    <owl:Class rdf:Proffessor">
        <rdfs:subClassOf rdf:Staff"/>
    </owl:Class>
    <owl:Class rdf:Program"/>
    <owl:Class rdf:Regular">
        <rdfs:subClassOf rdf:Program"/>
    </owl:Class>
    <owl:Class rdf:Service"/>
    <owl:Class rdf:Staff">
        <rdfs:subClassOf rdf:Person"/>
    </owl:Class>
    <owl:Class rdf:Student">
        <rdfs:subClassOf rdf:Person"/>
    </owl:Class>
    <owl:Class rdf:Summer">
        <rdfs:subClassOf rdf:Program"/>
    </owl:Class> . . .
    <owl:NamedIndividual rdf:Alem">
        <rdf:type rdf:Student"/>
        <hasID>Alem05</hasID>
        <hasEmail xml:lang="yahoo.com">alemmeaza</hasEmail>
    </owl:NamedIndividual>
    <owl:NamedIndividual rdf:Dejene">
        <rdf:type rdf:Proffessor"/>
        <hasEmail xml:lang="yahoo.com">ejigud</hasEmail>
        <isTeacherOf rdf:Yonas"/>
    </owl:NamedIndividual>
    <owl:NamedIndividual rdf:Yonas">
        <rdf:type rdf:Student"/>
        <hasID>GSR/1173/04</hasID>
        <hasEmail xml:lang="gmail.com">yonitad</hasEmail>
    </owl:NamedIndividual>
</rdf:RDF>
```

Fig 5.2: A sample representation from one card system scenario

Rules, using SWRL, are also included to represent explicit context information for the concept relationship provided. A reasoning service is made on the modeled context

information combining all concept relationships and rules. It is to draw important context results using reasoning engines like Pellet and Jena's inference API.

A SPARQL query is used to navigate through RDF concept specification and retrieve results based on the context decision made using the reasoner. For instance, a sample SPARQL query to select list of users who are students along with location information can be given as:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX campus: <http://www.owl-ontologies.com/Ontology1399933040.owl#>
Select ?Name ?Location
WHERE{
        ?Name rdf: type campus:Student
        ?Name campus:locatedIn ?Location
}
```

5.2.2 Library Check Point Notification Scenario

The first scenario presented only focused on identification of users by capturing some important context data related to the user. It then associates them by looking for availability of service types required to be delivered for the user. It helps identify users without requiring them to participate in the process. This minimizes much user effort and prevents unnecessary disturbance, thus preserving one of main concepts of pervasive computing. It can be considered as an input to several services that require context management as well as identification. However, this work's focus is not limited to such context management. It also intends to relate contexts with service deliveries. This in turn can result in further context management to cope with the dynamically changing available context data.

Notification service is the most common service type in different application areas holding the necessary requirements that can show the expected functionalities from components of CASMA. Notification in a campus includes but not limited to services like grade reports to students, announcements, campus service information, library notifications, and other user related status information. Implementation for notification of checkpoint service in a university library setting is selected from the available options in this scenario. Notification

for building information is also included in the library notification services where information about internal organization of the library is to be delivered for newly arriving student.

Library check point notification includes services such as notification for new book arrivals and control of book issue and return. Questions that should be answered in such service scenarios include: Who is the user located near the computer screen or other display? What is the user work status? What is the required related notification service to deliver? If the service is notification of new arrival books, then when was the last time the respective user checked into the service area? Also, what is the user preference in terms of study category? Is there any available newly entered book related to categorical reference of the user? Or if the service is book issue and return, then is a deadline met according to the timeframe?

The ontology modeled for the first scenario is reused to get an input and merge with a new additional domain level ontology to handle service contexts. The domain level ontology can be represented in a similar way to the first scenario. It can help identify and represent user context information for required services. Service type can then be identified using user status context. For example, an ontology representation can be given to identify whether the user located in the library is a student, library worker or other matching profile. For newly arrival books, historical context information of a previously occurred event has to be checked. Based on the historical and newly sensed context data from physical and virtual source, an action can be drawn. Similar consideration applies to the book issue and return service. Graphical ontology representation of the part of context is presented in Fig. 5.3 and sample merged SWRL rules for explicit specification of context demand for both scenarios is given in Fig. 5.4.

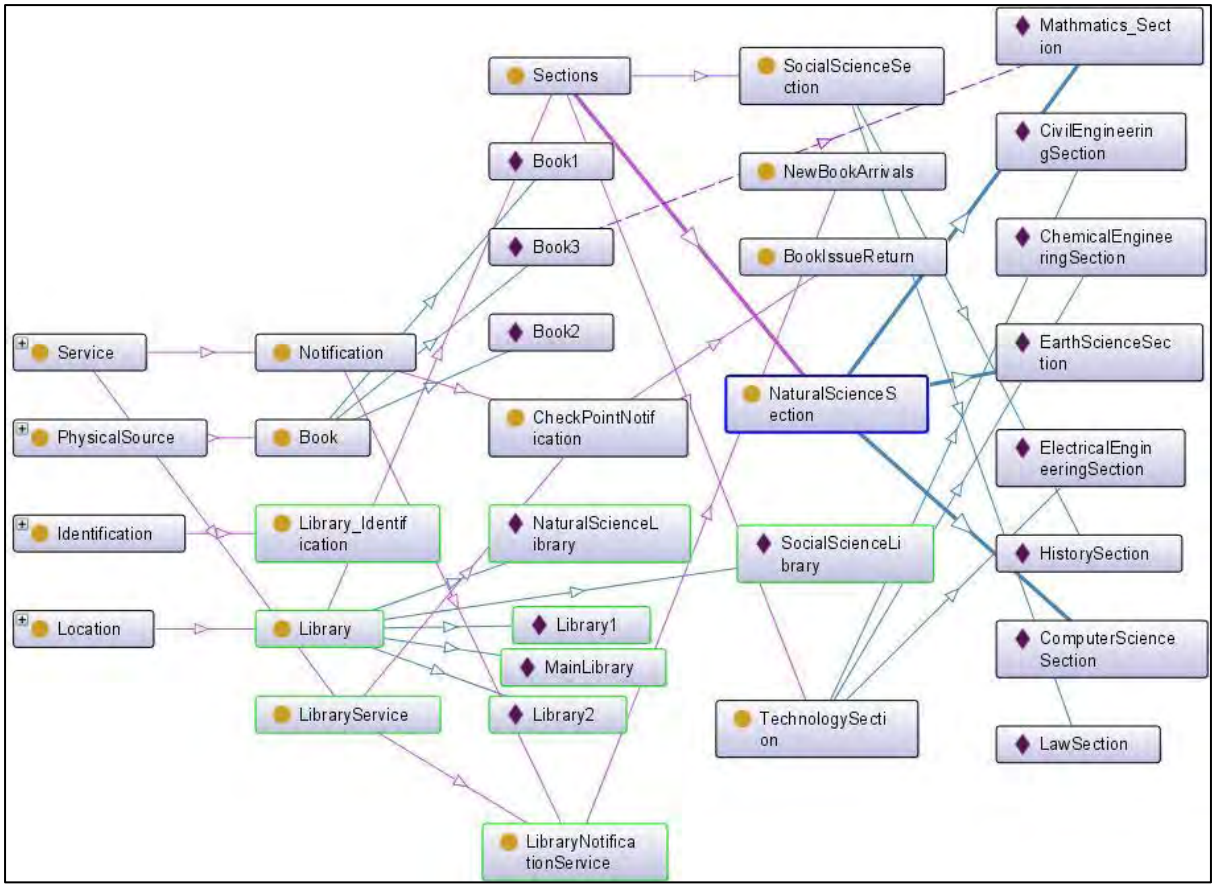


Fig. 5.3: Graphical representation of sample library context ontology

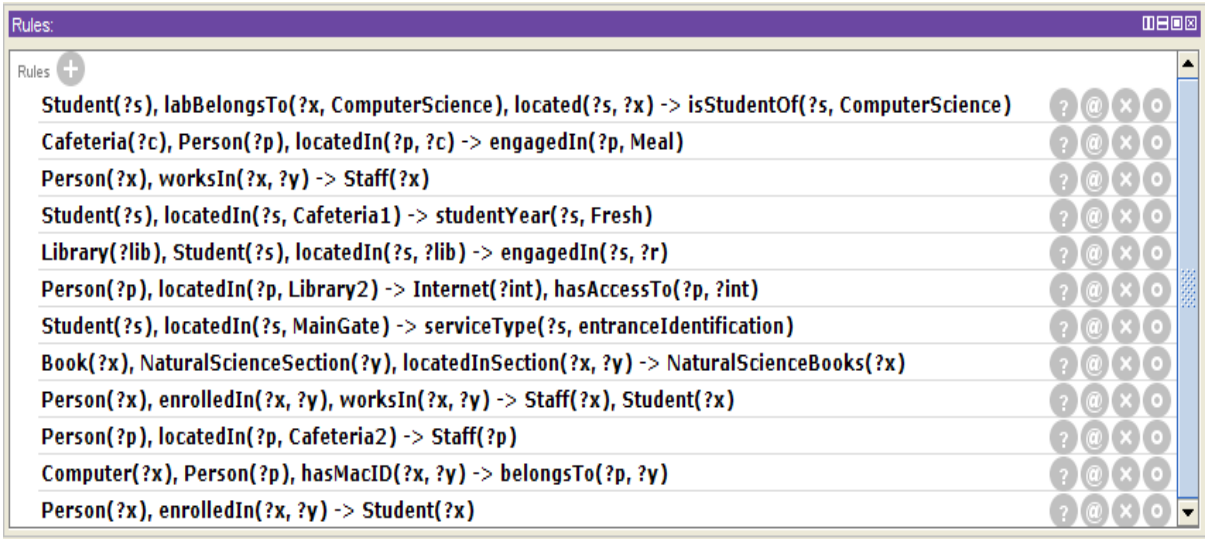


Fig 5.4: Sample representation of SWRL rule

5.2.3 Demonstration and Evaluation

Physical and virtual context sources are used to experiment the applicability of resource management and context sensing. Virtual source like calendar is accessed to input context data. Physical context sources like RFID and wireless connectivity such as Bluetooth and Wi-Fi technology are included. An RFID reader along with different tags is primarily referred for identification of user while existence of related Bluetooth devices is also used for additional capturing of context data. Screen shot to show resource management and context capturing is provided in Fig. 5.5.

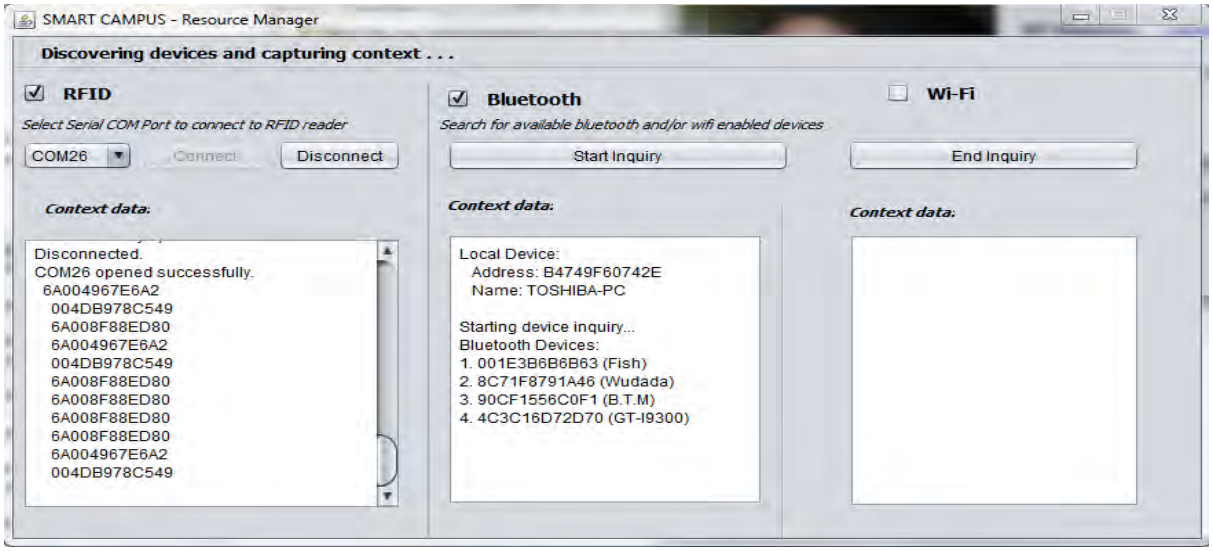
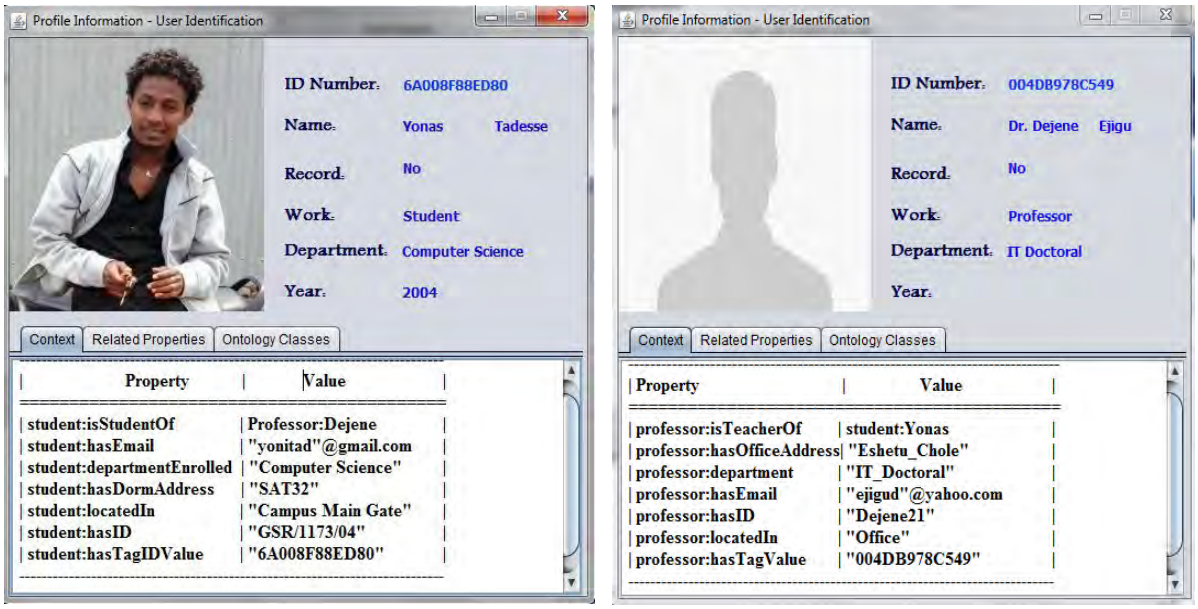


Fig. 5.5: Screen shot for resource management and context capturing processes

The scenario implementation involved varying number of context instances created from context entities. Context instances that vary from 100 to 6,000 comprising individuals, relations, rules, location, and resource instances are used. Reasoning service on the provided context data size is made. It helps to check ontology consistency and provide new results of context information with acceptable system performance. It can infer new results of context relation based on the asserted context information. For example, consider a student X having a relation *isStudentOf* with individual Y of class type Teacher. If the property is asserted to be an inverse functional property of *isTeacherOf*, then the reasoner can automatically infer relation that individual Y *isTeacherOf* X. Another example of reasoning can be provided in relation to rules. If a user is located in a library and if there is available new books in his preference category, then provide a notification of list of new books description. The reasoner can trigger the action accordingly to such associated rules.

The components of CASMA provide a platform for implementation, running and testing of services with the presented context data and context decision through context management. Sample screen shots are provided for the one card system scenario in Fig. 5.6, and for library checkpoint notification service in Fig. 5.7. Description of sample code segment is provided in Annex I. Demonstration of sample ontology context data and sample rules are also provided in Annex II and III, respectively.



(a) (b)

Fig.5.6: Screen shots of the one card system scenario implementation

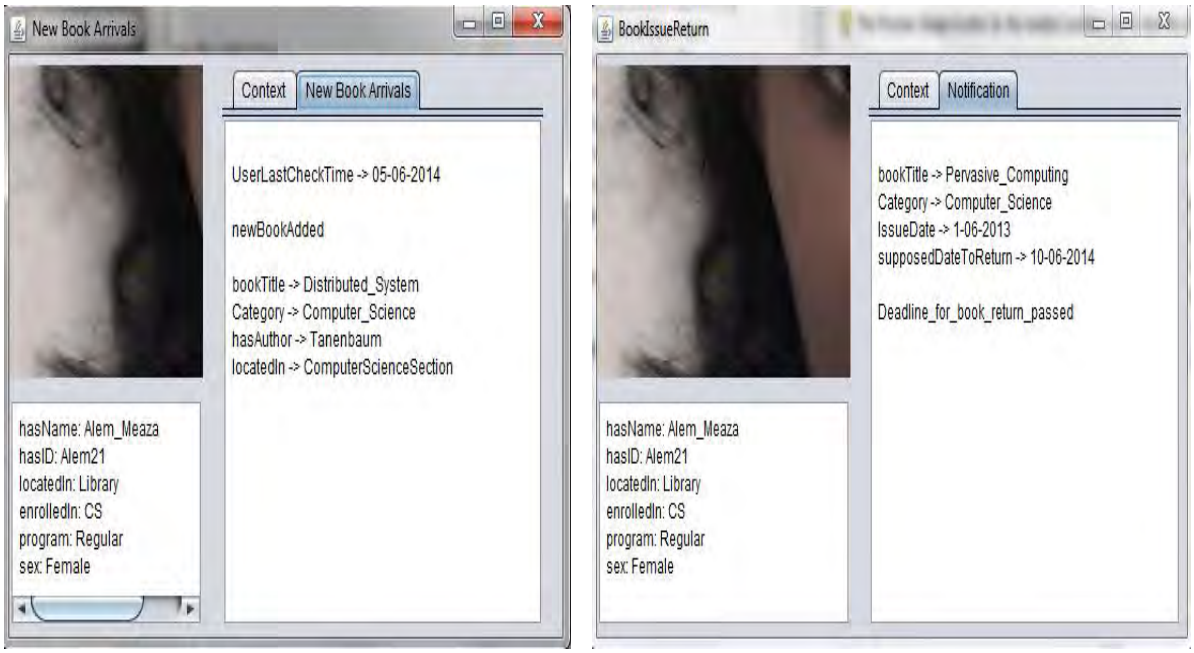


Fig. 5.7: Screen shots of the implementation for library checkpoint notification scenario

In terms of security, log monitoring is tested to evaluate its importance in addition to standard securities like authentication and access control. It is used to monitor event actions recorded in log file in the process of context aware service delivery. For instance, it is used to check records that tell information about an unknown tag that access services which are restricted. It provides information such as timestamp, tag id detected and service type accessed. Also, a recognized tag that has limited access to specific service types is considered. It reacts to such behaviors to prevent malicious activity in progress. Similarly, information about the user identity, time of access and service types accessed is recorded to enable later monitoring of information.

As conclusion, functionality of CASMA is addressed using a walkthrough on example scenarios presented. The effects of context resource management, context capturing, context management, and service delivery along with security mechanism are demonstrated using examples from identification and library notification services. It is applicable for data size from small to large variations. Context reasoning in context management is also tested with context ontology and rules to provide relevant results with acceptable performance. The platform usability to inputs from other domains also is shown with a change of context input to both scenarios. This is due to the reason that input components are not system dependent and are stored separately. Context information represented and modeled can also be re-used to other context service manipulations proving re-use of context information. The merge of context data also proves the scalability of the platform towards context data inputs. Management of logs tested also shows applicability of monitoring of security in service deliveries.

CHAPTER SIX

CONCLUSIONS AND FUTURE WORKS

6.1 Conclusions

Following results of this research, pervasive computing effort to augment computing devices with self-awareness and awareness of their environment is shown. An overview of context, context awareness and context aware computing is provided. Context, according to Dey and Abowd [29], is any information that can be used to characterize the situation of an entity. Context-aware computing deals with context awareness concepts, which focuses on how to be able to use relevant context information. Context acquisition along with resource management is an important element to context management.

A modeling approach is required to represent and model context. A survey is made on available context modeling approaches. It confirms that ontology based model fulfills important requirements better than other relating modeling approaches. In this approach, Ontology is the main concept. It specifies the concepts, relationships, and other distinctions that are relevant for modeling a domain context. A formal ontology language is required to encode the ontology. OWL is a semantic web language which is a W3C recommendation. It is designed to represent rich and complex knowledge about things, groups of things and relations among them. There are also potential development tools, like protégé, available to support the use of OWL.

Architectural support is needed towards addressing the requirements of context awareness. Many works have been suggested, in its early development of context awareness, which base their focus on location aware services. They also could not satisfy the required level of context abstraction, extensibility and reasoning support in context management. Later works provide semantically rich and expressive models and a support for a higher degree of reasoning. It was required to have further study to offer way of resource discovery, handling historical context information with additional learning mechanism, handling of context change and security. This study provides generic context-aware service management architecture that deals with these components. It uses ontology modeling approach to model and represent context. It ensures necessary requirements of pervasive system that consider existence of various entities, services and their collaboration with the frequent change of

context in pervasive environment. It also embeds security using log security management in addition to standard security mechanisms such as authentication and access control.

University setting is selected after investigating potential application areas to visualize computing resource advantage and context information requirements in the proposed architecture. University campus is an ideal location for the development, deployment and testing of ubiquitous computing technologies. This is due to existence of computing devices, mobile access to information in everyday student encounters and strong infrastructure support. Most of the research projects done in this domain are limited to enhance techniques of teaching and learning. Few consider working on delivery of various services offered in the domain. However, no proper mechanism is provided to handling contextual information around the campus and to distinguish between low level context data from its high level interpretation. These issues are addressed in the implementation of the proposed architecture with support for context management of modeled context.

An experiment made on one card system and notification service shows the usability, applicability and scalability of the proposed architecture. The platform is also usable to inputs from other domains. It also preserves concepts of pervasive computing. It bases itself on context information of its surrounding and minimizes user effort required than its traditional counterpart in delivering services.

6.2 Contributions

The contribution of this thesis work is summarized as follows:

- Provide a novel architecture for context aware service management with capabilities of context acquisition, abstraction, management, reasoning and decision along with a way of service integration. It also includes a log management security for pervasive delivery of services.
- Enhance resource management and context acquisition techniques utilizing computing devices.
- Provide a way of integrating application advantage of pervasive computing into the environment in different application areas.
- Enhance a delivery of services by embedding concepts of context awareness.
- Enhance security mechanism in pervasive computing to enable prevention and monitoring of security threats.

6.3 Future Works

The following are some of the potential future works to the continuation of this work.

- Providing an enhanced way of security and privacy mechanisms. This study mainly considers log management towards providing security mechanism. However, further studies should be made on issues of privacy. Issue of log data volume management should also be studied further.
- A better security mechanism towards communication between pervasive devices can also be further studied. Candidate solutions to be further investigated include custom protocol and encryption mechanism in enabling secure communication between pervasive devices.
- Provide identification service with additional resources like camera and GPS, and embedding capabilities of face identification concerning domains that demand higher security.
- In terms of application area: integration of pervasive inner classroom activities, especially delivery of courses, with the proposed model of outer class activities.
- Embedding capabilities of mash-up service so that machines can share information to provide the user with required service according to context. For instance, libraries can share what is available on their library displays; researches conducted can be retrieved based on different categories of search.

REFERENCES:

[1] Friedemann Mattern. The Vision and Technical Foundations of Ubiquitous Computing. Upgrade European Online Magazine, pp. 5-8, October 2001.

[2] Mark Weiser. The Computer for the Twenty-First Century. Scientific American, 265(3), pp. 94-100, September 1991.

[3] Louise Barkhuus and Paul Dourish. Everyday Encounters with Context-Aware Computing in a Campus Environment. Proceedings of the Sixth International Conference on Ubiquitous Computing, pp.232-249, 2004.

[4] Sung, M., Gips, J., Eagle, N., Madan, A., Caneel, R., Devaul, R., et al.. Mobile-IT Education (MIT. EDU): m-Learning applications for classroom settings. Journal of Computer Assisted Learning, 21(3), 229–237, 2005.

[5] Stephen S. Yau, Sandeep K. S. Gupta, Fariaz Karim,, Sheikh I. Ahamed, Yu Wang, and Bin Wang. Smart Classroom: Enhancing Collaborative Learning Using Pervasive Computing Technology. In ASEE 2003 Annual Conference and Exposition, 2003.

[6] Yue Suo, Naoki Miyata, Hiroki Morikawa, Toru Ishida, Yuanchun Shi. Open Smart Classroom: Extensible and Scalable Learning System in Smart Space Using Web Service Technology. IEEE Transaction on Knowledge and Data Engineering, 21 (6), pp. 814-828, June 2009

[7] Dekdouk, A. Integrating mobile and ubiquitous computing in a smart classroom to increase learning effectiveness. IEEE, Education and e-Learning Innovations (ICEELI) International Conference, pp.1-5, July 2012.

[8] Weiser, M. The future of Ubiquitous Computing on Campus. Communications of the ACM, 41(1), January 1998.

[9] M. Satyanarayanan. Pervasive Computing: Vision and Challenges. IEEE Personal Communications, 8(4), pp. 10-17, August 2001.

[10] Paul Dourish. What we talk about when we talk about context. Personal and Ubiquitous Computing, 8(1), pp.19-30, 2004.

[11] Katz, R.H., Long, D., Satyanarayanan, M., Tripathi, S. Workspaces in the Information Age. In Report of the NSF Workshop on Workspaces in the Information Age. Leesburg, VA, October, 1996.

[12] Baek, S. H., Lee, H. J., Lim, S. Y., & Huh, J. D. Managing mechanism for service compatibility and interaction issues in context-aware ubiquitous home. IEEE Transactions on Consumer Electronics, 51(2), pp.524–528, 2005.

[13] Intille, S. S. Designing a home of the future. IEEE Pervasive Computing, 1(2), pp.76–82, 2002.

[14] Schulzrinne, H., Wu, X., Sidiroglou, S., & Berger, S. Ubiquitous computing in home networks. IEEE Communications Magazine, 41(11), pp.128–135, 2003.

[15] Bottazzi, D., Corradi, A., & Montanari, R. Context-aware middleware solutions for anytime and anywhere emergency assistance to elderly people. Communications Magazine, IEEE, 44(4), pp.82–90, 2006.

[16] Favela, J., Tentori, M., Castro, L. A., Gonzalez, V. M., Moran, E. V., & Martínez-García, A. I. Activity recognition for context-aware hospital applications: Issues and opportunities for the deployment of pervasive networks. Mobile Networks and Applications, 12(2–3), pp.155–171, 2007.

[17] Kjeldskov, J., & Skov, M. B. Exploring context-awareness for ubiquitous computing in the healthcare domain. Personal and Ubiquitous Computing, 11(7), pp.549–562, 2007.

[18] Rodríguez, M. D., Favela, J., Martínez, E. A., & Muñoz, M. A. Location-aware access to hospital information and services. IEEE Transactions on Information Technology in Biomedicine, 8(4), pp.448–455, 2004.

[19] Kj Korpipää, P., Mäntyjärvi, J., Kela, J., Keränen, H., & Malm, E. J. Managing context information in mobile devices. IEEE Pervasive Computing, 2(3), pp.42–51, 2003.

[20] Kwon, O. B. Modeling and generating context-aware agent-based applications with amended colored petri nets. Expert Systems with Applications, 27(4), pp.609–621, 2004.

[21] Schilit B, Theimer M. Disseminating active map information to mobile hosts. IEEE Network, 8(5), pp.22–32, 1994.

[22] Schilit B, Adams N and Want R. Context-aware computing applications. Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications, Santa Cruz, CA, IEEE Press, New York, 1994.

[23] Brown, P.J. The stick-e document: a framework for creating context-aware applications. Proceedings of the Electronic Publishing, Palo Alto, pp.259–272, 1996.

[24] Ryan, N., Pascoe, J. and Morse, D. Enhanced reality fieldwork: the context-aware archaeological assistant. Proceeding of the 25th Anniversary Computer Applications in Archaeology, 1997.

[25] Hull, R., Neaves, P. and Bedford-Roberts, J. Towards situated computing. Proceedings of the First International Symposium on Wearable Computers (ISWC), p. 146, 1997.

[26] Albrecht Schmidt: Ubiquitous computing - computing in context. The University of Lancaster, Doctoral, pp.1-294, 2003.

[27] Guanling Chen and David Kotz. A Survey of Context-Aware Mobile Computing Research. Dartmouth Computer Science Technical Report TR2000-381, November 2000.

[28] Russell Beale and Peter Lonsdale. Mobile Context Aware Systems: the intelligence to support tasks and effectively utilize resources. Mobile HCI, pp.240-251, 2004.

[29] Dey A, Abowd G and Salber D. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. Hum-Comp Interact 16(2–4), pp. 97–166, 2001.

[30] Dey, A.K. Context-aware computing: the CyberDesk project. Proceedings of the AAAI, Spring Symposium on Intelligent Environments, Menlo Park, CA, pp.51–54, 1998.

[31] Bolchini, C., Schreiber, F. A., & Tanca, L. A methodology for a very small database design. Information Systems, 32(1), pp.61–82, 2007.

[32] Zhu, F., Mutka, M. W., & Ni, L. M. Service discovery in pervasive computing environments. IEEE Pervasive Computing, 4(4), pp.81–90, 2005.

[33] Jongyi Hong, Euiho Suh, Sung-Jin Kim: Context-aware systems: A literature review and classification. Expert Syst. Appl., 36(4), pp.8509-8522, 2009.

[34] Want, R., Hopper, A., Falcao, V. and Gibbons, J. The active badge location system. ACM Transactions on Information Systems, 10(1), pp.91–102, 1992.

[35] Abowd, G.D., Atkeson, C.G., Hong, J., Long, S., Kooper, R. and Pinkerton, M. Cyberguide: a mobile context-aware tour guide. Wirless Networks, 3(5), pp.421–433, 1997.

[36] Sumi, Y., Etani, T., Fels, S., Simonet, N., Kobayashi, K. and Mase, K. C-map: Building a context-aware mobile assistant for exhibition tours, Community Computing and Support Systems, Social Interaction in Networked Communities [the book is based on the Kyoto Meeting on Social Interaction and Communityware, held in Kyoto, Japan], Springer-Verlag, London, UK, pp.137–154, June 1998.

[37] Cheverst, K., Davies, N., Mitchell, K., Friday, A. and Efs-tratiou, C. Developing a context-aware electronic tourist guide: some issues and experiences. Proceedings of

the SIGCHI conference on Human Factors in Computing Systems, ACM Press, New York, USA, pp.17–24, 2000.

[38] Nissanka Bodhi Priyantha, Allen K. L. Miu, Hari Balakrishnan, Seth Teller. The Cricket Compass for Context-Aware Mobile Applications. Proc. of the 6th ACM MOBICOM Conf., Rome, Italy, July 2001.

[39] Byun, H. E., & Cheverst, K. Utilizing context history to provide dynamic adaptations. Applied Artificial Intelligence, 18(6), pp.533–548, 2004.

[40] Claudia Linnhoff-Popien and Thomas Strang. A Context Modeling Survey. Workshop on Advanced Context Modelling, Reasoning and Management, UbiComp 2004 - The Sixth International Conference on Ubiquitous Computing, Nottingham/England, August 2004.

[41] Karen Henricksen, Jadwiga Indulska, Andry Rakotonirainy. Modeling Context Information in Pervasive Computing Systems. Pervasive, pp.167-180, 2002.

[42] Gerhard Fischer. Context-Aware Systems—The 'Right' Information, at the 'Right' Time, in the 'Right' Place, in the 'Right' Way, to the 'Right' Person. AVI, pp.287-294, 2012.

[43] Rohit Pathak, Satyadhar Joshi. Recent Trends in RFID and a Java based Software Framework for its Integration in Mobile Phones. First Asian Himalayas International Conference on Internet AH-ICI2009 Hyatt Regency Kathmandu, KATHMUNDU, NEPAL 3-5th of November, 2009.

[44] Weixin W., Jongwoo S., Daeyoung K. Complex Event Processing in EPC Sensor Network Middleware for Both RFID and WSN. Proc. 11th IEEE International Symposium on Object Oriented Real-Time Distributed Computing (ISORC), pp. 165–169, 5-7 May 2008.

[45] George Roussos, Vassilis Kostakos. RFID in pervasive computing: State-of-the-art and outlook. ELSEVIER, Pervasive and Mobile Computing, 5, pp.110-131, 2009.

[46] Pereyma M., Motyka I., Lobur M. Perspectives of Smart RFID Tags Usage Fabricated by MEMS Technologies. Proc. International Conference on Perspective Technologies and Methods in MEMS Design, MEMSTECH, pp. 113–113, 23-26 May 2007.

[47] Jeremy Landt. Shrouds of Time, the History of RFID. AIM Publication, Pittsburgh, PA, USA, October 2001.

[48] Aikaterini Mitrokotsa and Christos Douligeris. Integrated RFID and Sensor Networks: Architectures and Applications. Zhang/RFID and Sensor Networks, Chapter 18, AU7777_C018, June 22, 2009

[49] Elisabeth Ilie-zudor, Zsolt Kemény, Péter Egri, László MonostorI. The RFID Technology and Its Current Applications. Proceedings of The Modern Information Technology in the innovation Processes of the Industrial Enterprises – MITIP 2006, pp.29-36, 2006.

[50] Loc Ho, Melody Moh, Zachary Walker, Takeo Hamada, and Ching-Fong Su. A Prototype on RFID and Sensor Networks for Elder Healthcare. In E-Wind '05: Proceedings of the 2005 ACM SIGCOMM workshop on Experimental approaches to wireless network design and analysis New York, NY, USA. ACM, pp.70-75, 2005.

[51] P.C.Jain, K.P.Vijaygopalan. RFID and Wireless Sensor Networks. Proceedings of ASCNT, CDAC, Noida, India, pp. 1 – 11, 2010.

[52] Hai Liu, Miodrag Bolic, Amiya Nayak, Ivan Stojmenoviæ. Integration of RFID and Wireless Sensor Networks.

[53] Lei Zhang and Zhi Wang. Integration of RFID into Wireless Sensor Networks: Architectures, Opportunities and Challenging Problems. Proceedings of the Fifth International Conference on Grid and Cooperative Computing Workshops (GCCW'06), IEEE Computer Society, 2006.

[54] J. Cho, Y. Shim, T. Kwon, and Y. Choi. SARIF: A novel framework for integrating wireless sensor and RFID networks. IEEE Wireless Communications, 14(6), pp.50–56, December 2007.

[55] Tomas Sanchez Lopez, Daeyoung Kim, Gonzalo Huerta Canepa and Koudjo Koumadi. Integrating Wireless Sensors and RFID Tags into Energy-Efficient and Dynamic Context Networks. Oxford University Press on behalf of The British Computer Society, The Computer Journal Advance Access, July 16, 2008

[56] Deborah Estrin, David Culler, Kris Pister, and Gaurav Sukhatme. Connecting the Physical World with Pervasive Networks, Pervasive Computing. IEEE, 1(1), pp.59 – 69, Jan.-March 2002.

[57] R. Clauberg. RFID and sensor networks. Proceedings of RFID Workshop, University of St. Gallen, Switzerland, Sep. 2004.

[58] Frank Siegemund, Michael Rohs. Rendezvous layer protocols for Bluetooth-enabled smart devices. Personal and Ubiquitous Computing, 7(2), pp.91-101, 2003.

[59] Md Touseef Sumer, Ismat Fathima. Current Advancements in Wireless Communication Technology. International Journal of Engineering Sciences and Research Technology, 3(1), pp.293-297, January 2014.

[60] Michael Przybilski, Petteri Nurmi, Patrik Floréen. A Framework for Context Reasoning Systems. IASTED Conf. on Software Engineering, pp. 448-452, 2005.

[61] Claudio Bettini, Oliver Brdiczka, Karen Henricksen, Jadwiga Indulska, Daniela Nicklas, Anand Ranganathan, Daniele Riboni. A survey of context modeling and reasoning techniques. Pervasive and Mobile Computing, 6(2), pp.161-180, 2010.

[62] Matthias Baldauf, Schahram Dustdar and Florian Rosenberg. A survey on context-aware systems. Int. J. Ad Hoc and Ubiquitous Computing, 2(4), pp.263-277, 2007.

[63] T. R. Gruber. A translation approach to portable ontologies. Knowledge Acquisition, 5(2), pp.199-220, 1993.

[64] M. Rahamatullah Khondoker, Paul Mueller. Comparing Ontology Development Tools Based on an Online Survey. Proceedings of the World Congress on Engineering (WCE), London, U.K, June 30 - July 2, 2010.

[65] R. Want, B. N. Schilit, N. I. Adams, R. Gold, K. Peterson, D. Goldberg, J. R. Ellis and M. Weiser. An Overview of the ParcTab Ubiquitous Computing Experiment. IEEE Personal Communications, 2(6), pp.28-43, December 1995.

[66] J. Pascoe. The Stick-e Note Architecture: Extending the Interface Beyond the User. Proceedings of International Conference on Intelligent User Interfaces, pp.261-264, 1997.

[67] G. Abowd, C. Atkeson, .J Hong, S. Long, R. Kooper, and M. Pinkerton. Cyberguide: A mobile context-aware tour guide, CHI'96 Shor paper, 1997.

[68] K. Cheverest, D. Nigel, M. Keith, F. Adrian, E.Christos. Developing a Context-aware Electronic Tourist Guide: Some Issues and Experiences. CHI Letters, 2(1), 2000.

[69] Moeiz Miraoui, Chakib Tadj, Chokri ben Amar. Architectural Survey of Context-Aware Systems in Pervasive Computing Environment. Ubiquitous Computing and Communication Journal, 3(3), pp.68-76, 2008.

[70] A. Dey, G. D. Abowd, and D. Salber. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. Human- Computer Interaction, 16, pp.97–166, 2001.

[71] T. Hofer, W. Schwinger, M. Pichler, G. Leonhartsberger, J. Altmann. Context-awareness on mobile devices – the hydrogen approach. Proceedings of the 36th Annual Hawaii International Conference on System Sciences, 2002 pp.292–302.

[72] Manuel Roman, Christopher K. Hess, Renato Cerqueira, Anand Ranganathan, Roy H., Campbell, and Klara Nahrstedt. Mobile Computing and Communications Review, 6(4), pp.65-67, 2002.

[73] P. Fahy, S. Clarke. CASS – a middleware for mobile context-aware applications. Workshop on Context-Awareness. MobiSys, 2004.

[74] G. Biegel, V. Cahill. A framework for developing mobile, context-aware applications. Proceedings of the 2nd IEEE Conference on Pervasive Computing and Communication, pp.361–365, 2004.

[75] P. Korpipää, J. Mantyjarvi, J. Kela, H. Keranen, E-J. Malm. Managing context information in mobile devices. IEEE Pervasive Computing, 2(3), July–September, pp.42–51, 2003.

[76] Jason I. Hong. The context fabric: an infrastructure for context-aware computing. CHI Extended Abstracts, pp.554-555, ACM, 2002.

[77] Chen H. An Intelligent Broker Architecture for Context-Aware Systems. A Ph. D. Thesis, University of Maryland Baltimore County, January 2003.

[78] Tao Gu, Hung Keng Pung, Daqing Zhang: A service-oriented middleware for building context-aware services. J. Network and Computer Applications, 28(1), pp.1-18, 2005.

[79] Dejene Ejigu, Marian Scuturici, Lionel Brunie: Hybrid Approach to Collaborative Context-Aware Service Platform for Pervasive Computing. JCP, 3(1), pp.40-50, 2008.

[80] B. G. Griswold et al. Using mobile technology to create opportunistic interaction on a university campus. In Proceedings of UBICOMP Workshop on Supporting Spontaneous Interaction in Ubiquitous Computing Settings, September 2002.

[81] Aidan Boran, Ivan Bedini, Christopher J. Matheus, Peter F. Patel-Schneider, John Keeney .A Smart Campus Prototype for Demonstrating the Semantic Integration of Heterogeneous Data. In proceeding of: Web Reasoning and Rule Systems - 5th International Conference, RR 2011, Galway, Ireland, August 29-30, 2011.

[82] Haron NS, Saleem NS, Hasan MH, Ariffin MM, Aziz IA. A RFID-based campus context-aware notification system. J. Comput, 2, pp.122–129, 2010.

[83] Kim, T. H., Shin, D. M., Kim, M. S., Kim, J. S., Ryu, K. H., Shin, M. S. An Ontology based Context Aware System to Control the Robot Manipulator in Delivery Logistics Industry. In Proceedings of the 2nd International Conference on Information Science and Industrial Applications, pp. 221-226, Cebu, May 2012.

[84] Mohcine Madkour, Abdelilah Maach. Ontology-based Context Modeling for Vehicle Context-Aware Services. Journal of Theoretical and Applied Information Technology, 34(2), pp.158-166, December 2011.

ANNEXES

I. Sample Code Segments

//Resource Manager Samples

//**RFID**

```
// Search for ports
ports = CommPortIdentifier.getPortIdentifiers();
while (ports.hasMoreElements(){
   CommPortIdentifier curPort = (CommPortIdentifier)ports.nextElement();
   //example: get only serial ports
   if (curPort.getPortType() == CommPortIdentifier.PORT_SERIAL){
      window.cboxPorts.addItem(curPort.getName());
      portMap.put(curPort.getName(), curPort); }
   }
   //Connect to selected port – using selectedPort variable or assign port identifier
   selectedPortIdentifier = (CommPortIdentifier)portMap.get(selectedPort);
    CommPort commPort = null;
    commPort = selectedPortIdentifier.open("TigerControlPanel", TIMEOUT);
    serialPort = (SerialPort)commPort;
   //start event listener to read data whenever available on the connected port from tags
   serialPort.addEventListener(this);
   serialPort.notifyOnDataAvailable(true);
```

//Bluetooth

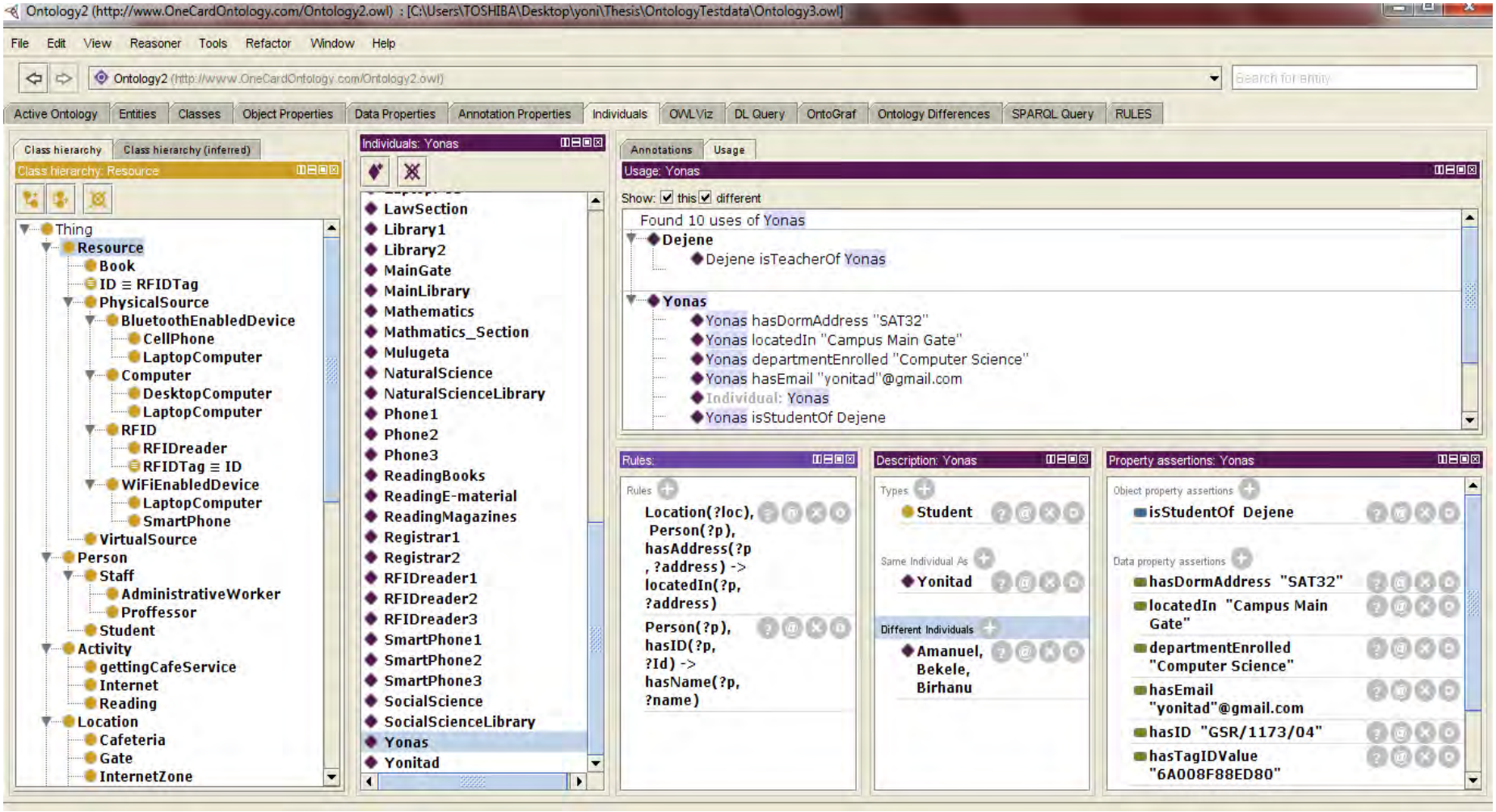
```
LocalDevice localDevice = LocalDevice.getLocalDevice(); //local device information

//find devices
      DiscoveryAgent agent = localDevice.getDiscoveryAgent();
      agent.startInquiry(DiscoveryAgent.GIAC, client);
      synchronized(lock){
       lock.wait();}
//Access devices in vecDevices
    int deviceCount=vecDevices.size();
      for (int i = 0; i <deviceCount; i++) {
       RemoteDevice remoteDevice=(RemoteDevice)vecDevices.elementAt(i);
      // remoteDevice.getBluetoothAddress()
    // remoteDevice.getFriendlyName(true)}
```

//Code segment on reading OWL file and using SPARQL queries to query data

```
  Model model = ModelFactory.createDefaultModel();
  String uri = "file:///C:/Users/TOSHIBA/Desktop/Yoni/Thesis/OntologyTestData/Ontology2.owl";
  InputStream in = FileManager.get().open( uri );
  if (in == null) {
    throw new IllegalArgumentException("File: " + uri + " not found");
  }
model.read(in, null);
model.write(System.out);
    String sparqlQueryString1= "PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>" +
          "PREFIX owl: <http://www.w3.org/2002/07/owl#>" +
          "PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>"+
          "PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>"+
          "PREFIX student: <http://www.OneCardOntology.com/Ontology2.owl#>"+
          "select distinct ?Property ?Value where {"+
          "student:Yonas ?Property ?Value ."+
          "filter ( ?Property not in ( rdf:type ) )"+
          "}";
     Query query = QueryFactory.create(sparqlQueryString1);
     QueryExecution qexec = QueryExecutionFactory.create(query, model);
     ResultSet results = qexec.execSelect();
```

II. Demonstration sample ontology and context data

Sample protégé screenshot showing ontology definition for identification and library notification scenarios



Sample Ontology from Protégé

```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [
<!ENTITY owl "http://www.w3.org/2002/07/owl#" >
<!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
<!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
<!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" > ]>
<rdf:RDF
xmlns="http://www.CASMAOntology.com/OneCard.owl#"
xml:base="http://www.CASMAOntology.com/OneCard.owl"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
<owl:Ontology
//////////////////////////////////////////////////////////////////////////////////
   // Object Properties
//////////////////////////////////////////////////////////////////////////////////
   <rdf:Description rdf:about="bookBelongsTo"/>
   <owl:ObjectProperty rdf:about="engagedIn">
      <rdfs:range rdf:resource="Activity"/>
      <rdfs:domain rdf:resource="Person"/>
   </owl:ObjectProperty>
   <owl:ObjectProperty rdf:about="hasID">
   </owl:ObjectProperty>
   <owl:ObjectProperty rdf:about="hasProgramType">
      <rdfs:range rdf:resource="Program"/>
      <rdfs:domain rdf:resource="Student"/>
   </owl:ObjectProperty>
   <owl:ObjectProperty rdf:about="hasSection">
      <rdfs:domain rdf:resource="Library"/>
      <rdfs:range rdf:resource="Sections"/>
   </owl:ObjectProperty>
   <owl:ObjectProperty rdf:about="isStudentOf">
      <rdfs:range rdf:resource="Proffessor"/>
      <rdfs:domain rdf:resource="Student"/>
      <owl:inverseOf rdf:resource="isTeacherOf"/>
   </owl:ObjectProperty>
   <owl:ObjectProperty rdf:about="isTeacherOf"/>
   <owl:ObjectProperty rdf:about="locatedIn">
      <rdfs:range rdf:resource="Location"/>
      <rdfs:domain rdf:resource="Person"/>
   </owl:ObjectProperty>
   <owl:ObjectProperty rdf:about="ownedBy">
      <owl:inverseOf rdf:resource="owns"/>
   </owl:ObjectProperty>
   <owl:ObjectProperty rdf:about="owns">
      <rdfs:domain rdf:resource="Person"/>
      <rdfs:range rdf:resource="PhysicalSource"/>
   </owl:ObjectProperty>
   <rdf:Description rdf:about="requires"/>
//////////////////////////////////////////////////////////////////////////////////
    // Data properties
//////////////////////////////////////////////////////////////////////////////////
   <rdf:Description rdf:about="departmentEnrolled"/>
   <rdf:Description rdf:about="hasAuthor"/>
   <rdf:Description rdf:about="hasDormAddress"/>
   <rdf:Description rdf:about="hasEmail"/>
   <rdf:Description rdf:about="hasID"/>
   <rdf:Description rdf:about="hasTagIDValue"/>
   <rdf:Description rdf:about="hasTitle"/>
   <owl:DatatypeProperty rdf:about="locatedIn">
   </owl:DatatypeProperty>
   <owl:DatatypeProperty rdf:about="teachesCourse">
      <rdfs:domain rdf:resource="Proffessor"/>
      <rdfs:range rdf:resource="&xsd;string"/>
   </owl:DatatypeProperty>

//////////////////////////////////////////////////////////////////////////////////
   // Classes
//////////////////////////////////////////////////////////////////////////////////
   <owl:Class rdf:about="Activity"/>
   <owl:Class rdf:about="AdministrativeWorker">
      <rdfs:subClassOf rdf:resource="Staff"/>
   </owl:Class>
   <owl:Class rdf:about="BluetoothEnabledDevice">
      <rdfs:subClassOf rdf:resource="PhysicalSource"/>
   </owl:Class>
   <owl:Class rdf:about="Book">
      <rdfs:subClassOf rdf:resource="Resource"/>
   </owl:Class>
   <owl:Class rdf:about="BookBorrowedTime">
      <rdfs:subClassOf rdf:resource="Time"/>
   </owl:Class>
   <owl:Class rdf:about="BookIssueReturn">
<rdfs:subClassOf rdf:resource="CheckPointNotification"/>
   </owl:Class>
   <owl:Class rdf:about="BookReturnDeadline">
      <rdfs:subClassOf rdf:resource="Time"/>
   </owl:Class>
   <owl:Class rdf:about="CellPhone">
<rdfs:subClassOf rdf:resource="BluetoothEnabledDevice"/>
   </owl:Class>
   <owl:Class rdf:about="CheckPointNotification">
      <rdfs:subClassOf rdf:resource="LibraryService"/>
      <rdfs:subClassOf rdf:resource="Notification"/>
   </owl:Class>
   <owl:Class rdf:about="Course">
      <rdfs:subClassOf rdf:resource="Service"/>
   </owl:Class>
   <owl:Class rdf:about="Department">
      <rdfs:subClassOf rdf:resource="College"/>
   </owl:Class>
   <owl:Class rdf:about="DesktopComputer">
      <rdfs:subClassOf rdf:resource="Computer"/>
   </owl:Class>
   <owl:Class rdf:about="EntranceIdentification">
      <rdfs:subClassOf rdf:resource="Identification"/>
   </owl:Class>
   <owl:Class rdf:about="Gate">
      <rdfs:subClassOf rdf:resource="Location"/>
   </owl:Class>
```

```
    <owl:Class rdf:about="ID">
        <owl:equivalentClass rdf:resource="RFIDTag"/>
        <rdfs:subClassOf rdf:resource="RFID"/>
        <rdfs:subClassOf rdf:resource="Resource"/>
    </owl:Class>
    <owl:Class rdf:about="Identification">
        <rdfs:subClassOf rdf:resource="Service"/>
    </owl:Class>
    <owl:Class rdf:about="InternetZone">
        <rdfs:subClassOf rdf:resource="Location"/>
    </owl:Class>
    <owl:Class rdf:about="Laboratory">
        <rdfs:subClassOf rdf:resource="Location"/>
    </owl:Class>
    <owl:Class rdf:about="LaptopComputer">
        <rdfs:subClassOf
rdf:resource="BluetoothEnabledDevice"/>
        <rdfs:subClassOf rdf:resource="Computer"/>
        <rdfs:subClassOf rdf:resource="WiFiEnabledDevice"/>
    </owl:Class>
    <owl:Class rdf:about="LastServiceAccessTime">
        <rdfs:subClassOf rdf:resource="Time"/>
    </owl:Class>
    <owl:Class rdf:about="Library">
        <rdfs:subClassOf rdf:resource="Location"/>
    </owl:Class>
    <owl:Class rdf:about="LibraryNotification">
        <rdfs:subClassOf rdf:resource="LibraryService"/>
        <rdfs:subClassOf rdf:resource="Notification"/>
    </owl:Class>
    <owl:Class rdf:about="LibraryService">
        <rdfs:subClassOf rdf:resource="Service"/>
    </owl:Class>
    <owl:Class rdf:about="Library_Identification">
        <rdfs:subClassOf rdf:resource="Identification"/>
    </owl:Class>
    <owl:Class rdf:about="Location"/>
    <owl:Class rdf:about="NaturalScienceSection">
        <rdfs:subClassOf rdf:resource="Sections"/>
    </owl:Class>
    <owl:Class rdf:about="NewBookArrivals">
        <rdfs:subClassOf rdf:resource="LibraryNotification"/>
    </owl:Class>
    <owl:Class rdf:about="NewServiceAccessTime">
        <rdfs:subClassOf rdf:resource="Time"/>
    </owl:Class>
    <owl:Class rdf:about="Notification">
        <rdfs:subClassOf rdf:resource="Service"/>
    </owl:Class>
    <owl:Class rdf:about="Office">
        <rdfs:subClassOf rdf:resource="Location"/>
    </owl:Class>
    <owl:Class rdf:about="Person"/>
    <owl:Class rdf:about="PhysicalSource">
        <rdfs:subClassOf rdf:resource="Resource"/>
    </owl:Class>
    <owl:Class rdf:about="Proffessor">
        <rdfs:subClassOf rdf:resource="Staff"/>
    </owl:Class>
    <owl:Class rdf:about="Program"/>
    <owl:Class rdf:about="RFID">
        <rdfs:subClassOf rdf:resource="PhysicalSource"/>
    </owl:Class>
    <owl:Class rdf:about="RFIDTag">
        <rdfs:subClassOf rdf:resource="RFID"/>
    </owl:Class>
    <owl:Class rdf:about="RFIDreader">
        <rdfs:subClassOf rdf:resource="RFID"/>
    </owl:Class>
    <owl:Class rdf:about="Registrar">
        <rdfs:subClassOf rdf:resource="Office"/>
    </owl:Class>
    <owl:Class rdf:about="Resource"/>
    <owl:Class rdf:about="Sections">
        <rdfs:subClassOf rdf:resource="Library"/>
    </owl:Class>
    <owl:Class rdf:about="Service"/>
    <owl:Class rdf:about="SmartPhone">
        <rdfs:subClassOf rdf:resource="WiFiEnabledDevice"/>
    </owl:Class>
    <owl:Class rdf:about="SocialScienceSection">
        <rdfs:subClassOf rdf:resource="Sections"/>
    </owl:Class>
    <owl:Class rdf:about="Staff">
        <rdfs:subClassOf rdf:resource="Person"/>
    </owl:Class>
    <owl:Class rdf:about="Student">
        <rdfs:subClassOf rdf:resource="Person"/>
    </owl:Class>
    <owl:Class rdf:about="TechnologySection">
        <rdfs:subClassOf rdf:resource="Sections"/>
    </owl:Class>
    <owl:Class rdf:about="Time"/>
    <owl:Class rdf:about="VirtualSource">
        <rdfs:subClassOf rdf:resource="Resource"/>
    </owl:Class>
    <owl:Class rdf:about="WiFiEnabledDevice">
        <rdfs:subClassOf rdf:resource="PhysicalSource"/>
    </owl:Class>
    ///////////////////////////////////////////////////////////////////////////////////
    // Individuals
    ///////////////////////////////////////////////////////////////////////////////////
    <owl:NamedIndividual rdf:about="LaptopPC1">
        <rdf:type rdf:resource="BluetoothEnabledDevice"/>
        <rdf:type rdf:resource="Computer"/>
        <rdf:type rdf:resource="LaptopComputer"/>
        <rdf:type rdf:resource="PhysicalSource"/>
        <rdf:type rdf:resource="Resource"/>
        <rdf:type rdf:resource="WiFiEnabledDevice"/>
    </owl:NamedIndividual>
    <owl:NamedIndividual rdf:about="LawSection">
        <rdf:type rdf:resource="Library"/>
        <rdf:type rdf:resource="Location"/>
        <rdf:type rdf:resource="Sections"/>
        <rdf:type rdf:resource="SocialScienceSection"/>
    </owl:NamedIndividual>
```

```
    <owl:NamedIndividual rdf:about="Library1">
        <rdf:type rdf:resource="Library"/>
        <rdf:type rdf:resource="Location"/>
    </owl:NamedIndividual>
    <owl:NamedIndividual rdf:about="MainGate">
        <rdf:type rdf:resource="Gate"/>
        <rdf:type rdf:resource="Location"/>
    </owl:NamedIndividual>
    <owl:NamedIndividual rdf:about="MainLibrary">
        <rdf:type rdf:resource="Library"/>
        <rdf:type rdf:resource="Location"/>
    </owl:NamedIndividual>
<owl:NamedIndividual rdf:about="Mathematics">
        <rdf:type rdf:resource="College"/>
        <rdf:type rdf:resource="Department"/>
        <rdf:type rdf:resource="Location"/>
        <rdf:type rdf:resource="Office"/>
    </owl:NamedIndividual>
    <owl:NamedIndividual rdf:about="Mathmatics_Section">
        <rdf:type rdf:resource="Library"/>
        <rdf:type rdf:resource="Location"/>
        <rdf:type rdf:resource="NaturalScienceSection"/>
        <rdf:type rdf:resource="Sections"/>
    </owl:NamedIndividual>
    <owl:NamedIndividual rdf:about="NaturalScience">
        <rdf:type rdf:resource="College"/>
        <rdf:type rdf:resource="Location"/>
        <rdf:type rdf:resource="Office"/>
    </owl:NamedIndividual>
    <owl:NamedIndividual
rdf:about="NaturalScienceLibrary">
        <rdf:type rdf:resource="Library"/>
        <rdf:type rdf:resource="Location"/>
    </owl:NamedIndividual>
    <owl:NamedIndividual rdf:about="RFIDreader1">
        <rdf:type rdf:resource="PhysicalSource"/>
        <rdf:type rdf:resource="RFID"/>
        <rdf:type rdf:resource="RFIDreader"/>
        <rdf:type rdf:resource="Resource"/>
    </owl:NamedIndividual>
    <owl:NamedIndividual rdf:about="RFIDreader2">
        <rdf:type rdf:resource="PhysicalSource"/>
        <rdf:type rdf:resource="RFID"/>
        <rdf:type rdf:resource="RFIDreader"/>
        <rdf:type rdf:resource="Resource"/>
    </owl:NamedIndividual>
    <owl:NamedIndividual rdf:about="SocialScience">
        <rdf:type rdf:resource="College"/>
        <rdf:type rdf:resource="Location"/>
        <rdf:type rdf:resource="Office"/>
    </owl:NamedIndividual>
    <owl:NamedIndividual rdf:about="Yonas">
        <rdf:type rdf:resource="Person"/>
        <rdf:type rdf:resource="Student"/>
<hasTagIDValue
rdf:datatype="&xsd;string">6A008F88ED80</hasTagIDValu
e>
<departmentEnrolled    rdf:datatype="&xsd;string">Computer
Science</departmentEnrolled>
 <hasID rdf:datatype="&xsd;string">GSR/1173/04</hasID>
<hasDormAddress
rdf:datatype="&xsd;string">SAT32</hasDormAddress>
<hasID>GSR/1173/04</Ontology2:hasID>
<hasTagIDValue>6A008F88ED80</hasTagIDValue>
<departmentEnrolled>Computer
Science</departmentEnrolled>
        <locatedIn>Campus Main Gate</locatedIn>
        <hasDormAddress>SAT32</hasDormAddress>
        <hasEmail xml:lang="gmail.com">yonitad</hasEmail>
        <isStudentOf rdf:resource="Dejene"/>
    </owl:NamedIndividual>
</rdf:RDF>
```

III. Demonstration sample rules

```
SWRL Rules:
     -->
```

DLSafeRule(Body(ClassAtom(CampusRule:Person Variable(<urn:swrl#x>))
ObjectPropertyAtom(CampusRule:enrolledIn Variable(<urn:swrl#x>)
Variable(<urn:swrl#y>)) ObjectPropertyAtom(CampusRule:worksIn Variable(<urn:swrl#x>)
Variable(<urn:swrl#y>)))Head(ClassAtom(CampusRule:Student Variable(<urn:swrl#x>))
ClassAtom(CampusRule:Staff Variable(<urn:swrl#x>))))

DLSafeRule(Body(ClassAtom(CampusRule:Computer Variable(<urn:swrl#x>))
ClassAtom(CampusRule:Person Variable(<urn:swrl#p>))
DataPropertyAtom(CampusRule:hasMacID Variable(<urn:swrl#x>)
Variable(<urn:swrl#y>)))Head(DataPropertyAtom(CampusRule:belongsTo
Variable(<urn:swrl#p>) Variable(<urn:swrl#y>))))

DLSafeRule(Body(ClassAtom(CampusRule:Cafeteria Variable(<urn:swrl#c>))
ClassAtom(CampusRule:Person Variable(<urn:swrl#p>))
ObjectPropertyAtom(CampusRule:locatedIn Variable(<urn:swrl#p>)
Variable(<urn:swrl#c>)))Head(ObjectPropertyAtom(CampusRule:engagedIn
Variable(<urn:swrl#p>) CampusRule:Meal)))

DLSafeRule(Body(ObjectPropertyAtom(CampusRule:locatedIn Variable(<urn:swrl#s>)
CampusRule:Cafeteria1) ClassAtom(CampusRule:Student
Variable(<urn:swrl#s>)))Head(ObjectPropertyAtom(CampusRule:studentYear
Variable(<urn:swrl#s>) CampusRule:Fresh)))

DLSafeRule(Body(ObjectPropertyAtom(CampusRule:locatedIn Variable(<urn:swrl#p>)
CampusRule:Cafeteria2) ClassAtom(CampusRule:Person
Variable(<urn:swrl#p>)))Head(ClassAtom(CampusRule:Staff Variable(<urn:swrl#p>))))

DLSafeRule(Body(ClassAtom(CampusRule:Student Variable(<urn:swrl#s>))
ObjectPropertyAtom(CampusRule:labBelongsTo Variable(<urn:swrl#x>)
CampusRule:ComputerScience) DataPropertyAtom(CampusRule:located
Variable(<urn:swrl#s>)
Variable(<urn:swrl#x>)))Head(ObjectPropertyAtom(CampusRule:isStudentOf
Variable(<urn:swrl#s>) CampusRule:ComputerScience)))

DLSafeRule(Body(ObjectPropertyAtom(CampusRule:worksIn Variable(<urn:swrl#x>)
Variable(<urn:swrl#y>)) ClassAtom(CampusRule:Person
Variable(<urn:swrl#x>)))Head(ClassAtom(CampusRule:Staff Variable(<urn:swrl#x>))))

DLSafeRule(Body(ClassAtom(CampusRule:Library Variable(<urn:swrl#lib>))
ClassAtom(CampusRule:Student Variable(<urn:swrl#s>))
ObjectPropertyAtom(CampusRule:locatedIn Variable(<urn:swrl#s>)

Variable(<urn:swrl#lib>)))Head(ObjectPropertyAtom(CampusRule:engagedIn Variable(<urn:swrl#s>) Variable(<urn:swrl#r>))))

DLSafeRule(Body(ObjectPropertyAtom(CampusRule:locatedIn Variable(<urn:swrl#p>) CampusRule:Library2) ClassAtom(CampusRule:Person Variable(<urn:swrl#p>)))Head(ObjectPropertyAtom(CampusRule:hasAccessTo Variable(<urn:swrl#p>) Variable(<urn:swrl#int>)) ClassAtom(CampusRule:Internet Variable(<urn:swrl#int>))))

DLSafeRule(Body(ObjectPropertyAtom(CampusRule:locatedIn Variable(<urn:swrl#s>) CampusRule:MainGate) ClassAtom(CampusRule:Student Variable(<urn:swrl#s>)))Head(ObjectPropertyAtom(CampusRule:serviceType Variable(<urn:swrl#s>) CampusRule:entranceIdentification)))

DLSafeRule(Body(ClassAtom(CampusRule:Book Variable(<urn:swrl#x>)) ClassAtom(CampusRule:NaturalScienceSection Variable(<urn:swrl#y>)) ObjectPropertyAtom(CampusRule:locatedInSection Variable(<urn:swrl#x>) Variable(<urn:swrl#y>)))Head(ClassAtom(CampusRule:NaturalScienceBooks Variable(<urn:swrl#x>))))

DLSafeRule(Body(ObjectPropertyAtom(CampusRule:enrolledIn Variable(<urn:swrl#x>) Variable(<urn:swrl#y>)) ClassAtom(CampusRule:Person Variable(<urn:swrl#x>)))Head(ClassAtom(CampusRule:Student Variable(<urn:swrl#x>))))

)

DECLARATION

I, the undersigned, declare that this research is my original work and has not been presented for degree in any other university, and that all sources of materials used for the research have been acknowledged.

Declared by:

Name: **Yonas Tadesse Woldegebriel**

Signature: _____

Date: _____

Confirmed by advisor:

Name: **Dr. Dejene Ejigu**

Signature: _____

Date: _____

Place and date of submission: Addis Ababa University, May 2014.