ADDIS ABABA UNIVERSITY
SCHOOL OF GRADUATE STUDIES
FACULITY OF INFORMATICS
DEPARTMENT OF COMPUTER SCIENCE

CONTEXT AWARE DIARY BUILDER (CADB)

By: **Beza Mamo Rabdo**

A THESIS SUBMITTED TO
THE SCHOOL OF GRADUATE STUDIES OF THE ADDIS ABABA UNIVERSITY IN
PARTIAL FULFILLMENT FOR THE DEGREE OF MASTERS OF SCIENCE
IN COMPUTER SCIENCE

October, 2009
ADDIS ABABA

ADDIS ABABA UNIVERSITY
SCHOOL OF GRADUSTE STUDIES
FACULITY OF INFORMATICS
DEPARTMENT OF COMPUTER SCIENCE

CONTEXT AWARE DIARY BUILDER (CADB)

By: Beza Mamo Rabdo

ADVISOR:
Dejene Ejigu (PhD)

APPROVED BY

EXAMINING BOARD:

1. Dr. Dejene Ejigu, Advisor _____
2. _____ _____
3. _____ _____

Dedication

- ☞ To My Mother "***tati***"
- ☞ To My Wife "***ADi***"
- ☞ To My Brother "***Mudidi***"

Acknowledgements

Be grateful Jesus! You are really a path for many of us in earth. I glorify your name through this accomplishment you are a means and an end to me.

Many thanks to my advisor Dr. Dejene Ejigu, it is your directions and continuous follow-up (that you made every Monday) makes this work to be concluded like this. I taught a lot of bundled experience from your critical and scientific observation regarding this work.

I wish to extend my deep gratitude to my family members including: Hiwot Mesele (mama), Mahdere Mamo (younger brother), Admas Mamo (my lovely wife) and to the entire family members of Emhay Genet W/Tsadik.

Biniyam Asfaw, Shiferahu Abebe, Bethelihem Nega, Henok Sahilu, Fiakdu Wanaw, Melese Tamiru, Mohammed-Hussen M., Yared Bezu and other friends not mentioned in this list but having a lot of contribution for the realization of this work in one or another way I said "*God Bless you!"*.

"*Thank you*" all my instructors (staff of Addis Ababa University Computer Science Department). Finally, special thanks to Ethiopian Civil Service College for granting my tuition fee in the course of the past two years.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

AA : Application Agent
ABM : Agent Based Modeling
ACC : Agent Communication Channel
AM : Agent Manager
AMS : Agent Management System
AP : Agent Platform
API : Application Program Interface
CADB : Context Aware Diary Builder
CADB-CS : CADB-Core Service
CADB-MAS : CADB-Multi-Agent System
CBA : Context Binder Agent
CBD : Computer Based Diary
CIS : Context Information Service
CKDD : Context Knowledge Discovery
CLA : Context Level Agreement
CM : Communication Manager
CPA : Context Provider Agent
DF : Directory Factory
EA : Extended Agent
ED : Episode Discovery
AA : Application Agent
GPS : Global Positioning System
HCOM : Hybrid Context Management
HTTP : HyperText Transfer Protocol
KA : Knowledge Agent
KDD : Knowledge Discovery
MANET : Mobile Ad-hoc Network
MAS : Multi-Agent System
MD : Manual Diary
OD : Online Diary
ACAI : Agent-based Context-Aware Infrastructure
CAMUS : Context-Aware Middleware for URC System
EHRAM Entity, Hierarchies, Relationship, Axiom and Metadata Model
MySQL : An open source relational database management system. It is based on the structure query language (SQL)
SPARQL : **S**PARKL **P**rotocol **A**nd RDF **Q**uery **L**anguage
OWL : Web Ontology Language
PDA : Personal Digital Assistant
PHP : HyperText Processor
RDF : Resource Description Framework
RDFS : Schema for RDF
RFID : Radio Frequency Identification
TMM : Task-based Markov Model
URC : Ubiquitous Robotic Companion
WAMP : Window Apache Mysql PHP
XML : Extensible Markup Language

Abstract

The major attempt of pervasive computing is to reduce the required user effort in using applications in their surrounding by identifying and recognizing resources; and reacting automatically based on their interest. Spontaneous interaction of heterogeneous resources to achieve context awareness requires the detection of entities together with its description in smart space environment (such as location, status, capability, history, etc). Once resources in smart space environment are identified; application developer can produce pervasive application that is required in the daily life of the society.

There are a number of researches in pervasive computing that aim at improving our daily life. This research work emphasizes on producing and introducing a new way of diary capturing for users in their day-to-day activities. Shortly we claim that our research work is a technology driven study as a result of advancement in smart space environment and mobile computing competence.

This thesis work, proposed an architecture for Context Aware Diary Builder (CADB) along with a number of algorithms used to realize it. The architecture of CADB consists of four layers; i) Acquisition Layer (deals with extraction of context information from various sensors and user's profile); ii) Context Aggregator Layer (deals with characterization of sensor data via a preprocessor, feature extractor and context classifier components); iii) Context Reasoner Layer (reasoning on the user's context by using ontology and rule based reasoner); and iv) Application layer (deals with how user's view their daily activity annexed with date information).

Moreover, this study indicates a new way of context agreement/negotiation deployed at the second layer of the CADB architecture. It was used a multi-agent system to achieve context agreement/negotiation. Six software agents were introduced and used to facilitate and simplify the diary writing process.

Key Word: Pervasive Computing, Diary, Multi-Agent-System, Context Aware Diary Builder

1 Introduction

1.1 Background

Currently, mobile devices are rich in knowledge of their context [24, 41] and play an important role in many application domains. According to [24, 29], a context is any situation of an entity such as person, place, event, and object. For example, considering an office environment, context can refer to: Identity –of an entity; Location; Service –function offered by the entity; and Event –asserted by specific time of day.

Contexts provide information about the present status of people, places, things and devices. Literally speaking, contexts include: location, time activity and preference of each entity. However the introduction of spontaneous application introduces new requirement and challenges. Semantically rich information (location, identity and activity), context management (require by certain events to trigger actions) and context dependency (relationship between different aspect of context information) are some of the requirements for dynamic and adaptive provisioning [48].

Spontaneous phenomena, arising from natural interactions: and they are controlled and directed internally, localized, and handled without previous planning. Recent researches (including [13] and [30]) suggest that spontaneous interactions are: interaction between devices (and resources) that do not have prior knowledge of each other. More specifically, it refers to devices that are not explicitly designed to work with each other and that do not require pre-configuration to work with each other. The work of Lim B. Y., *et al*, [38] allows mobile users to spontaneously discover services without any specialized installation. Interaction that can happen spontaneously without administrative overhead, precisely: they are unplanned (encountered opportunities) and “self-acting” (plug and play).

Guanling Chen and Dauid Kitz [9] also discussed about the nature of pervasive-computing environment as crowded, heterogeneous and always changing. Thus, pervasive-computing applications must discover and use resources based on current context of entities. Accordingly, they define context as:

“Circumstance in which an application runs, and may include physical state, computational state, and user state.”

An application that adapts to changing environments are considered as context aware applications and exhibit the following list of features as stated in [15, 16]:

- Presentation of information and services to user;
- Automatic execution of a service for a user; and
- Tagging of context information to support later retrieval

Pervasive computing environments are characterized by high heterogeneity of user, devices, and service. The user is highly involved in adaption of the context environment at least during the first time. User preference, context data, and knowledge about services are some of the information to be maintained [26].

As it is clearly indicated in [13], context information ranges from low-level context such as: temperature, noise level, location coordinates, etc to high-level context such as: activity scheduling, event profile, etc. High-level context information can be obtained from the various data source by logic and reasoning mechanism. In short, context information can be categorized as either physical context (which is a low-level context) or semantic context (for example activity within a room). Currently a number of smart spaces[1] are existing including; smart homes, smart offices, smart workplaces, smart classrooms and smart vehicles that provide/utilize different types of semantic information. Furthermore, within a workplace, there are atomic locations that can contain a desk, a PC, a table, a shelf or it can be empty. All these atomic locations can be identified easily by using a combination of sensing technologies such as WiFi (Wireless Fidelity) and RFID (Radio Frequency Identification).

An ontology based formal Context Engine (CE) architecture in [29] describes a critical design issue of context-aware architecture for smart spaces including semantic representation, logic-based reasoning and knowledge sharing. As shown in [13], semantic context service provides context information by using diverse type of data sources and sensing techniques to integrate context information characterized by a specific format.

On the other hand, the MobiLife project [24] provides context-aware and proactive services and applications. The Reconfigurable Context Sensitive Middleware (RCSM) adapts its object discovery and connection management mechanisms, depending on the context-sensitive behavior

[1] A smart space is a space that is equipped with sensors and diversified network for resource identification.

of the applications. In particular, the MobiLife project intended to advance in mobile application and services to attain users in their everyday life phenomenon.

Information associated with locations is most important for context determination and can be exploited by location-aware mobile systems. The TEA project (Technology for Enabling Awareness) point out that context of personal mobile devices can be asserted from multi-sensor input [41]. The Mediacup project on the same article [41], describes how to embed awareness technology to obtain a rich context on everyday activity. Numerous sets of sensors and perception techniques are integrated to achieve context awareness from algorithmic level to architectural level. However, the TEA project devices basic concepts including: i) Integration of diverse sensors, for acquisition of multi-sensor data; ii) Association of multi-sensor data and iii) Implementation of hardware- i.e. methods for computing situational context sensors in an embedded device. Accordingly, application builders can determine what behavior or features they want to support their applications and what context is required to achieve the desired behaviors.

Currently, two recent trends highly facilitate the vision of ubiquitous computing: wireless hotspots and the explosion of mobile devices. [2, 30] point out networking technology facilitates, the spontaneous connection of user personal devices to encounter (immediate) services seamlessly. But the difficulty lies on knowing the available services in the user environment. On the other hand, [56] indicates that even if cell phones are found everywhere, different mobile devices exhibit to offer different functionality and have different CPU speed, memory capacity and power. Furthermore, mobile devices make use of different sensing mechanisms such as GPS, Wi-Fi, Bluetooth, and others use a combination of listed mechanisms, or neither. Additional differences among mobile devices are also inevitable with respect to support of programming languages for application developers. For example, Nokia 95 use Java and C++ and Apple's iPhones support JavaScript and AJAX only as realized form [56]. Thus, application developed in one device might or might not be run on others.

In summary, a pervasive (ubiquitous) system is characterized by: physical integration (integration between computing nodes and the physical world) and instantaneous interoperation (devices interoperate spontaneously in changing environments, e.g., a device changes its partners as it moves or as the context changes) [35].

1.2 Motivation

The explosion of mobile devices and also the advancement in wireless communication are the major source for the reality of pervasive computing. Pervasive or ubiquitous computing is a vision of Weiser [42]. More specifically, pervasive computing environments are packed with several computing resources and also have high communication facility, which are interlinked with human user in their daily life.

Modern homes and offices are equipped with quite a lot of sensors and wireless hotspots, as it can be learned from [4, 17]. Hand held devices are also getting advanced from time to time and hence previous problems associated with processing; memory and the like are scaled-up. Most research in IBM lies for better daily life in home and office environment, one best example of such research is the work of Chio P. *et al*. [10] which emphasizes on the formation of a personalized and context aware work space (BlueSpace). The figure 1.1 depicts BlueSpace user environment for context awareness perspective by tracing atomic location and services. In this case every atomic location and services can be used for defining context knowledge of user environment.

This research work assumes such environment, given in figure 1.1 in which pervasive computing is no more a dream. Such environment is promising to trace user atomic location and activity via different sensors embedded in user's working space. The daily activity of the user can be easily captured and interpreted so as to build user activity diary in a usable form and hence increase productivity, satisfaction, behavior correction and motivation of user progress in their daily life. In the context of this study, a diary is defined as a daily report of individual activities together with emotional expression about the event.

Most of us wish to have a diary, but the following points make it cumbersome:

- Maintaining all the data every day;
- Take time to recall what we did in the past;
- Lack of interest to build personal profile as the activity requires dedication;
- Wishing to recall only few wonderful moments and ignoring all the others;
- Using it for building and follow up our progresses in whatever aspect; and
- Some other personal constraints can be added to this list.

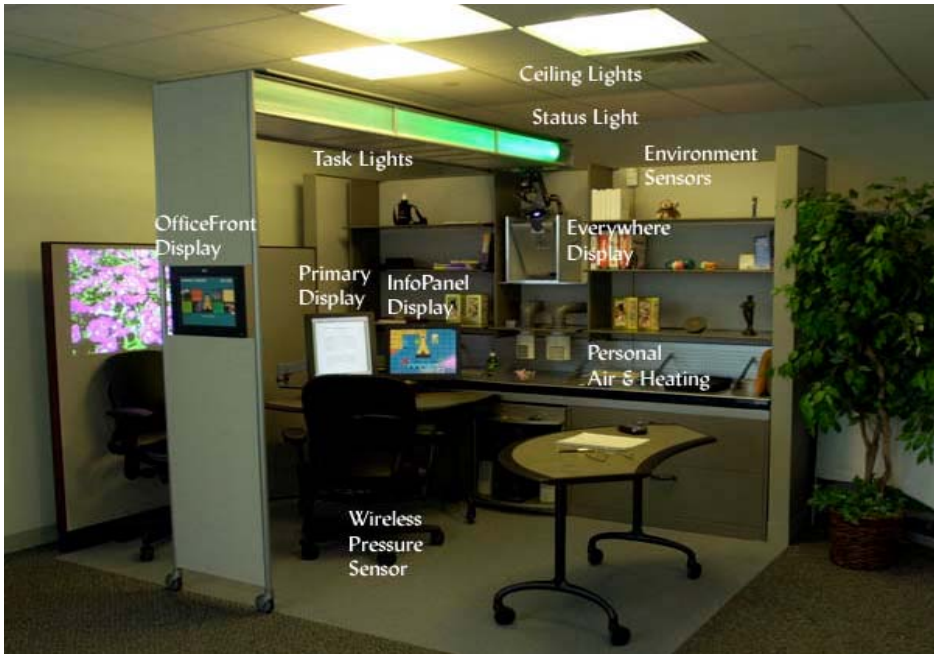


Figure 1-1: BlueSpace Environments

Thus, only a small number of people maintain their daily activities through a manual diary writing mechanisms and some are getting started to have digital diary (i.e., computer based and online diary). Some of the common limitation of the exiting diary writing approach includes: identification of exact location, referring to global time, and producing relationship with previous events (or activities of a user). In addition to this event retrieval, searching and some other activities related to stored data manipulations require a considerable time.

Accordingly, this research work is a technology driven study as a result of advancement in pervasive technology environments. The application assumes to encourage most people to capture their daily routine activity so as to have a more informed direction about their life, situations around them, etc.

Despite of the activity a person is engaged in; the most important, common, and valuable information of a diary is location, date and time. These three attribute of any diary can be easily captured using ubiquitous (pervasive) technology in automatic fashion and could be annotated with user's emotional expressions about whatever sort of event they experience through. Location of users, where about of system components, and local availability of resources are some of the

aspects which are related to mobile system environments [41]; and hence it is also the core motivation behind this work.

Hence, this research work is an attempt to come up with an architecture with which diary building can be done in automatic fashion with less distraction of user by utilizing a number of pervasive computing approaches.

1.3 Statement of the Problem

Diary writing has a lot of significance for the society. Some of the benefit of diary writing includes: healing personalities, meditation, provides emotional stability, provides place for dreams and ideas to grow, and provides insight of self-experiences as learned from [68].

The existing methods of diary writing require the devotion or high-involvement of individuals. Both manual diary and digital diary (i.e., computer based or online diary) demands the dedication of individuals to compose their day-to-day routine activities. On the other hand, pervasive or ubiquitous computing reduce the required user effort in using applications in their surrounding by identifying and recognizing resources; and reacting automatically based on their interest.

This study tries to answer the following questions:

i. How to avail a context-aware diary builder that reduces the effort of individuals while capturing their day-to-day activities by determining and being aware of the user environment?

ii. How context-awareness is is created?

Dey A. and Abowd G. [16] describes context as: who's, where's, when's, and what's. This includes location, identity, activity, and time. Here the problems are precision of context aware data, prioritization of user's expected activity based on previous histories and the use of agent technology to support context prediction.

In summary, the research investigates a number of issues in pervasive computing from context acquisition to application development in producing a context aware diary builder. The development of context aware diary also introduces new way of multi-agent agreement/negotiation at the lower layer (i.e., during context aggregation) as pointed out in this study.

1.4 Objective

1.4.1 General Objective

The general objective of this research work is to design a model and develop a prototype for context-aware diary builder by addressing the problems stated above.

1.4.2 Specific Objectives

The specific objectives of this study include:-

- Study the nature of pervasive computing;
- Develop a method for acquiring and adapting user environment in automatic way;
- Develop/select appropriate algorithm and techniques in relation to the problem domain;
- Develop a technique to attain static and dynamic context binding;
- Build an architecture that can be used to realize context aware diary builder and make use of MAS to attain context level agreement/negotiation ;
- Develop a context aware diary builder prototype; and
- Implement a prototype that demonstrates the acquisition and utilization of user's activity for the purpose of building context aware personal diary.

1.5 Scope and Limitation

The availability of resources such as Radio Frequency Identifier (RFID) reader/tag is major limitation of this study. This problem hinders the full realization and deployment of context aware diary builder at large. With this background, the scope of this research work can be stated as follows:

- Developing method for user context determination such as location, activity and available resources;
- How to use static context information in order to increase the knowledge of user environment and associated activities; and
- Knowledge representation for diary writing process; and

- Use of agent technology for the development of intelligent system in smart environment.

Generally, the major focus of this research work is to propose a model that reduces the effort of building a diary by monitoring the context of the user.

1.6 Methodology

A brief explanation of the methodology will be elaborated under this section. The following lists of methodology are used to realize context aware diary builder:

a) Literature Review and Related Works:

In the course of this study, a number of published research papers; complete thesis works; books and web sites in the area of pervasive computing in general and in the act of diary building process in particular are examined.

b) Tools and technologies:

Exploring the required tools and technology to avail Context aware diary builder, the tools and technology we have used in the course of this research will be presented in Chapter 5 of this documentation.

c) Simulation:

To substitute the RFID reader/tagger, we developed RFID based sensor simulator using a Java code that provides location information from three RFID readers for tagged resources in the campus scenario.

d) Prototyping and validation:

To validate the proposed architecture and algorithms suggested in this work, we used a prototype as a means of proving the concepts.

1.7 Application of the Study

The result of this study can span from activity tracking up to context identification and sensing. Some of the ongoing researches under activity tracking that could be highly related with this research work include [28, 39, 60]:

- Health activity tacking systems;
- Prisoner and correctional facilities tracking systems;
- Activity based and task level computing;
- Child activity tracking systems; and
- User activity tracking systems in general.

Additionally, this research work will provide a number of applications for the community and individuals. Some of the applications are:-

- It simplifies the task of diary building mechanism;
- It helps to annotate users habitual daily activities with its environment;
- It helps to conduct research on users behavioral change and the influence of environment (user context) in general;
- It helps to motivate a number of individuals to build a profile of their activities in a daily basis;
- It helps to improve individual performance, efficiency, effectiveness, and the like with regard to the activity she/he is engaged in by providing a summarized report; and
- It helps to know the time spent to complete certain tasks.

1.8 Thesis Organization

The remaining part of this documentation is organized as follows. The next chapter deals with literature review. The literature review part will try to give a summarized report on the state of art in relation to pervasive computing in general. Chapter three is about related works that are highly relevant to our study. Chapter 4 will discuss the architecture that this study proposed in order to build a context aware diary builder. Chapter 5 explains the implementation issues of the context aware diary builder. The last chapter will conclude the paper by leaving recommendation and potential research areas that can be considered as a continuation to this work.

2 Literature Review

A number of papers in pervasive computing research findings including activity tracking, context mining, context middleware, event prediction agent based modeling and context management tools are reviewed in this chapter. Detail of the particular contemporary art work in pervasive computing are discussed and considered in the subsequent sections.

The last section of this chapter provides a concise summary of all the literature that we reviewed and described.

2.1 Activity Tracking

User activity tracking is the core concept behind this study. Some of the ongoing researches under activity tracking that could be highly related with this research includes: healthcare activity tracking systems; prisoners and correctional facilities tracing systems; activity based and task level computing; child activity tracking systems; and user activity tracking in general.

This sub-section specifically discuss activity tracking that are aspired to offer improved health care facilities of a user in their daily life. A number of activity diary researches in relation to medical healthcare have undergone [19, 20, 28]. The notions of such researches are to collect physiological data and user action in order to generate an activity diary of a patient. Some health monitoring devices require the user to keep writing all the activity logo, but these records are described by inaccuracy and/or lack of detail elucidation to make informed medical decisions. Moreover, some context information is not easily reported such as temperature, allergens and barometric pressure as pointed out by [19, 20].

E-textile system [20] is a wearable context sensor system which monitor and records data automatically and intelligently to annotate physiological data with patient's context without the user intervention. On the other hand, physiological data alone is meaningless without additional information concerning patient's activities, environment and experiences [19].

In addition to the physiological data, the system named as, shakra [28] which is used for fitness tracking and motivation runs on a mobile phone to trap individual physical activity level. The physical activities recorded and classified by using back propagation neural network approach. The

current activity of the user is then classified every 30 seconds. The shakra application uses fluctuation in GSM signal and neighboring cell information to determine user's mobility [28, 39]. All the physical activities walking, running, bicycling or driving can be distinguished based on movement, fluctuation between cell ID, and therefore unable to detect stationary movements. As shown in [28], the artificial neural network inputs are: the sum of signal strength fluctuation across all monitored cells and the number of distinct cells over a given time interval. The network consists of a single layer of eight hidden neurons; weights are learnt using back propagation methods.

The above researches [28, 39] emphasize on tracking user's daily activity for better healthcare that helps to maintain routine exercise of an individual. All healthcare pervasive services considered under this section ignore the user privacy issues and the user is monitored by the server side program logic and the user has little control over the resources surrounding him/her.

2.2 Context Mining

This section is highly associated with pervasive event prediction approach. Context mining enables to associate context history with current user context to increase the prediction of user action in general with high accuracy. This section discusses four mining approaches followed by different researchers. Episode Discovery (ED), Mining user Models, Context Knowledge Discovery (CKDD) and Sequential Mobile Access Pattern-mine (SMAP-mine) will be discussed under this section.

A) EPISODE DISCOVERY

Episode Discovery (ED) is a data mining algorithm introduced in MavHome [17] which identifies episodes within an inhabitant event history. Episodes occur at regular interval in response to other significant episodes called triggers. Many of the episodes occur daily or weekly and need to be recognized for this regularity.

Pridict algorithm was identified as a prediction algorithm (meta-predictor) in MavHome. The predict algorithm use a back propagation neural network to learn confidence value for each prediction algorithm based on the gathered data and accuracy of the algorithm on this data along with meta-features. Possible meta-features could be the amount of training data, the number of devices, the number of inhabitants in the home, and the significance of the event as determined by

ED. The final prediction could be generated, using a weighted voting scheme by each individual algorithm.

B) MINING USER MODELS

The mining approach introduced in [58] based on user habits recur when similar phenomenon are observed. Thus past behavior of user will provide useful knowledge for predicting current and future user desire and actions. The mining approach consists of four distinct phases.

Case Selection: → Mining: → Rule Selection: →Adapt

Case Selection: ensure only relevant information is chosen for making adaption decisions, a semantic and context based selection technique were used. Application and object hierarchy structure enable to define semantic similarity, by selecting user model cases that are semantically similar to the current user problem. Context based selection is achieved by chosen set of semantically similar cases, which are similar to current user situation.

Mining: analysis of relevant past user behavior provides useful knowledge for predicting user desires and actions. Behavioral history of a user is evaluated using association mining. The mining major task is identification of user behavior pattern.

Adaption Rule selection: only relevant user model cases are selected using semantic and context based information selection; on the other hand, mining yield a large number of potentially conflicting rules. Adaption rule selection generate rules according to their strength, association mining provides two measures, confidence and support for computing rules. Rule ranking and selection favor stronger rules, the stronger the rule, the more accurate and recurring it.

Adapt: user rules to make adaption decision.

In summary, Tsang1 S. L. and Clarke S [58] come up with a brief generalization of their finding by comparing the mining technique with the common methods such as Neural Network (NN) and rule-based approaches. *Table* 2.1 shows the research finding reported by [35].

Table 2-1: Mining versus Rule-based and Neural Network Approach

% Cases with Greater Accuracy	Questionnaire Cases		New Cases	
Mining v Rules	Mining	Rules	Mining	Rules
	74%	9%	48%	19%
Mining v NN	Mining	NN	Mining	NN
	63%	19%	52%	19%

C) CONTEXT KNOWLEDGE DISCOVERY

To discuss the Context Knowledge Discovery (CKDD), let us compare it with the original KDD (Knowledge Discovery) process. Table 2.2, discusses the difference between CKDD and KDD.

CKDD is the core function of learning and reasoning module in CAMUS [55], which includes the steps pointed out below as reported on [51]:

i) Context data processing, which includes ontology, mining …. ;

ii) User identification (using RFID badge, PDA …) and context recognition (using neural network, Bayesian network…);

iii) Context mining, which mines association rules, classification rule … to provide input to learning step; and

iv) Learning, by the rule algorithm.

Table 2-2: Difference of CKDD and KDD

CKDD	-CKDD deals with context data in context aware systems.
	-CKDD tries to model the user and their behavior, as well tries to "understand" the needs of user.
KDD	-KDD Work with transactional data in business and commercial systems.
	-KDD normally discovers the interesting patterns of customers and sales.

D) SMAP-mine

The trend of anytime and anywhere services in pervasive computing environment is changed in the case of mobile services (m-service) for web environment as indicated in [59]. M-service provide personalized (preference-aware) and history aware computing in web environment; historical behavior of user is also another characteristics that enables prediction of appropriate services to the user desire. Additionally, m-service provide sequential mobile access pattern (SMAP). SMAP-mine is nothing but result of coining the concept of data mining with SMAP.

The system architecture for iMoWes (intelligent mobile web service) explained in [59], and shown in figure 2.1, provide the mobile user location dependent context-aware and personalized m-services by the interaction among distributed agent. iMoWes system divided in to four functional parts: i) m-servic registration mechanism, ii) mining mechanism for discovering sequential mobile access patterns, iii) personalization mechanism, and iv) context-aware m-service furnishing mechanism.

As demonstrated on figure 2.1; data mining mechanism helps to discover the SMAP which form the basis of personalized and history aware recommendations. The discovered patterns will be retrieved by PA (Personalization Agent). The mobility log, log integrator and sequential mobile access pattern mining components address the mining mechanism.

Mobility Log and Log Integrator

During runtime, the MPA (M-service Portal Agent) acquires the current location of the user from LAA (Location Acquisition Agent). Each service requests together with location information this repository in the format of (UserID, Location, ServiceID, time).

Sequential Mobile Access Pattern Mining

It can discover patterns of sequential movement associated with requested services for mobile users in mobile web systems. SMAP-mine deals with location and service dimensions; which are a two-dimensional datasets. In order to providing personalized and history-aware recommendation, the discovered pattern will be stored in a database. The proposed SMAP-mine method in [17], show a construction of SMAP-Tree and detailed algorithm for SMAP-mine. The SMAP-Tree compact a large patterns into the memory. Two principal benefits of SMAP-Tree are: a) only physical database scan to mine all of the large patterns, and b) data handling can be efficient as the data are

compacted. On the other hand, the algorithm of SMAP-mine is based on a well known depth-first search (DFS) approach. The algorithm recursively constructs the SMAP-Tree and mines the trees till termination condition is met.

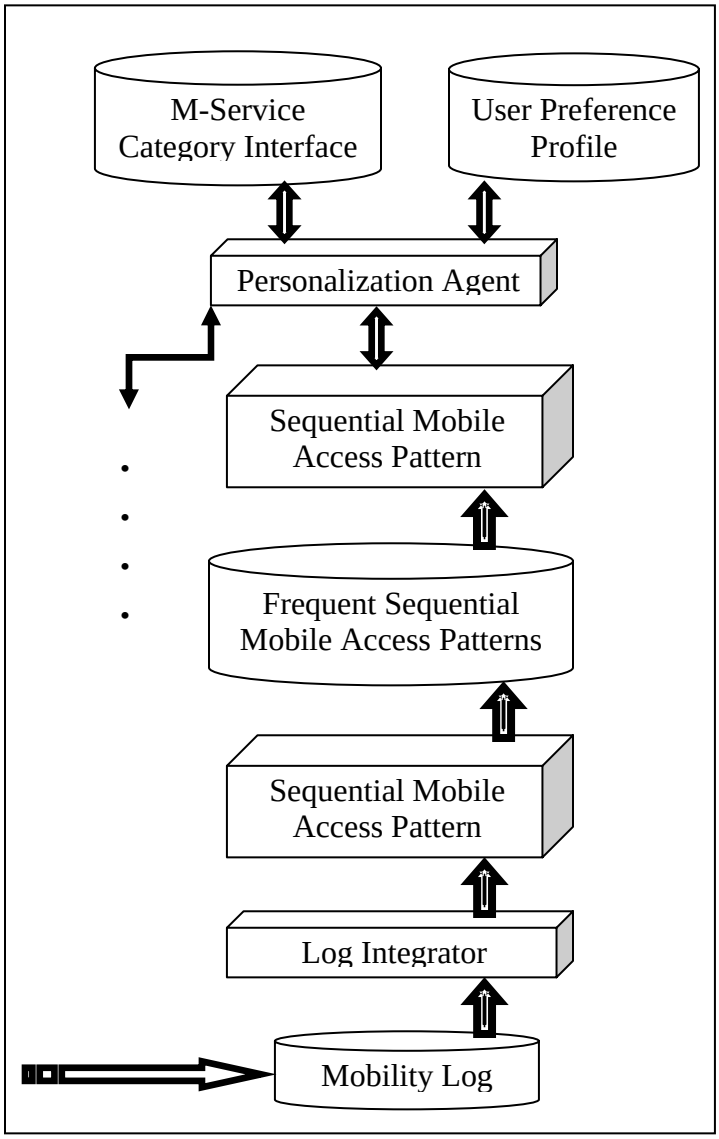


Figure 2-1: Partial architectural views of iMoWes

2.3 Context Middleware

In pervasive computing a user environment is dynamic, adaptive and interactive. Perceptual information about the location and status of the people, place and other devices in certain environment is provided as context. The environment becomes context-aware when it can capture, interpret, and reason about this information [34].

Precisely in a dynamic system, configuration change and resource variability is highly exhibited. Applications running in such environment adapt their behavior in accordance to the rapidly changing operating conditions. A model discussed in [32] indicates four unique features of context middleware: i) Abstraction specification of global context; ii) Redefine the abstraction dynamically; iii) Continuous maintenance of desired contextual information and iv) Flexible support for range of context-aware interactions.

In this section the well known context middle ware called Gaia and a recent context middle ware proposed by Ejigu D *et, al*. [22] are discussed.

A) HYBRID CONTEXT MANAGEMENT

How context data is collected, organized, represented, stored and presented is the main challenge in pervasive computing. A context representation model introduced in [22] consists of hierarchies (H), of set of entities (E), set of relations (Re), set of attributes (Ra), set of axioms (A), and set of metadata (M), called EHRAM were used and explained by using case data from medical domain. The EHRAM encompasses two context representations: the context knowledge and the context data which are vital for reasoning and decision made in a context-aware computing environment.

To manage the context knowledge (semantic of a data) and the context data a Hybrid Context Management (HCoM) model was also proposed on the same paper [22]. HCoM combines the best features of ontology tools (represent statically the knowledge domain) and standard database (represent the context data) in one hand and also add some enhanced features (such as query optimization…). In addition HCoM include a heuristic component, which loads only relevant static context data in to the reasoner depending on the location, activity and device surrounding the user. It uses a pattern matching technique on accumulated information of the user past history.

B) GAIA

Gaia [27] is a middleware-based meta-operating system; it enables communication and coordination between different devices and programming languages. Gaia application consists of a model, presentations, controllers, adapters and a coordinator distributed across devices in pervasive computing environment.

- The **coordinator manages** the composition of the application and provides a means to dynamically change the applications configuration by adding or removing components as needed within the environment.
- The **model** contains the application logic and maintains the state of the application. That interfaces the remote procedure call.
- The **presentation** component service as output mechanism for applications; it may change based on events it receive from the model.
- The **controller** components allow user to interact with application logic of the model by invoking methods on the model that change the applications state.
- The **adapter** components, assigned for each controller that maps methods from the controller to the method of the model. That ensures reusability of the model.

Applications and services within a pervasive environment in inter-application coordination are achieved using a specialized component called bridges. The task of a bridge is to receive events from source application, and invokes a method on the target application or services. And hence the state of the target application is affected by changes in source application state.

2.4 Event Prediction

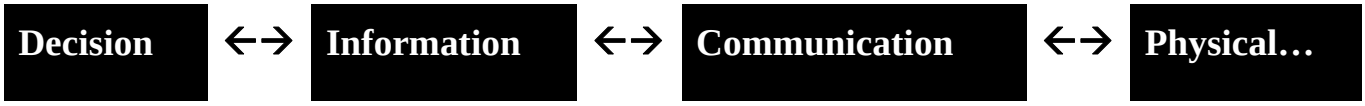
The most important design element in pervasive computing is how application obtains information they require in order to implement adaptive behavior [28]. Predicting events is one of the challenging phenomenon in pervasive environment. There are a number of research works that relates with event prediction based on context histories and current action of a user, devices and other resources in the user environment. MavHome, composite Service architecture, and CIS

consider how event prediction in pervasive computing environment is possible and will be explained respectively under this section.

A) MAVHOME

MavHome project [17], focused on perceiving the state of home through smart sensors and acting upon the environment. In short the house must be able to detect, predict, reason about, and adapt to its inhabitants. The MavHome intelligent agent predicts its inhabitant next action by automate repetitive tasks of the inhabitant. Past inhabitant activities can be used to determine how to control device through home. The SHIP algorithm matches the most recent sequence of events with sequence in collected histories. Another algorithm named as, Active LeZi (ALZ) were also used to process historical action sequence. To perform prediction ALZ calculates the probability of each action occurring in the passed sequence, and predicts the action with the highest probability.

The architecture of MavHome comprised of four layers.



- *Layer-4:* The **decision layer** selects action for agent based on the information provided by the information layer.
- *Layer-3:* The **information layer** is responsible to gather, store and generate knowledge, which is important for decision making.
- *Layer-2:* The **communication layer** is supposed to facilitate request and queries information among agents.
- *Layer-1:* The **physical layer** contains a number of hardware such as devices, transducers and some other network hardwires.

MavHome uses Task-based Markov Model (TMM) algorithm to identify high-level task action sequence. Given the current state of agent, it fir a simple Markov model, and it could be possible to predict the next action from collected action sequence. The sequence of events, or action, is first partitioned into individual tasks. A change in task is identified by gap in activities and changes in location of the action being performed. Additionally, an algorithm named as clustering was used to cluster the partitioned task sequences into sets of similar tasks, and task-meta features were

supplied to the cluster. The task-meta features include length in time and length in number of actions of the task sequence. In short, a collection of task sets, each of which can be labeled as separate task type is the output of the clustering algorithm.

In summary, MaveHome, allows integration of research in machine learning, database, mobile computing, robotic and multimedia computing for the provisioning of context predication in smart home. MaveHome, uses several prediction algorithms including SHIP (uses sequence matching), Active Lezi (enhanced LZ78 text compression algorithm to improve prediction), TMM algorithm (to predicting actions and ED algorithm (to mining data to be discussed under Section 2.2).

B) COMPOSITE SERVICE

Users influence the decision making process at large in pervasive environment. Decision making processes are triggered by event regarding context, service, and device availability. If context changes, ether a context condition of a composite service evaluates to true or it evaluates to false. Composition is totally possible in the first case, otherwise it is decomposition.

A pervasive service model and architecture given in [26], clearly describe that, services may be composed if they are interoperable. Interoperability comprised of syntax, semantics, protocol and context. The architecture of composite services in figure 2.2 consists of the service management and the environment management. The service management contains: central management, composite service management and state management.

- The **central management** is responsible for coordination of all composite services (based on user request or context changes).
- The **Composite service Management** is allocated for each composite services.
- The **state management** component provides reactive and proactive state migration, during adaption of composite services.

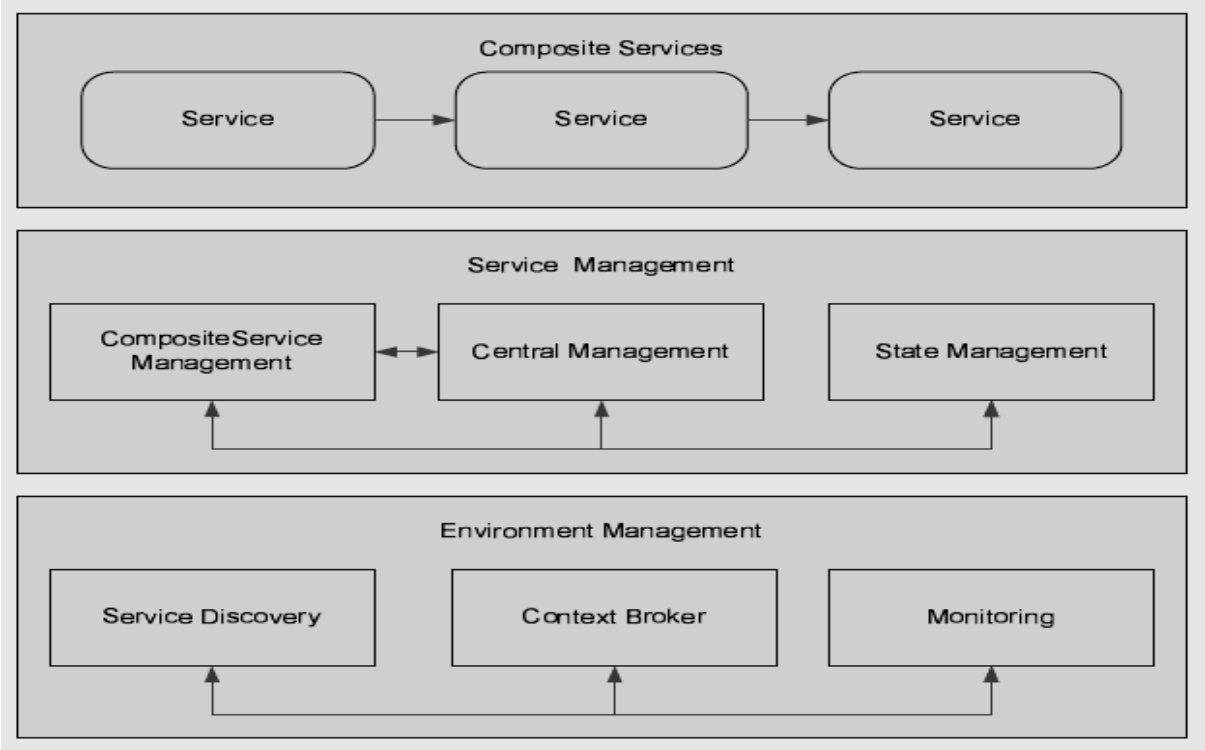


Figure 2-2: Service Composition Architecture

The environment management consists of context broker, service discovery and monitoring.

- The **context broker** provides relevant information by communicating with sensors or other similar context brokers.
- The **service discovery** can search for available services based on service attribute specified in query.
- The **monitoring** component update change of context and service availability.

The architecture introduced a two level decision making system, at the composite service and system level.

ii. **Composite service level**- a decision taken by composite service manger, in which decisions are made using linear programming.

iii. **System level**- a decision taken by central manager, using rule based control that facilitate three kinds of decisions.

   a. New composition service,
   b. Adaption of existing composite services, and

c. Transfer of composite service to another device.

In summary, on the first level best combination of available services is determined and on the second level the best composite services based on the overall running system is found.

C) CONTEXT INFORMATION SERVICES

Adoption can be reactive or proactive, i.e. anticipating the user's needs. The Context Information Services or CIS, introduced in the aura project [28], relies on a shared service. CIS was implemented using a virtual database, as contextual information often has meta-data associated with it (such as accuracy and prediction) and normal database approaches don't provide support for this.

The CIS [45] provide information about device, people, network, and physical spaces, and also the relationship between these entities. Figure 2.3 depicts the CIS provider class scheme.

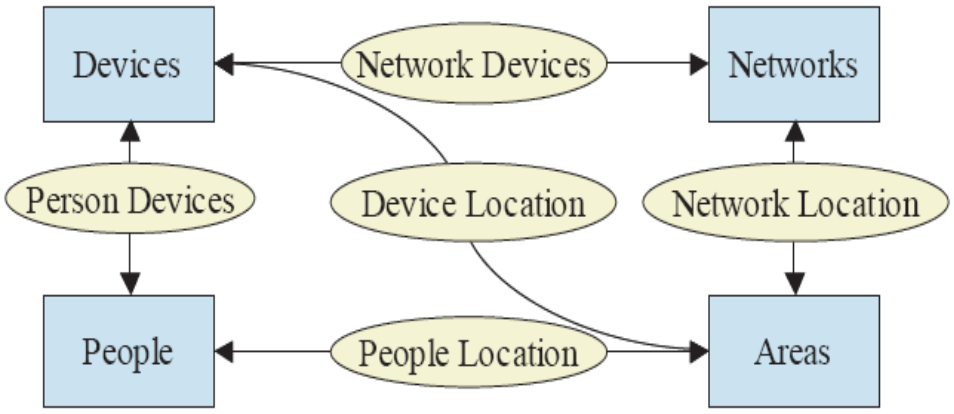


Figure 2-3: CIS Provider Class

As reported in [45], the adaptive and proactive applications of CIS supports a number of applications, more emphasis were given on proactive task assistant, which tries to anticipate the user's future needs. Predicting context in proactive application is much harder as compared reactive application. Proactive prediction is highly related to task-driven computing. In task driven computing, the computing environment keeps an explicit list of activities tasks, i.e. tasks that the user is currently working on or plan to work on.

Generally, CIS propose how XML/OWL (Semantic Web Language) sent over HTTP protocol for the ultimate of context information service provisioning.

2.5 Agent Base Modeling

In agent-based modeling (ABM), a system is modeled as a collection of autonomous decision-making entities called agents. Each agent individually assesses its situation and makes decisions on the basis of a set of rules. Agents may execute various behaviors appropriate for the system they represent—for example, producing, consuming, or selling. [8] Clearly suggests that, we live in an increasingly complex world, thus agent based modeling will replace the traditionally modeling tools and approaches.

An “agent” is defined in several ways by different scholars. Some consider agents as software module that can travel over a network. Other assume an agents as means that acts on be behalf of a human user or having minimal rank of intelligence that can determine and manipulate explicit models of Beliefs, Desire and Intensions (BID) as reported on [40]. Among the list of definitions provided the definition that imagines agents as a “proactive object” is the most appropriate and fit with the concepts that are introduced and discussed in this research work.

Charles M. and Michael J. [8] indicate and describe the major characteristics of agents as: i) An agent is identifiable (agents are self-contained); ii) An agent is situated (living in an environment with which it interact with other agents- by using protocols); iii) An agent may be goal-directed (undergoes outcome evaluation); and iv) An agent is flexible (having the ability to learn and adapt its behavior with the environment). Additional characteristics that an agents exhibit as a basic attributes includes: collaborative behavior (the ability to work with other agents), Inferential capability (the ability to act on abstract task specification), Mobility (the ability to migrate in a self-directed way), personality (ability to “believable” human character), and temporal continuity (persistence of identity and state over long periods of time) as realized from [3]. A pictorial description given in figure 2.4 from [8] also elaborate what an agent is more generalized way.

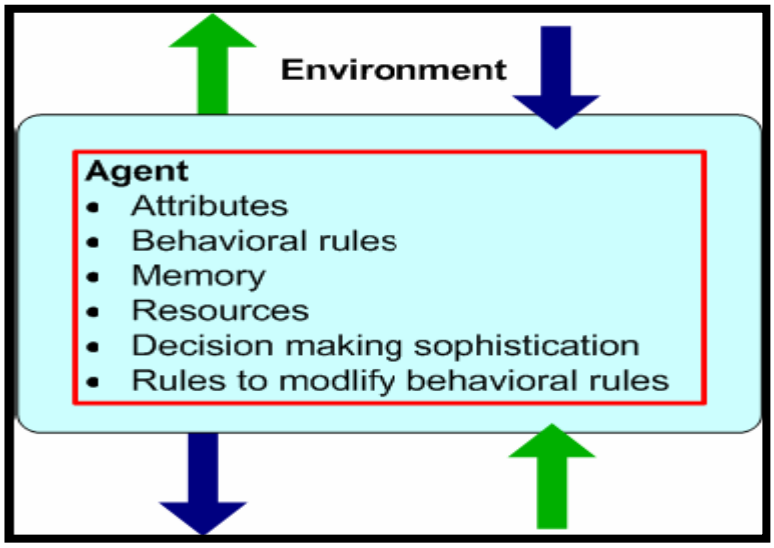


Figure 2-4: An Agent Description

As explained and elaborated in [25] agent infrastructure provides the regulations that agent follows to communicate and understand each other, thereby enabling knowledge sharing. Agent infrastructures deal with the following aspects:

- *Communication Protocols:* describe languages for agent communication.
- *Communication Infrastructures*: specify channels for agent communication.
- *Ontologies*: allow agents to agree about the meaning of concepts.
- *Interaction Protocols*: describe conventions for agent interactions.

According to [40], agent communication channels and agent communication protocols are grouped into four major classes. Below is a list of different agent communication channels:

i) *Medium*- Agents may communicate only through changes in their shred environment.
ii) *Addressing*- Broadcast to the entire community via direct address or subject-based addressing.
iii) *Persistence*- Message may persist for a period after they are sent.
iv) *Locality*- Changing communicative proximity in communication space.

On the other hand, conversations among agents are structured via a communication protocol. The following communication protocols were reported by the same article [40]:

i) *Directive*- gives order to one another among agents.
ii) *Voting*- Measure of aggregation (such as sum, average, etc) over the scalars of different agents.

iii) *Negotiation* - Exchange a serious of message before a decision is made.

iv) *Speech Acts* – building grammars out of which arbitrary protocol can be constructed.

The study in [25] also defines and describes Multi-Agent Systems (MAS) as loosely coupled problem-solver described by the list of characteristics given below:

- each agent has incomplete capabilities to solve a problem;
- there is no global system control;
- data is decentralized; and
- computation is asynchronous.

In summary the introduction of agent system changes the paradigm of software development to be component based as already discussed so far.

2.6 *Context Management Tools*

A concise review of weka, TopBraid Composer and Jena tools that are vital for context management and also used for the purpose of this study are considered in this section.

WEKA:

Currently, a number of pervasive computing researches support the preprocessing of raw context using a machine learning tools. Research in [17, 45, 53] uses a machine learning reasoning mechanisms, with the aid of WEKA toolkit. Using the APIs provided by WEKA as in the case of [53] the following interfaces are very vital: Classifier, Instance, Attributes and Filter (data pre-processing).The Weka's J48 implementation of C4.5 decision tree algorithm is a means to achieve the best classifier in context determination as clearly pointed out by [54].

TOPBRAID COMPOSER:

TopBraid Composer discussed and implemented in [33], explains the capability of this software as it is designed to be plug into Eclipse. TopBraid Composer allow automatic mapping of relational database schema into predefined OWL schemas in addition to the ontology representation feature. It also enables users to use different workspaces to import and use a combination of more than one ontology.

JENA:

As studied from [7], Jena Semitic Web Framework or Jena, is an open-source framework written in Java and provides a programmatic environment to develop semantic knowledge base using RDF, RDFS and OWL languages. The Jena framework also includes a rule-based inference engine that can be used to deduce new information from the predefined knowledge base (i.e., ontology representation).

2.7 Summary

This chapter covers the different contemporary art work in pervasive computing. As already discussed this chapter raised five core issues of pervasive computing research domain. ***Activity tracking*** is one of the aspects of research in pervasive computing which is highly related to the advancement of senor technology. As elucidated on [19, 20, 28, 39, 43, 60] activity tracking will be attained by means of RFID-tag and GSM signal fluctuations for tagged cell phones together with neural network approaches. Another research concern in pervasive computing and means for event prediction is ***context mining***. Context mining is characterized by knowledge discovery from existing context information and achieve by means of machine learning approach as studied from [17, 58, 59]. Moreover, this chapter also identifies how ***context middleware*** are working in pervasive computing for better context abstraction.

Event prediction is one of the challenging design goals of pervasive computing. The research finding considered in this chapter shows event prediction could possibly comprehend in pervasive computing by means of machine learning approaches as in the case of MAVHome [17] or via the usage of XML/OWL as indicated by Context Information Service (SIS) [45]. The general objective of ***agent based modeling*** are also examined as this study introduce a new way of agent based approach in pervasive computing domain. Section 2.5 indicates how agent infrastructures are possible by using communication protocol, communication infrastructure, ontology, and interaction protocol.

Finally, this chapter concluded by discussing three **Context management tools** that are vital for developing pervasive application.

3 Related Work

This chapter discussed a number of issues in relation to diary writing process and agent based approaches. The first section will explain existing diary writing mechanisms and its associated impact to human beings. Agent based approaches is considered in relation to our study under Section 3.2. The use of agent based approach is to facilitate intelligence for diary writing process as elaborated under chapter 4 of this paper. The last section of this chapter (Section 3.3) offers a summary of related work by providing a gap analysis which is already addressed and covered by this study.

3.1 Diary Writing

This section is organized in two sub sections in relation to diary writing. The first sub-section elaborates the importance of expressive diary writing for human cognitive development. The second part describes the common diary writing approaches.

Before proceeding to the stated issues, it is preferable to define and describe what a diary is. There are a number of definitions regarding diary. According to [75], a diary, or a journal is defined as a permanent personal record that is kept of events, thoughts, and ideas associated with an individual.

Another definition given in [78] defines and describes diary in terms of a class room environment where students, instructors, etc are interacting. Accordingly A diary is typically a notebook, booklet of blank pages, or any source for students to record thoughts, reactions to learning experiences, and even innermost fears about a learning activity.

Regardless of the particular format of a diary (Paper based, electronic, audio, etc) entries of daily shows experiences, insights, and problems often are made: Diary writing usually involves the unstructured, chronological recording of the events of a person's life" as they are perceived. Day-to-day experiences, social commentary, complaints, prose, illicit thoughts and anything can be written on diaries.

" however, that the mere fact of continuously writing entries, as is done in the keeping of a diary, is not sufficient in itself to bring about deep changes in a person's life" [52].

In the context of this research, a diary is defined as an activity based record of individuals in their day-to-day interactions with resources (such as computer, projectors, and other machineries), peoples, and some other entities in the user environment in a given location and time. Additional emotional expressions as a result of a particular interaction are the most vital and it could be annotated whenever required.

3.1.1 The Impact of Expressive Writing in Human Cognitive

A research work by Kitten Klein [36] indicates that expressive writing reduces interfering and avoidant thoughts about negative events and improves working memory. These improvements, researchers believe, may in turn free up our cognitive resources for other mental activities, including our ability to cope more effectively with stress.

Klein also states that:

> *"for fairly minor life problems, something as simple as writing about the problem for 20 minutes can yield important effects not only in terms of physical health and mental health, but also in terms of cognitive abilities,"*

Writing about stressful events has long been known to cause improvements in health and psychological well-being. A recent study by Pennebaker and his research collogues [11, 57], indicates the role of journal writing in helping people cope with job loss at mid-life. Persons who wrote out their feelings about job loss for a period of 20 minutes a day for five days within a week were more likely to be working again when followed after a couple of months, in comparison to those who did not do the writing exercise. These findings suggest that putting feelings down on paper may help to get rid of feelings of anger or other negative emotions and allow a person to move forwards towards constructive action.

In general, open up persons feeling is not a simple task, journaling or diary writing is one of the simplest method to use for the identification of the insight of an individual. A number of physicians and therapists have used this technique to come across with the feeling of an individual as indicated in [36].

Precisely speaking our research work has a valuable input and also indicates a new direction of using computer based technologies to support and facilitate diary writing process. The users are

expected to annotate some deeper emotional feelings of their own using the intelligent system that our study possibly make available.

We strongly argue that it is the memory that comes first before the emotional expression narration. Thus our system provides to build user to have a review of memory about their daily activities followed by allowing annotation of their emotional feeling in their convenient time.

3.1.2 Common Diary Writing Approaches

This sub-section identified and summarized the existing diary writing approaches that the society uses up to date. The manual diary and digital diary (the computer diary and online diary) facilities are some of the common methods used in the society. The three commonly used diary writing approaches the manual diary, the computer based diary and the online diary are discussed and explained in detailed below:

I. MANUAL DIARY:

A Manual Diary (MD) is a paper entry and the most common form of personal daily profile. A person who wishes to take events, emotions … of his/her daily experience suppose to purchase specific notebooks or blank books just for keeping own journal entries. It requires paper and pens just for writing the diary entries.

II. COMPUTER BASE DIARY:

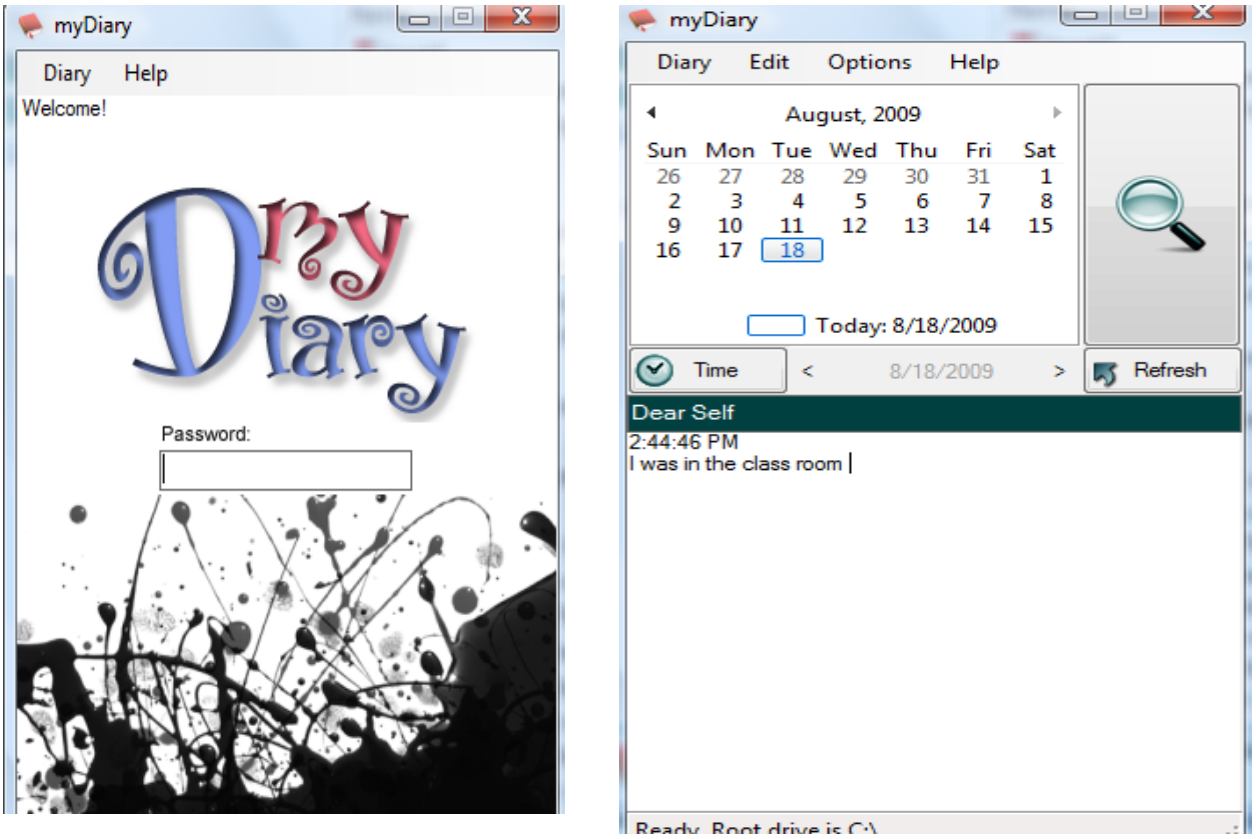
Computer Based Diary (CBD): becoming more common as technology is integrated more and more into our daily life. One can write his/her experiences on using personal computer through a word processing program or, simple computer applications. Computer base diary is reliable for storing everyday blogs or journals in digital format. One of such applications that can be downloaded and used is myDiary version 4.5[2]. This program is designed and developed by William w. Mensah.

myDiary, is password protected, and blogs are encrypted using a simple yet powerful encryption algorithm to ensure that your journals are secured. myDiary also automatically saves as all the information as you type so you don´t have to worry about saving. Once entering password, the user

[2] A software which is freely available in the web address www.wilmens.net

can pick date/time information, and start building the diary. Stored diary or personal journal can be searched and retrieved if required.

When the application (myDiary 4.5 beta version) runs it provides the interfaces indicated in figure 3.1.



a) **b)**

Figure 3-1: myDiary user interfaces a) login interface b) diary composer interface

Other desktop diary builder applications are also available and can be downloaded via the following web addresses [63, 69, 73].

III. ONLINE DIARY (OD):

With the increased usage of Internet in our daily life and with improved bandwidth and memory online diary comes to the life style of today's society. There are a number of online diary service providers including [70, 71, 76, 77]. Online diaries were introduced in 1994 by Media Lab in the

University of Manchester [64]. Online diaries or journals are also known to be blogs or personal blogs. Figure 3.2 provides a typical diary composer screen shot of online diary.

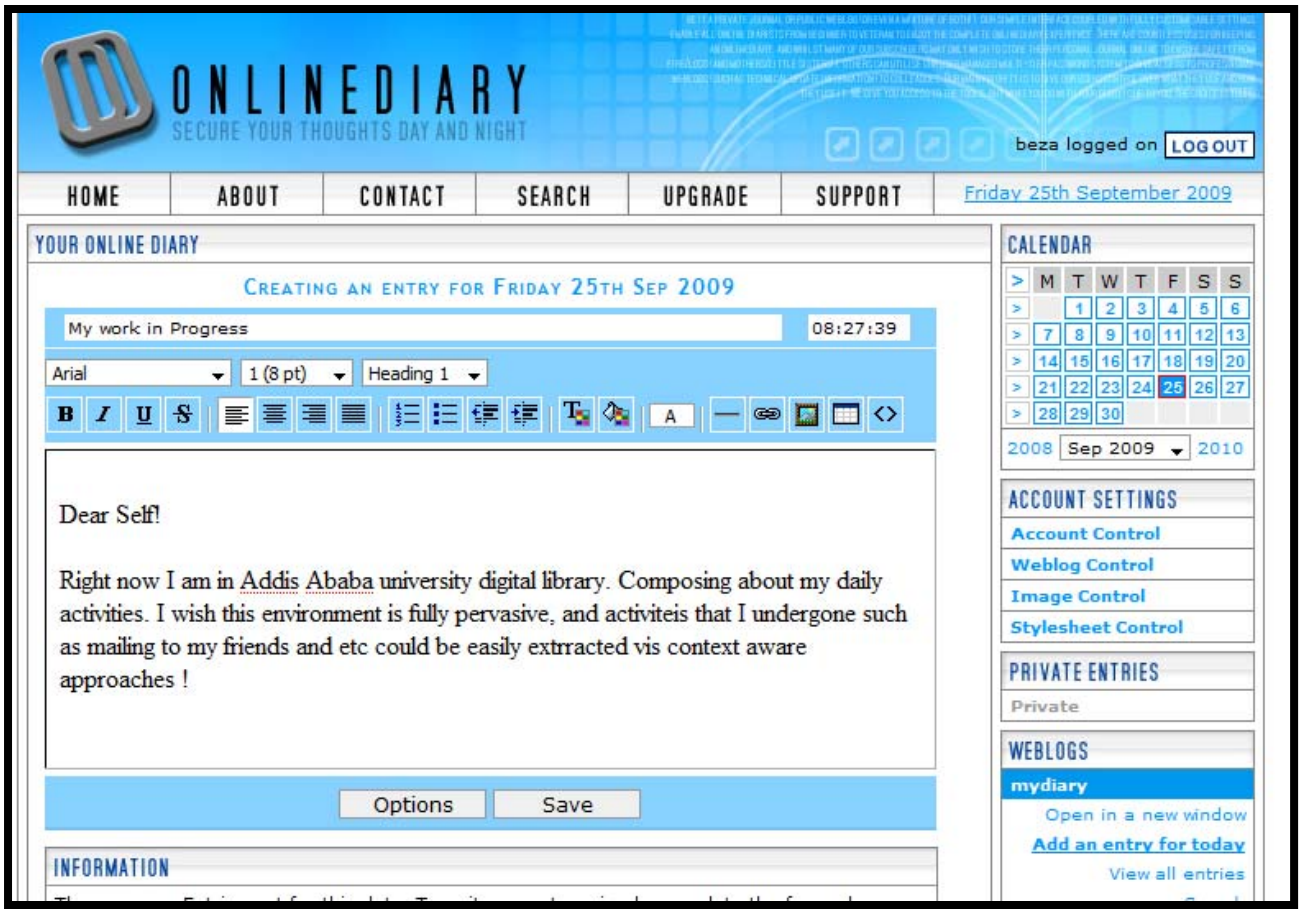


Figure 3-2: Online Diary composer[3]

3.2 Agent Based Context Aware Services

This study uses software agents as a means of information advertising and publication while generating an activity and location related data in producing a context aware diary. This research tries to assess efforts regarding agent based approach in the area of pervasive computing.

In this section four different pervasive environment agent based approaches will be described, discussed and summarized by indicating their corresponding advantages and limitations. The ACAI (Agent-based Context-Aware Infrastructure, CLA (Context Level Aggregation), Intelligent

[3] Online diary interface found on http://www.olinediary.net

Artefact's Agent based approach and Mobile Agent approaches will be discussed under the subsequent sub section.

3.2.1 Agent-based Context-Aware Infrastructure

Agent-based Context-Aware Infrastructure (ACAI) [48] has a layered architecture (figure 3.3) that allows context information to be collected, processed, inferred, and disseminated to spontaneous applications.

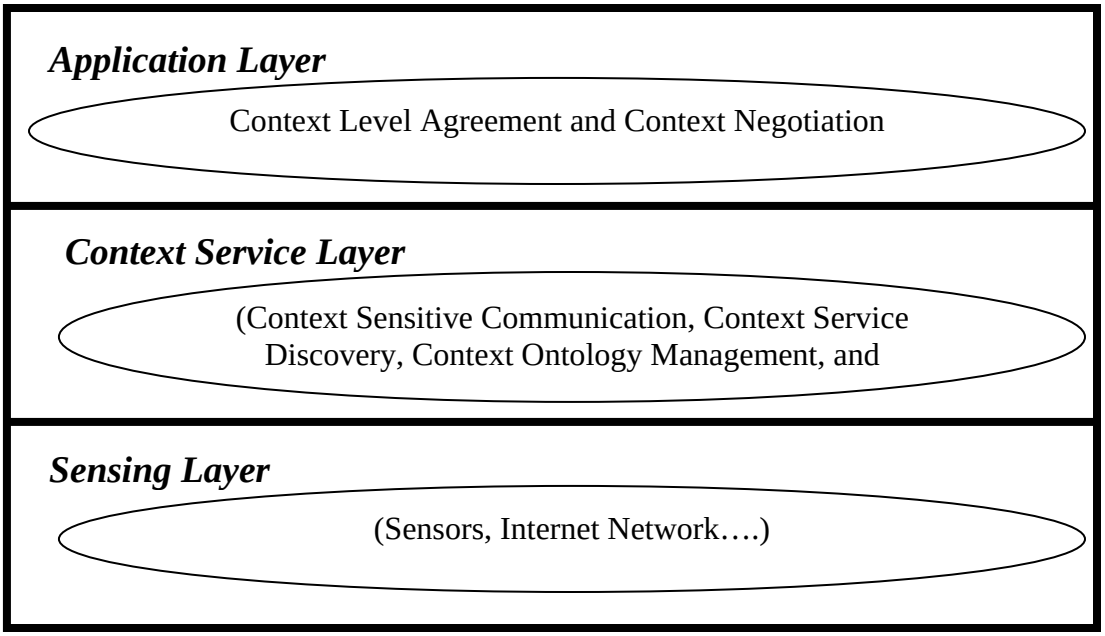


Figure 3-3: ACAI Layered Architecture

The 1st layer: named as the sensing layer, is responsible for capturing and acquiring context information; where various sources are embedded in the environment.

The 2nd layer: named as the context service layer, has two goals:

A) Structure context, unified, composable and extensible model; and

B) To provide context-based services.

A) Will be attained by context ontology and context inference module. On one hand, context is interpreted and structured by context ontology. On the other hand, context ontology module deduces other contextual information that has not been explicitly sensed by the first layer.

B) Will be achieved through the following modules:

Context Management: deals with how context can be stored, queried and accessed by users and resources with limited capabilities.

Context based service discovery: provide service and resources through Peer-to-Peer (P2P) discovery process. Each peer node holds information about services, users and even context of interest to others.

Context sensitive communication: devices, services and user interaction with spontaneous application seamlessly, and hence this module provide presence, notification and network connectivity with the same or with the different domain.

The 3rd layer: named as the application layer, interface between mobile user and application on one hand, and the ACAI context service function on the other hand. Two processes are implemented: a) CLA (Context Level Agreement) and b) Context negotiation of context specification

Detail of context level agreement (CLA) is discussed in Section 3.2.2. Negotiation at active context level aims to achieve common understanding of context's meaning and format. On the other hand, negotiation at the dynamic level aims to provide personalized context such as storage, delivery, remote access, and inference.

ACAI Agent Model:

The heart of ACAI's multi-agent system consists of: the Context Management Agent (CMA); the Coordinator Agent (CA); and the Ontology Agent (OA). The Reasoner Agent (RA) and the System Knowledge Base Agent (SKBA) are optional and require in certain situations based on specific application requirements. The scholars of [48] Khedr M., Karmouch A. also identify Context Provider Agent (CPA).

In summary, ACAI illustrates how adaption strategy in pervasive environment could be possible via the usage of a number of agents. Spontaneous interaction among a number of entities could be possible using multi-agent approach as illustrated by ACAI. The following points are identified as a limitation of ACAI:

- Implementing context level agreement at the application level;
- No effort to address the issue of context refinement;

- How context negotiation is measured? - Is not clearly indicated; and
- How agent response will be aggregated and summarized for the purpose of reducing user adaption effort is not clearly justified.

3.2.2 Context Level Agreement

The context-aware architecture given in [34] is an extension effort proposed by Khedr M., Karmouch A. [48], which gives high emphasis for context level agreement to be implemented at the application layer of ACAI. The following issues are raised and discussed by Context Level Agreement (CLA): user and service agents, context management agent, inference agent, ontology agent, and context provider agent perform the function required for context negotiation and provisioning. In short, for CLA to be dynamic, automated, and extensible, it must provide a framework for negotiating context specifications and an ontology that provides standard and extensible construct of context and negotiation per-formatives.

The following lists are identified as advantages of CLA according to [34]:

- It simplifies interpretation, management and reasoning aspects of context for the end-user;
- It protect context providers against unexpected degradation in performance;
- It provides seamless context-based service assurances; and
- It offers for context provider to intelligently provide and monitor context and resources.

Types of CLA:

1. Passive context: context requirements and operations from the context provider are simple.
2. Active context: user are more interactive; complex operation on the part of the providers, such as filtering, merging, composition and monitoring.

3. Dynamic (spontaneous) context: where context management and inference are essential to provide higher context levels.

Context-aware systems require the following components:

Context Ontology and Inference:- to facilitate the sharing, management, and inference of a given context;

Context-level Negotiation: - to agree on levels of context that can be provided and enable personal context provisioning; and

Context Management: - to store, retrieve, query, and modify context information.

We have the same critique regarding this paper as that of the ACAI, the CLA lacks the context level agreement and negotiation implementation at lower level. Context agreement and negotiation on raw context information is really improving the performance of the context prediction process in general.

3.2.3 Intelligent Artifact's Agent based approach

The major aim of Intelligent Artifact's [46] is to address service discovery functionality of computing devices operates in pervasive computing environments. Change of location and orientation are frequent phenomenon in mobile and wireless computing environment. Sensing and interpreting the event that occur in user environment is the task of context-aware application. On the other hand, mobility of service users and service devices potentially brought a number of problems for context-aware application.

According to [46], pervasive computing elements fall in to three major categories: Intelligent Artifacts, Sensors and Actuators.

An Intelligent Artifacts is a mobile computing device able to offer high complexity service to make decisions derived from computational process and analysis of less composite services. The internal architecture of such entity is based on the technology of intelligent agents.

On the other hand, sensors provide data service (i.e. raw data, simple signals or features), while Actuators provide action service. Both Sensors and Actuators cannot advertise data by themselves instead they must be able to poll by an Intelligent Artifacts.

Minas P. and Nicolas T. proposed artifact's agent-based architecture [46]. As depicted in figure 3.4 the artifact's agent-based approach consisted of two basic components: the Agent Platform and the Communication Manager.

The Agent Platform includes the Service Manager(s), in which the controllers of the system that coordinate the message are passing protocol between Clients and Services, and Fig.3.4: Artifact's Architecture the Intelligent Unit(s), which perform intelligent tasks using the Primitive Services

provided, acting also as providers of Composite Services to other Clients (e.g. other Intelligent Units). The Communication Manager(s) handle all the lower level Communication. The Communication Manager (CM) is responsible for the communication between the Sensors or Actuators and the Service Manager. It can implement a number of different protocols by having different communication modules, for example, one that handles IR, another that works with Bluetooth, one that works with wireless LAN etc.

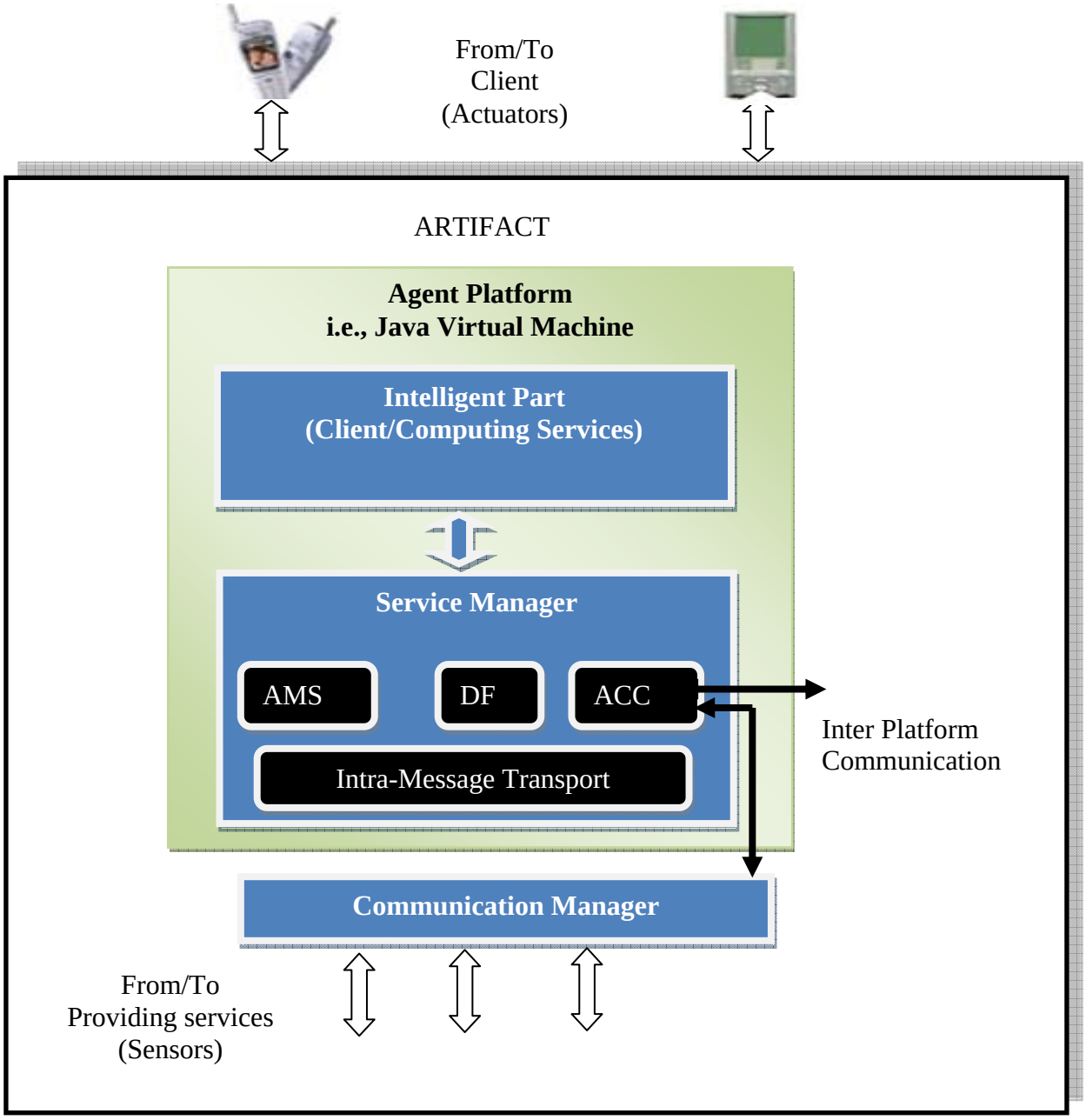


Figure 3-4: Artifacts architecture

In summary, the intelligent artifacts introduced in this research illustrate how service discovery is possible in pervasive computing Mobile Ad-hoc Network **(**MANET). The limitations that are identified in this paper are the following:

- No effort on how to build intelligence on Intelligent Units;
- No context refinement strategy; and
- Lack of usage of any reasoning on how to learn user environment.

3.2.4 Mobile Agent

Mobile agents [12] play an important role in inherent flexibility, and can bring a number of benefits to pervasive computing systems. The mobile agent research proposes an architecture that can use mobile agents to perform the common task of pervasive computing such as system adaption, component updates, and Quality of Service (QoS) negotiation.

The need for mobile agent is important in pervasive computing as the computing environments are always changing. Different devices and users become part of the system and leave them frequently. However network communications in pervasive environments are characterized by high error rate.

The approach proposed in this research more specifically focus on how to perform administrative tasks such as installing software and updating existing components with little or no human intervention. Shortly, mobile agents can solve issue related to heterogeneity of devices and environments. Moreover, mobile agents provide a number of advantages including performance, flexibility, and scalability as realized from [12].

Briefly, the three subsystems mentioned and explained by mobile agents are discussed in the paragraphs given here under.

The first subsystem is responsible for service adaption. Detecting changes on the environment and decides where to migrate the services is core issue of this subsystem.

The second subsystem controls the component updates through the environment. Updates of new component, distribution, or installation are the major issue to be handled by this subsystem. Component updates follow a push approach to install or update components. In case were services migrate to a new host a pull approach is used for component update.

The last subsystem deal with Quality of Service (QoS) negotiation; this subsystem addresses the hardware and software requirement of multiple QoS level among several application of pervasive computing environment.

The major limitation of mobile agent approach is summarized as follows:

- More emphasis was given for external agent communication and omits the internal integration and communication within a device;
- Lack of explanation where the different subsystem lies in pervasive environment;
- Unclear description regarding the QoS attainment; and
- Does not consider most of the common issue in mobile computing such as transparency, replication, relocation, etc.

3.3 Summary

This section provides the justification of our research problem as a summary of related work analysis. The existing diary approaches are summarized and provides a gap analysis to be covered by this research work. Why agent system and what are the observable limitations to the existing agent system approach in relation to pervasive computing with a mitigation strategy proposed by this study will be pointed out as well.

Table 3.1 summarizes existing diary writing mechanisms, including: Manual Diary (MD), Computer Based Diary (CBD) and Online Diary (OD). We have used our own parameter to evaluate the existing approach and to analyze the gap that our system expected to address.

According to the evaluation against the parameters that we set in Table 3.1 MD fails to fulfill any of the criteria and make use of physical security mechanism. The CBD facilitates and provides the searching/sorting of user digital diary and allow integration of multi-media partially (i.e., only to pictures, audio, etc available in the computer systems, if any). The CBD omits sharing, anywhere, anytime and distraction free parameters.

Table 3-1: Comparison of Existing Diary Writing Approaches

Approaches	Parameters							
	Any-where	Any-time	Sharing	Searching/ sorting	Security mechanism	Privacy	Multimedia Integrations	Distraction free
MD	-	-	-	-	Physical Security	?	-	-
CBD	-	-	-	++	Use computer security	?	+	-
OD	+	+	++	++	Use web security	?	++	-
(Key: ++ = Comprehensive + = Partial ? = Unknown - = None)								

On the other hand, OD addresses most of the limitations observed by MD and CABD. The major limitation of online diary is it is still characterized by high involvement of user while composing the diary (i.e., Lack of distraction free approach) and it is not also fully "anytime-anywhere". Precisely speaking, the major drawback of OD is associated with being context aware and lack of pervasiveness. Our proposed diary writing mechanism to be introduced and discussed in chapter 4 will address the limitations experienced by online diary writing mechanism by using the concept of pervasive computing approaches.

Consequently, the major problem to develop context-aware diary builder is related to the core concepts of pervasive computing, the potential research problems indicated in chapter 1 of this documentation will be considered and addressed by this study to make the diary writing approach fully context aware. We strongly argue that diary writing is more than keeping day-to-day activities of an individual rather it is also a means to evaluate past experience and mold future direction of an individual. A number of researches enclosed in this study including [10, 19, 20, 28, 39] indicates recorded diary are a means of monitoring user activity and improves the daily life of individuals.

Another problem justified in this thesis work is related to the usage of agent based approach in pervasive computing. Agent based approaches introduce new challenge and opportunity for context aware applications as pointed out in Section 3.2. None of the agent system covered in our discussion makes use of context level agreement/ negotiation at lower level and also lack to indicate how agent response is handled and evaluated. All research works explored to the best of

our knowledge try to make use of agent technology agreement/negotiation at the application layer. Context agreement/negotiation at lower level improves the performance of context reasoning. This study explore a way of achieving context level agreement/negotiation at lower level, specifically at the second layer of context aware diary builder architecture (CADB) which is discussed in the next chapter. Additionally, how agent response is handled and evaluated is also discussed and elaborated in depth in this research work.

4 Architecture of Context Aware Diary Builder

This chapter consists of five sections. The first Section provides an overview of CADB architecture. Description regarding the proposed architecture is the issue of Section 4.2. Section 4.3 discusses the components of core services in CADB architecture. Section 4.4 provides detail of multi agent system of CADB architecture. The final section concludes the chapter by providing a brief summary of CADB architecture.

4.1 Overview of the CADB Architecture

The proposed architecture for building context aware diary builder is a layered architecture that also contains a multi-agent platform, which shows how the diary write-up process takes place starting from context acquisition up to the application layer.

Agent based approach provide a number of advantage in the pervasive environment. Agent technologies seem to offer a suitable paradigm for context prediction. The major role of agents in pervasive computing is to monitor an entity. Information identified regarding any entity could be communicated by publishing, and advertising the information whenever any changing circumstances occurred [49]. In the case of our study, six agents are introduced that communicate and exchange raw context data by means of publication and advertising as the context of the user and its environment change dynamically.

In addition to this coordinating agents and vote for context information to support context prediction and confidence is another fundamental advantage that can be facilitated from agent based approach. Managing and coordinating agents to support better context prediction and simplifying the diary building process is the core hypothesis that this paper dealt with. This study does not address the issue of confidence among agents in context-awareness.

This research work uses one major multi-agent component named as Agent Manager that coordinate, and integrate several agents. The importance of the agents used throughout the proposed architecture is to facilitate raw context agreement/negotiation. The multi-agent approach that proposed in this study is discussed under Section 4.4.

On the other hand, a layered architecture provides tremendous benefits for the development of context-aware applications. A number of research including [47, 61, and 62] uses a layered approach for developing a context-aware application. As indicated in [67] some of the advantages that can be facilitated from layered approach include; increased life expectancy, mobility, value added features, modularity, innate plasticity, interoperability, greater compatibility, and better flexibility.

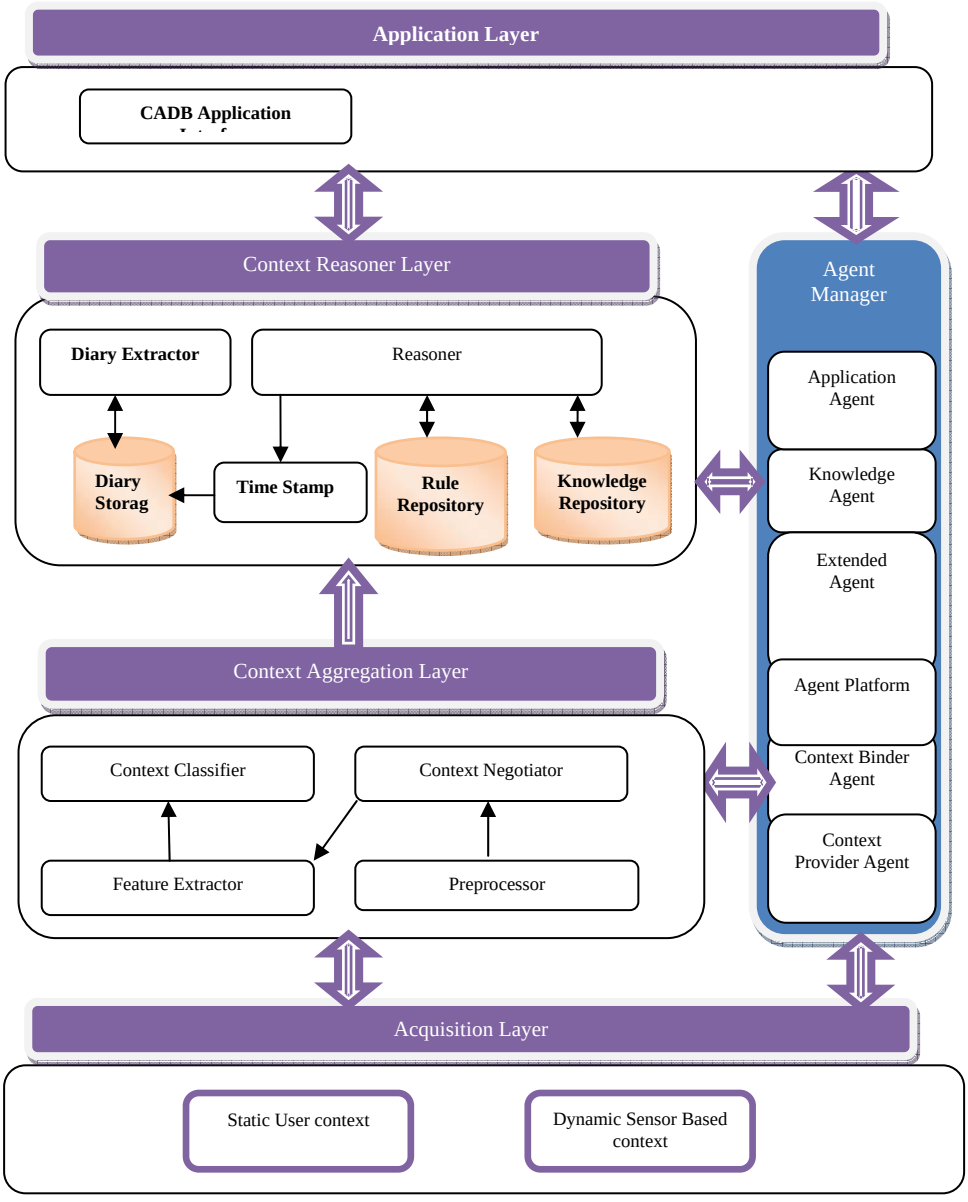


Figure 4-1: CADB Proposed architecture

A four layered architecture of Context-Aware Diary Builder (CADB) together with an agent manager component that has at least one agent representative at every layer is given in figure 4.1. The proposed CADB architecture categorized in to two broad architectural classifications. Those components residing under the Agent Manger are considered as CADB Multi-Agent System (CADB-MAS); and those components that are not residing under the Agent Manger are regarded as CADB Core Services (CADB-CS). Each of these two major architectural classes of CADB components are considered in Section 4.3 and 4.4.

4.2 Description of the Proposed Architecture

This section describes and provides an overview of the four layered architecture of CADB, depicted in figure 4.1. As can be seen from the diagram the proposed CADB-architecture consists of Acquisition, Context Aggregator, Context Reasoner and Application Layer; together with multi-agent manager components. The four layers are concisely described in the paragraphs given below.

Layer 1: ACQUISITION LAYER

The first layer of CADB, this layer is the primary means of capturing lower level context data. The acquisition layer contains two major components that are vital for triggering the process of diary building. According to the proposed architecture the sources of context data can be from heterogeneous sensors scattered in pervasive environment (which is a dynamic context data).

Context data could also be generated statically from user agenda, reminder, or to-do-list which is helpful for context aggregation, prediction, association, and reasoning. Our research work suggests a comprehensive approach that explains how the acquisition of context data can be implemented at lower level in systematic way. More detailed explanation about the static and dynamic data acquisition approaches deployed and used in CADB is given in Section 4.3.1.

Layer 2: CONTEXT AGGREGATION LAYER

The second layer of CADB, this layer facilitates and supports the reasoning process of diary writing. The purpose of this layer is to characterize the low level context data so as to extract high level context information. Characterization of low level context requires the use of machine learning approach, as in the case of our study. Raw context data that are captured from various

possible sources found in acquisition layer should pass through a number of processes to be meaningful and as well to provide high level context data to other higher level components found under the reasoner layer.

Four basic components were blended together and work incorporate fashion in this layer to come up with improved context aggregation. To elicit the high level context information about a user and its associated environment; characterization, feature extraction and classification are some of the mechanisms used. This layer establishes a means of getting an already maintained data to gain an increased aggregation on the context information being analyzed.

The different components that are located under this layer are the following: the Preprocessor, Context Negotiator, Feature Extractor and Context Classifier.

In addition to building a context aware diary, one of the core contribution of our research work lies on context negotiator component. The context negotiator component has a special capability of integrating the raw context data with existing high level user and environment information by deploying a multi-agent approach.

A closer look to the components found under this layer is discussed in Section 4.3.2.

Layer 3: CONTEXT REASONER LAYER

The third layer of CADB, this layer provides a more critical functionality of a context aware diary builder. Diary write-up process undertake at this specific layer after reasoning on the context being aggregated by the second layer. Three major components and three storage facilities enable the process of reasoning and diary write-up process by deploying appropriate algorithms, explained and indicated under Chapter 5.

On the classified context a Jena reasoning capability will be applied, so as to identify the exact event of the user, by using a suitable rule based inference engine. The rules are inferred from the rule repository. On the other hand, knowledge repository stores appropriate description of every entity in the user environment including the description of the user. After reasoning on the user activity, the event will have a time stamp before persistently stored in to diary storage. The stored diary of the user can be accessed by the application envisage to be developed via a diary extractor component. Detail explanation of the components and the storage facilities will be given in Section 4.3.3.

Layer 4: APPLICATION LAYER

The last (4th layer of CADB), the application layer provides user activity diary application interface that enables the user to navigate, edit, update, and cancel a concise summary of user activity. It also generates user activity progress in the form of report for a specific task. The report could be viewed as a daily activity of the user which can be viewed with respect to date information.

The application layer also takes part in adaption of user context, by providing appropriate interface to the user. Section 4.3.4 explains the application layer in depth.

This research work makes use of a multi-agent approach that can reside to the proposed layered architecture of CADB or to any pervasive environment application at large. Accordingly each layer has got at least one agent that can represent the entire layer. Detailed explanation about the multi agent approach followed by this study together with how it works will be discussed in Section 4.4.

4.3 Detailed View of Core Services in CADB Architecture

In this section detail view of the core services in CADB is explained and elaborated in depth. To realize the core service of CADB functionality examples, scenarios and steps are given whenever required.

4.3.1 Acquisition Layer Components

The acquisition layer comprise of two components namely, the dynamic sensor based context (which is the first to initiate any activity of diary writing process) and static user context (which stores the static information of user data and make use of it to support prediction, and reasoning). More detailed explanation about the two components will be presented in the subsections 4.3.1.1 and 4.3.1.2.

This study comprehensively provides a better view of how acquisition layer manage the raw context in a systematic way so as to simplify other tasks such as: context processing, aggregation, and reasoning. Figure 4.2 elicits the insight of the acquisition layer.

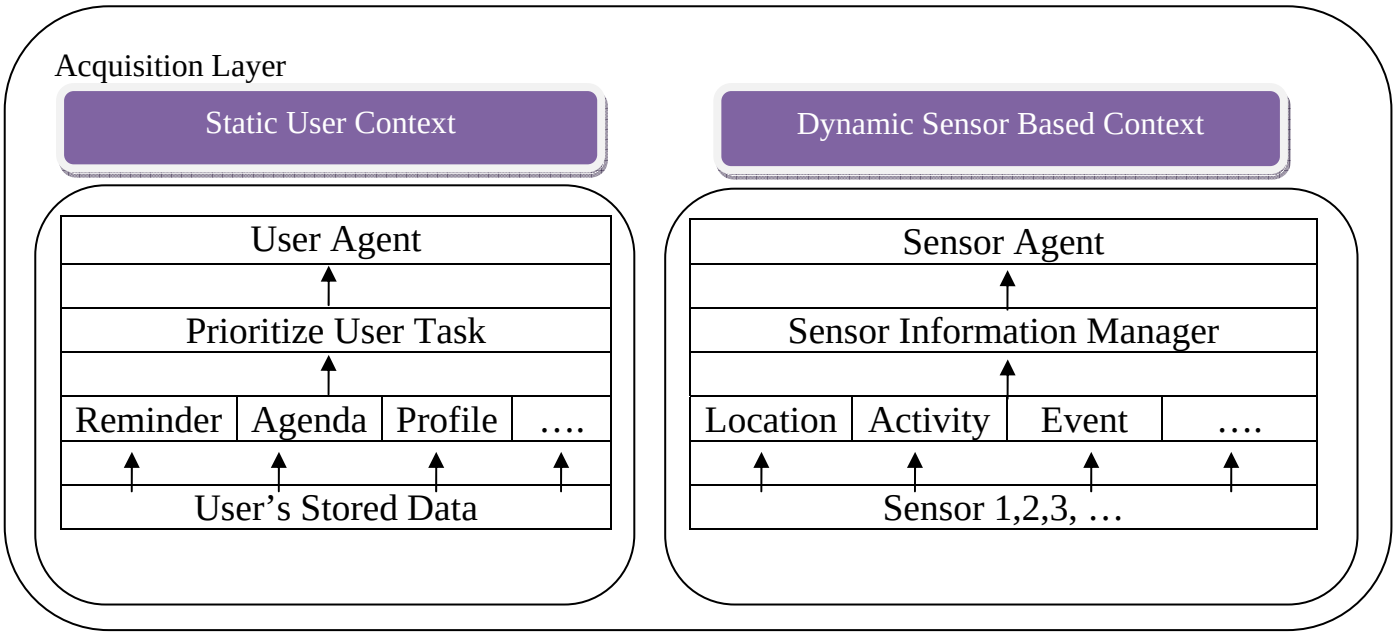


Figure 4-2: Acquisition Layer Components

4.3.1.1 Static User Context

The major source of static user context is the information stored in the user's hand held device. The user's stored data more specifically profile, well written reminder, agenda and to do list, as well as the previously maintained diaries are the core sources of static context.

Usually daily activity of an individual is highly influenced by habitual actions. In an environment such as a factory or some other industries the activities are more of routine than habitual. If a person is assigned in processing a certain product he/she is committed to accomplish the task of processing. On the other hand in environments like campuses or universities, the activities can be segmented in to yearly academic calendars, semester's course offerings, quarter's activities, and weekly and daily schedules of classes as well. Entities in campus environments such as Person (Students and Instructors), Locations (Class Rooms, Libraries, Dormitories, etc) and Activities within a campus (such as Learning, Advising, Seminar Presentation, Meeting, etc) could be considered as a common phenomenon. Once the mentioned activities are scheduled students, instructors, and other administrative staffs find themselves in accomplishing the scheduled task. The same scenario could be applied to several environments where activities are more of habitual actions.

Usage of static context actually not only supports adaptation of the application to users current situation but it also drastically decrease the data size of diary writing process at large. Association of context data to current event could also be another considerable benefit that could be brought from our research.

The static user context operates in offline mode by tracking the already stored profile inside the users' hand held devices or the data maintained by the user himself/herself. As depicted in figure 4.2, the static context information could be extracted from various applications included in the devices of the user (PDA/Smart Phone). After extracting the stored data via a common interface our approach make use of the algorithm defined in figure 5.4 to prioritize the tasks in accordance to the user environment and the pre scheduled user information. The task being prioritized for an existing environment will be published to other agents to increase the knowledge of the environment via the user agent. How to manage such agents is the concern of agent manager. The detail of agent manager is given in Section 4.4.1. Sensing the user environment is the task of the dynamic sensor based component.

4.3.1.2 Dynamic Sensor Based Context

The sources of dynamic sensor based context are the pervasive environment and the current activity of the user. As it is already mentioned, pervasive computing environments are characterized by having diversified sensor network. The communication among several resources (devices) is possible by sensing the environment. More specifically the services in the environment, the users within the environment, device capabilities (RAM, processor speed, etc), the location coordinates, and etc. are some of the vital information to be sensed and communicated among the different resources to support the provision of adaption and prediction.

In figure 4.2 given above it is clearly explained that the sensor data can be augmented in to location, activity, event, or to any other possible set of user circumstances information. Knowing the location of an entity can be easily facilitated by sensors as the case in many researches in pervasive computing [18 and 31]. Activity prediction is another sort of information that can be easily identified via sensors. To deduce the accomplishment of any activity the events are more vital sort of data that can be possibly extracted via sensors.

Shortly, sensors can provide a number of contextual data, if entities in smart space environment are tagged. The location of tagged entities, its movement, and its interaction with other entities could

be identified by aggregating diverse sensor information within the user environment. Thus sensors can potentially provide a number of information not limited to the location, activities, and events.

The possible sets of collected information via sensors (such as location, activity, event, etc) are managed by Sensor Information Manager Component. This component will provide a characterization of the sensed signal so as to produce well versed information of the raw context data. The sensor information manager also enables a mechanism of establishing trust and confidence on sensors crowded in the smart space environment which is not covered by this study.

Lastly, the senor information manager will pass whatever sort of raw context being characterized to an agent named as Sensor Agent. The sensor agent will publish and pass the information to the preprocessor component so as to start the process of context aggregation.

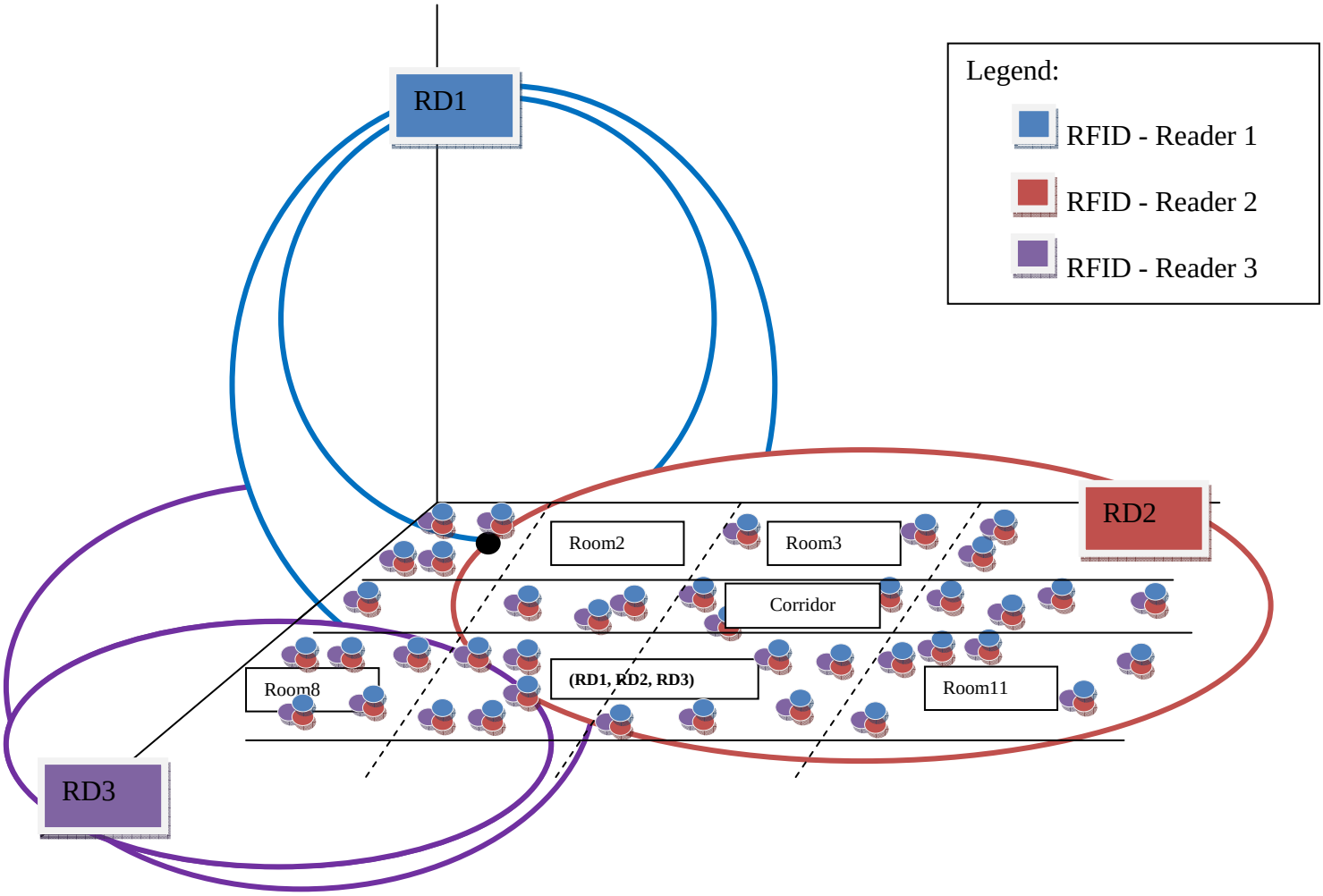


Figure 4-3: RFID-Reader and Atomic Location Points from Three Sensors

Location detection can be facilitated in a number of ways including: triangulation, trilateration, and multilateration. In this study, the well known technique of triangulation which requires a minimum of three sensors for the identification of user location is used. Once user location is identified it is simple to determine the associated activity and event via the higher level components of CADB such as preprocessor, feature extractor, and reasoner. The ideal problem of this thesis work was to use RFID tags and readers but latter on changed due to research supply input problem. This forced us to do extra work on developing the simulator. A dynamic-context provider is developed and simulated that generate an arbitrary RFID signal value from three RFID readers for a tagged user device. The RFID reader signal strength values are stored in a database. The stored signal responses for the tagged devices are then classified using a decision tree algorithm as demonstrated under the implementation part of this paper. Figure 4.3 briefly explains how atomic location can be extracted via three RFID readers.

4.3.2 Context Aggregation Layer

The context aggregation layer plays a major role in logical context determination. The contextual data gathered from scattered sensors in pervasive environment should be aggregated to produce high level context information. The representation of high level context information requires an integration of several components in CADB architecture to facilitate the reasoning of context information.

This sub layer of CADB architecture contains decisive components that work in sequential order for the determination of the high level context classes. Figure 4.4 explains how this layer works; it is at this specific layer that the context binder agent takes part to increase the precision and validity of the raw context. At the context negotiator component, the context binder agent intersects the raw context to provide a better feature extraction and classification of the raw context. Research finding including [43, 44 and 50] does not have the component that is proposed at this layer and follow the following pattern of context aggregation.

Preprocessing ➔ Feature Extraction➔ Context Classification

This research add context negotiator component to facilitate context negotiation/agreement on raw context information. The context negotiator component requires another major component which named as Context Binder Agent: part of the multi-agent system of CADB.

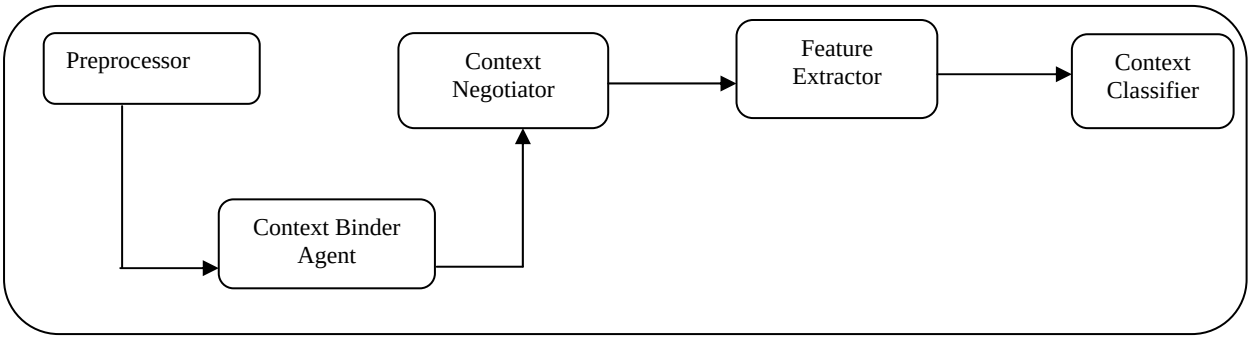


Figure 4-4: Detail of Aggregation layer Components

The paragraphs here under, provide a brief summary of how the components given in figure 4.4 works at the context aggregation layer of the CADB architecture.

PREPROCESSOR:

The preprocessor uses a run time algorithm to bind off-line data with sensor data which is generated dynamically. This preprocessing will consume some amount of time as the sensor information cannot be provided directly to next higher level information. As any context is identified via sensors the preprocessor will make use of the decision tree algorithm provided by Weka (i.e., machine learning tool) and suggest the possible atomic location information.

CONTEXT NEGOTIATOR:

The context negotiator component is an essential component in context aggregation. It collects the low level context data that comes from agents found under the agent manager. The publication and advertisement of the low level contextual data is accomplished by the context binder agent. Before dealing with feature extraction and other process in context aggregation it is really vital to deal with negotiation of raw context so as to increase the validity and correctness of context reasoning. Context level agreement: context data agreed among different agents (such as knowledge agent, application agent, and extended agent) about the unprocessed context data.

FEATURE EXTRACTOR:

This component generates the most appropriate feature of pervasive environment. The activity at this component could be considered as finding the attribute of pre processed raw context. The extracted features will be aggregated and belonging to a certain class which could be an issue of the next component in the process of context aggregation. Precisely feature extraction component of CADB deals with the characterization of information related to location, activity and resources in the user environment. For example extracting temperature, pressure and other features such as tightness of the environment with a lot of people, network capability and service providing capabilities are some of the aspects that can be address by this component.

CONTEXT CLASSIFIER:

This component classifies the features determined to a particular context class. High level context data is generated by combining activity and location features and also by aggregating different context features supplied from the feature extractor component. Unique feature of context enables to create a specialized context class on the top of an existing context class. The classified contextual information is the basic input for the reasoner layer to be discussed in the following section. For instance network capability of the environment can be classified as high, medium and low depending on the features extracted by the feature extractor component.

CONTEXT BINDER AGENT:

The context binder agent will pass the raw context information to other high level agents via the agent platform and agent response will be measured as agreement/negotiation of raw context. This agent is described under Section 4.4.1.2 together with other agents that works in a coordinate fashion with it.

Below is a scenario that elaborates how the different components of context aggregator work to support context aggregation.

1) Consider Eyob is entering into the library, Eyob location can be asserted from three or more RFID-readers by associating the tagged smart phone/PDA he holds.

2) The raw-context data that come from the three RFID-readers will be pre-processed (i.e., converted into identified atomic location of the library).

3) Right-now Eyob is in circulation section of the library, the context binder agent will bind Eyob's current position with his library study schedule.

4) The raw context data gathered so far, published to other agents found in multi-agent environment of CADB.

5) Response of the multi-agent component will be back to the negotiator component. (Detail of how multi-agent work is discussed in Section 4.4).

6) Depending on the context agreement/negotiation of the multi-agent deployed in CADB there are three cases:

> ***Case 6.1**: If all the agents agreed/negotiated on the raw context, then knowledge of the user environment is well known by CADB-MAS system. No context feature extraction and classification. Diary building is straight forward by using existing system knowledge.*
>
> ***Case 6.2**: If some agents (such as Application Agent, Extended Agent, but not Knowledge agent) does not agree/negotiate on the raw context, then knowledge of the environment and user context is partially known by CADB-MAS system. Additional feature extraction for an existing context class will be attempted, followed by updating the knowledge to the respective agents.*
>
> ***Case 6.3**: If agents have little or no knowledge about the environment and user context, then no agreement will be met among agents, thus knowledge capturing and assertion requires the involvement of users. New context features will be extracted followed by creating new context class via the context extractor and classifier components respectively.*

7) For case 6.2 and 6.3 additional feature extraction will be attempted by extracting a number of user environment attributes. Some of the attributes to be extracted by the Extractor component includes: Book title, Author, Volume, Publication year, and other information will be easily extracted from the tag of the book.

8) The last stage is supposed to create or call an existing context class for the features determined in step 6 given above.

4.3.3 Context Reasoner Layer

The aggregated context is assumed to be useful context class information that facilitate and support better reasoning. The context reasoner layer ought to handle very refined and characterized context information to achieve enhanced provisioning of context aware service.

All contextual information at this layer should be inferred with the rules imposed from rule repository and wrapping against the knowledge base repository. The classified context information from the context aggregator layer should be easily mapped to ontology available in the knowledge repository. The reasoner component is responsible to manage the communication and assertion technique among the two repository facilities.

Below is an explanation about the three components and the three storage facilities found under this layer:

REASONER:

This component is the major component that glue the different storage facilities and the time stamp component after reasoning on the classified context information. The Jena reasoner API was used to predict the exact activity of the user on classified contextual events that come from the context classifier. The prediction or inference ability of the reasoner, is highly facilitated via the ontology and the rule repository.

The contextual information being reasoned is then sent to the knowledge base repository, to increase the context prediction of subsequent events. After storing the description about the environment on the knowledge repository, then the reasoner will initiate the diary write up process (on the diary storage) by stamping date and time information through the time stamp component.

TIME STAMP:

The time stamp component stamps date/time information to the data supposed to be stored on the diary repository persistently. The time stamp contains the start/end time as well as the date in which an event/activity occurs at a particular location.

DIARY EXTRACTOR:

The diary extractor component lies between the diary application and diary storage. It is an intermediate component that enables the context aware diary application to extract the information from the stored user's diary. This component is vital for several computational implementation requirements on the stored information. Security, privacy, encryption, and query implementation can be possibly deployed by the means of this component. Issues such as security, privacy and encryption to protect the personal information are not considered in this study.

RULE REPOSITORY:

Among the available reasoning strategy, this study uses a rule based reasoning approach. Rules and policies are viewed in the form of event-conditions-actions and dictate the behavior of the services in reacting to service invocation [23].

In our study, there are a number of public spaces, devices and resources and as well as personal devices and gadgets. It is really important to support mandatory policies set by the space manager, as well as accommodating polices defined by the users. This study proposed and used a number of rules that are vital and supporting the reasoning of user context. For example if a Student S1 (Eyob) holds a device D1 (PDA567), and if D1 is located in atomic location La (i.e. Stage of Meeting hole), and if D1 is connected to another device D2 (i.e. Projector-456). Provided that the distance between D1 and D2 is less than 2meter and assuming the position of D2 is always fixed and configuration and training of the environment is already done. Moreover, the default activity of La in that given time is A1 (presentation), then we follows a mandatory rule to determine Eyob's activity. Accordingly the rule conclude as S1 (Eyob) is engaged in A1 (presentation) for the above stated

conditions. The rules used for our study will be discussed under the implementation part of this study.

KNOWLEDGE REPOSITORY:

The knowledge repository stores the smart environment information using an ontology based representation. Ontology is about the exact description of things and their relationships and it is also a widely used tool for modeling context information [37]. Figure 4.5 depicts the generic, domain and instance consideration of our study based on the implementation scenario produced and used for this particular research.

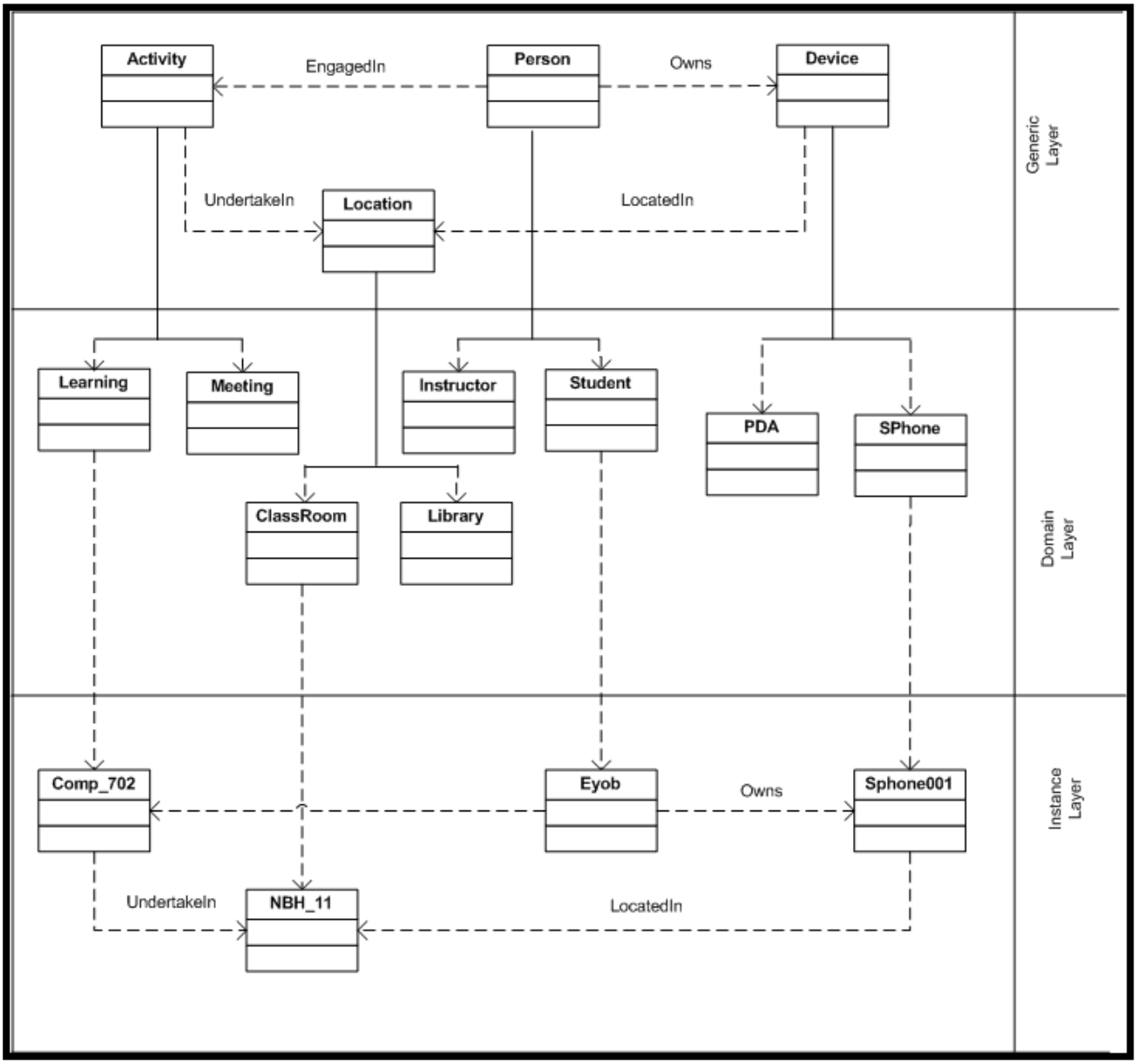


Figure 4-5: UML Representation of EHRAM Model for CADB

The generic context representation of CADB consists of Location, Device, Person, and Activity entities. The relationship Owns, LocatedIn, EngagedIn, and Undertaken are used for the interaction representation of the identified entities. As our study consider a smart campus domain, the generic context information will be mapped to Class_Room, SPhone, Student, and Learning with the indicated relationships. Instances are also mapped from the domain as stated above. Thus NBH-11 is an instance of Class_Room, Sphone_001 is an instance of SPhone, Eyob is an instance of Student, and Comp_702 is an instance of

Learning domain. UML representation of EHRAM: Entity, Hierarchies, Relation, Axiom and Metadata model proposed and used by Ejigu *et al*, [22] were used to elaborate the generic, domain and instances mentioned by this study.

The EHRAM Model allows to having an insight of the knowledge in pervasive domain. It comprises of a number of description about an entity, this study consider only the different entities and their corresponding relationships. Thus, figure 4.5 omits other descriptions such as Axioms, Metadata, etc. Explicate formulation of ontology requires the identification of concepts (entities), relationship, and a number of associated properties for better conceptualization of the domain knowledge suppose to be represented. Table 4.1 depicts partial conceptual representation followed while developing CADB ontology.

Table 4-1: Partial Conceptual Representation of CADB

No	Concepts	Description	Inherits/ Subclass of
1	Diary	Root class of CADB ontology	Thing
2	Person	A person who make use of Context Aware Diary Builder via the devices being tag, in pervasive environment	Diary
3	Location	A location which is asserted by three or more RFID readers as tagged resources are identified.	Diary
4	Activity	Is an abstract class for any activity followed by the user of CADB and reasoned via reasoning capability of CADB.	Diary
5	Device	Any device/resources tagged in pervasive environment.	Diary
6	PDA/SPhone	A device own by a person in pervasive environment and basically tagged.	Device
7	RFID-Reader	A device that track tag devices/resources in smart space environment.	Device
8	RFID-Tag	A tag that is attached with resources in pervasive environment.	Device
9	Meeting	An abstract activity to be held in specific location at specific time.	Activity
10.	Library	An indoor location that provide services.	Location
11.	Student	A person engaged in studying a specific area of study.	Person

List of Object properties identified in the course of developing CADB ontology will be indicated in Table 4.2. Generally, properties of an ontology are grouped in to five (including data type, annotation, depicted and functional properties). The object properties are vital for concept (object)

characterization and relationship illustrations. The object properties listed in the table given below are more elaborated by considering the domain, range and inverse of each property identified in our study. Figure 4.6 and 4.7 provides a pictorial description of two classes namely, the Activity and Person class respectively that we have used for ontology representation of CADB.

Table 4-2: Object Property description of CADB ontology

No.	Properties Name	Domain	Range	Inverse of
1	AccomplishedBy	Activity	Person	EngagedIn
2	AccomplishedIn	Activity	Location	
3	Connects	Device	Device	
4	Contains	Location	Device	FoundIn
5	hasDefaultActivity	Location		
6.	EngagedIn	Person	Activity	AccomplishedBy
7	FoundIn	Device	Location	Contains
8	HeldBy	Device	Person	Owns
9	LocatedIn	Person	Location	OccupiedBy
10	OccupiedBy	Location	Person	LoctedIn
11	Owns	Person	Device	HoldBy

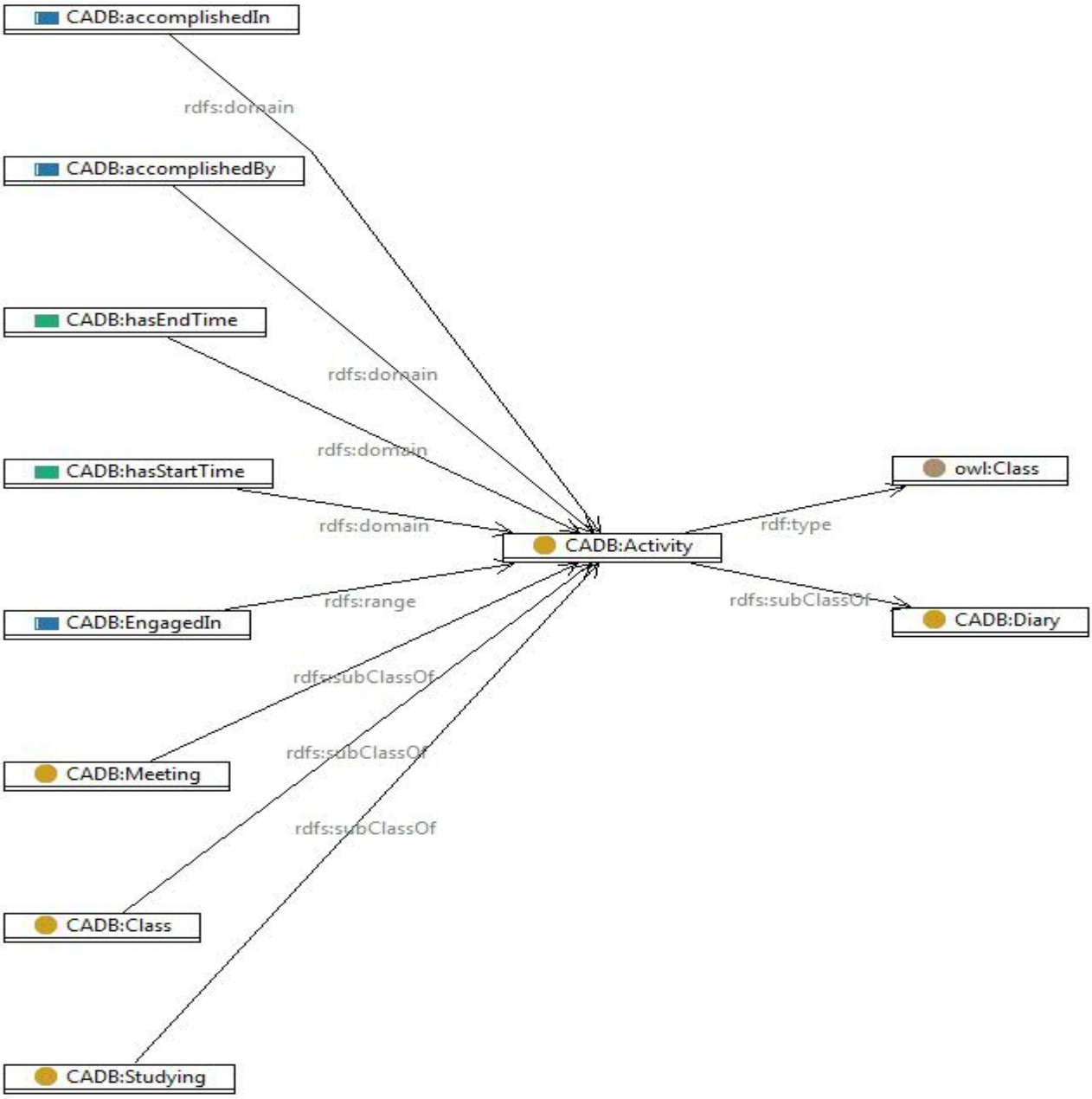


Figure 4-6: Activity class Ontology Graph representation together with subclasses and properties

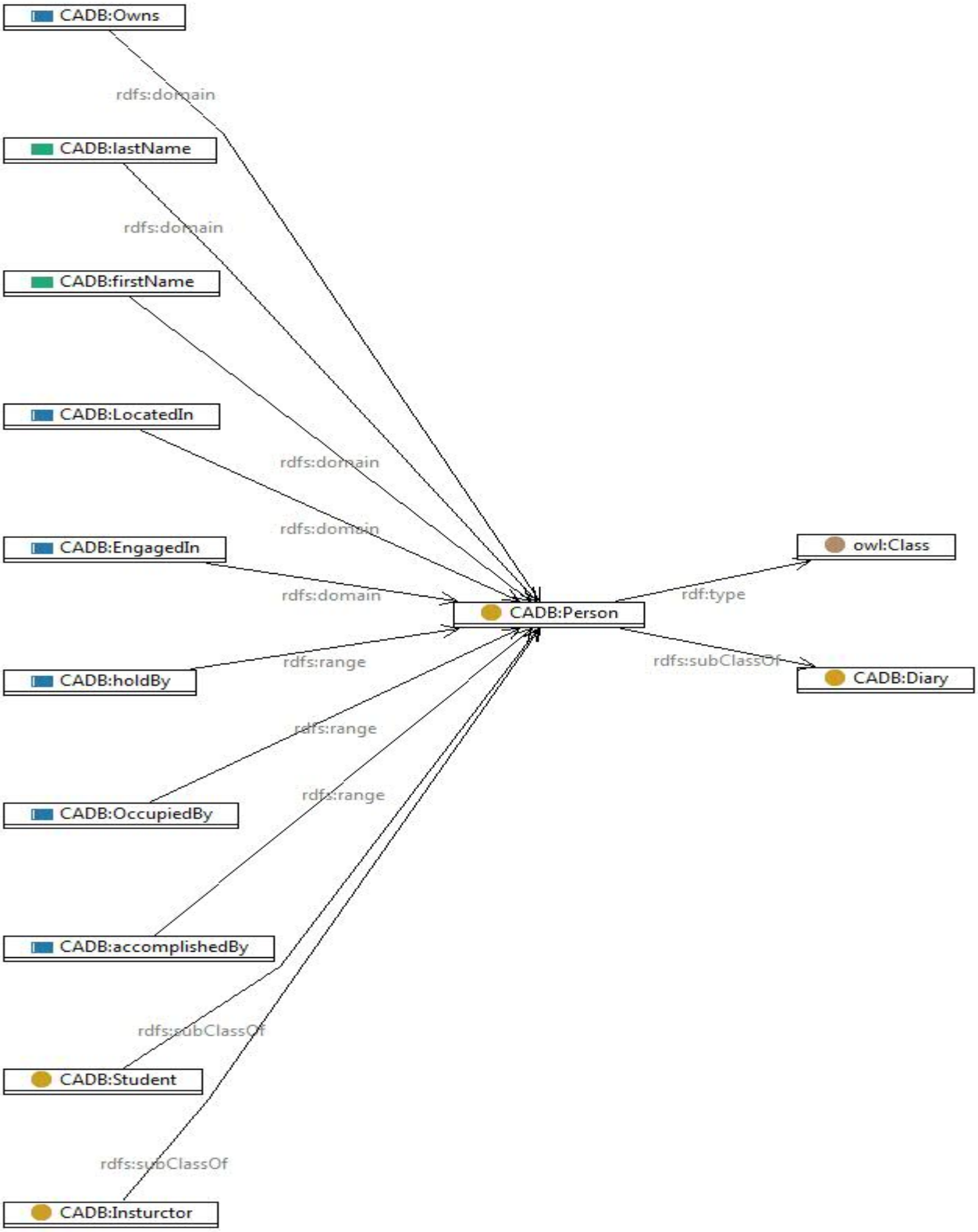


Figure 4-7: Person Class Ontology Graph representation together with subclasses and properties.

DIARY STORAGE:

The Diary storage will contain important information about the user's activity and location together with a short description. The descriptions are assumed to be emotional narration of the daily activity of the user, which cannot be traced by the CADB reasoning facilities. Additional attributes are also identified which support the diary write-up and retrieval process.

The record formats indicated in Table 4.3 is used to store user activity diary under the diary storage facility.

Table 4-3: Diary Table

DID	Date	StartTime	EndTime	ActivityN	LocationN	ActivityD

The identified fields are elaborated as follows:

Field Name	Description	Data Type	Remark
DID	Diary Identification	Integer	Auto Increment, Prim key
Date	Current Date	Date/Time	DD-MM-YYYY
StartTime	Activity/Event Start Time	Date/Time	00:00:00
EndTime	Activity/Event End Time	Date/Time	00:00:00
ActivityN	Name of an Activity	String	
LocationN	Name of Location	String	
ActivityD	Description of an Activity	String	

4.3.4 Application Layer

The application layer is the top most layer that handles tremendous issue in CADB, some of the issues that are address by this layer include:-

- It provides user interfaces, some of which are indicated in figure 4.8;
- It allows the user to annotate emotional description about his/her daily activity;
- It defines a kind of protocol standard being used for CADB (such as HTTP, as in the case of our study); and

- It determines and communicates with resources available in the user environment via a number of sensor network facilities (Including Bluetooth, WiFi, GPS etc).

A) Main Interface **B) Diary Composer**

Figure 4-8: Examples of CADB Application Interfaces

4.4 Detail view of Multi Agent System in CADB Architecture

In this section discuss and explain six agents and their relations for the ultimate of raw context agreement/negotiation to be deployed at the second layer of the proposed CADB architecture.

4.4.1 The Agent Manager

In this study, the agent manager allows six different agents to communicate with each other to support context level agreement/negotiation at the second layer of CADB. The coordination and interaction among the proposed agents brought the concept of multi-agent system management in pervasive computing.

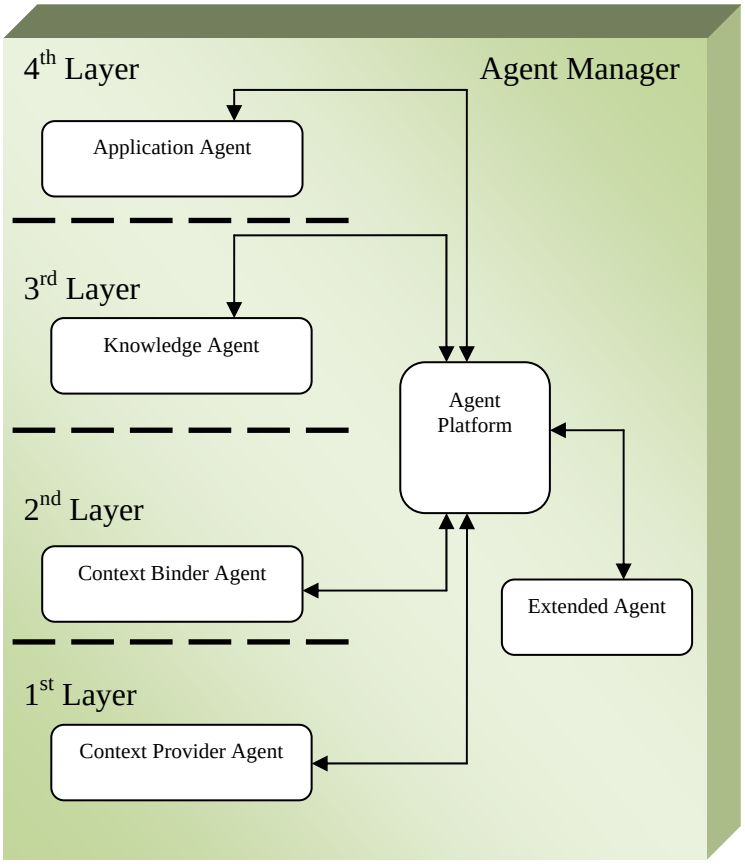


Figure 4-9: Agent Manager in CADB Architecture

The four agents available at each layer of CADB are assumed to be providing the available context information found at their specific layer. Figure 4.9 clearly depicts insight of the multi-agent communication and interaction. As can be seen from the figure the Agent Platform (AP) is a core agent that realizes communication of other agents. Detail of the six agents will be described in the subsections given below. The Context Provider Agent (CPA), the Context Binder Agent (CBA), the Knowledge Agent (KA) and the Application Agent (AA) are found within the CADB architecture given in figure 4.1. The Agent Platform (AP) and the Extended Agent (EA) found at the core of the Agent Manager (AM) component to facilitate integration and coordination among the four agents for better raw context fine-tuning to be used at the second layer of CADB. Below are descriptions of the six agents in our architecture:

The Context Provider Agent

The Context Provider Agent (CPA) found at the first layer of CADB, is responsible for how context is going to be provided to start the diary write-up process. The context provider agent is an aggregate of the user agent (which provides static user context) and the sensor agent (which provides dynamic sensor based context).

The information from the sensor agent is the primary means of possessing context information of the user environment. In addition a raw context data from sensor are dynamic in nature and coined to location, activity, event or any combination user context as explained in Section 4.3.1.2. Shortly, the CPA is also dynamic as it contains senor agent information to initiate and provide important feature of diary writing process. The context information from the sensor agent somehow pre-processed and bind with the user agent information. The user agent contains user profile information in accordance to the user environment location and /or activities mapped from the sensor agent (refer Section 4.3.1.1).

The aggregated contextual information of CPA captured from the sensor interface and from the static context sources (such as user agenda, reminder and to do list) are always updated to support incremental learning of user context together with environment profile.

Generally speaking, the CPA supply diversified user information with respect to the locations and/or the activities of the user. The CPA update the static user context component found at the first layer of CADB and improve the static context information (user environment profile) to supply context prediction, binding, association, and decision. The information from CPA inject to the Context Binder Agent (CBA) after attempting pre-processing, as already discussed in Section 4.3.2 of this work. Detail of CBA will be discussed in the subsection given below.

The Context Binder Agent

The Context Binder Agent (CBA) found at the second layer of CADB, is responsible to accomplish the task of publishing context information so that other agents (such as KA, AA, and EA) can add well versed information regarding the raw context that comes from the CPA.

This agent acting as a bridge which provides raw context information among other high level agents in CADB layered architecture. The purpose of CBA is not limited to supply unprocessed context information; this agent also contains other methods that collect response of higher level context information agents in CADB architecture (such as KA, AA, and EA).

Implementing context agreement/negotiation at raw context will ultimately address a number of issues in building a context-aware diary such as simplifying the effort of diary writing for a known context and building context knowledge to the core service functionality of CADB. To the contrary of a number of approaches discussed in Section 3.2 our approach provide context agreement /negotiation to be implemented at the lower layer during the context aggregation phase.

Once the entire agent deployed in our research agreed/disagreed on certain context information comes from CPA the next process such as feature extraction and context classification will be straightforward. If the KA, AA and EA agreed on raw context comes from CBA and advertised via AP, then the feature extraction and classification will be simpler. It is a matter of calling an existing context class and use it for the current user context with a bit modification regarding the status of environment information.

On the other hand, if the KA, AA, and EA do not agree or no comments totally on the raw context, then feature extraction and classification would not be a simple assignment. Creating new context class and characterizing its feature might require user involvement and conformation regarding the new context class and its corresponding attributes. The implementation part of our study will provide a better summary of how context agreement/negotiation will be possible in the course of developing CADB (refer to Section 5.5).

The Knowledge Agent

The Knowledge Agent (KA) found at the third layer of CADB, wraps the raw context data provided from CBA against the existing ontology which is stored as facts under the knowledge repository. The description about any entities before starting the diary writing process saved on the knowledge repository. The knowledge stored could be represented in the form of ontology and supports prediction, reasoning, and decision; as already explained in Section 4.3.3.

The KA revises its knowledge for the purpose of ensuring a consistent and coherent knowledge base representation of the user environment. The KA triggered a number of vital operations; such as object property and data property validation when some inconsistency is experienced by the knowledge base storage along with the current user environment. In our case, knowledge is detected as a result of user and environment interaction in pervasive environment and represented using ontology.

The task of the knowledge agent is on how to use existing user and its corresponding location, activity, and other vital information to support context prediction, association, decision, etc at the third layer and facilitate intelligence for the entire context prediction.

Figure 4.10 provides an ontology representation used for developing CADB; it explains all the entities together with their relationship and properties.

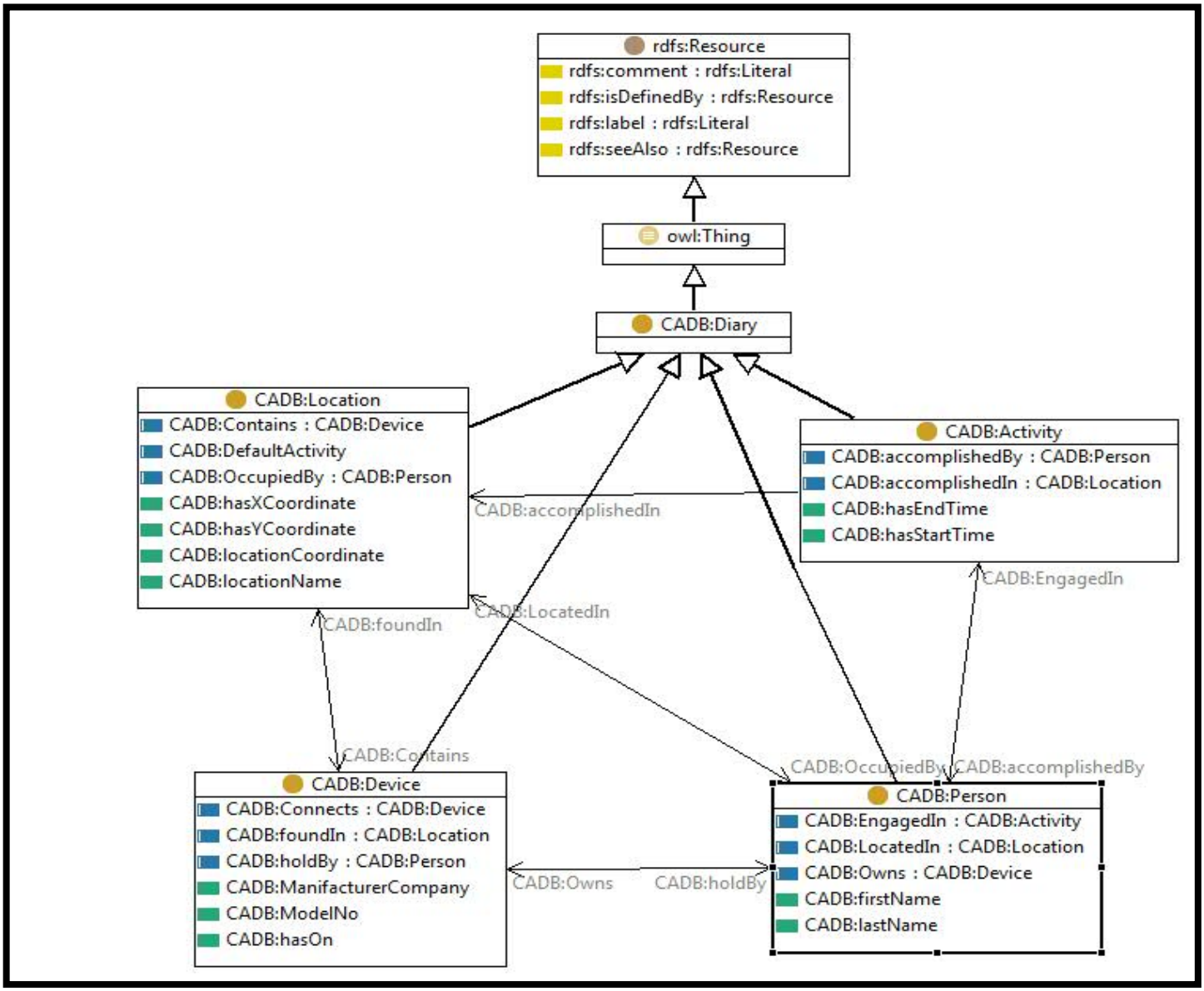


Figure 4.10: Ontology Diagram of CADB

The Application Agent

The Application Agent (AA) found at the fourth layer of CADB architecture. It enables to track the different applications in the user environment and provide their respective responses on unprocessed raw context information. How spontaneous integration of CADB application with other applications are also defined at this specific layer.

The application agent actually tries to correlate the raw context data that comes from CBA with previously maintained information as a summary of user activity. As depicted in figure 4.1 the last layer of CADB will address issues of providing interface for the users, so that users can view diary as a review of activity summary.

The Agent Platform

The Agent Platform (AP) is the core of the multi-agent system in CADB architecture. The AP glues the proposed agents discussed so far, together with the EA.

The Agent Platform advertises the raw context information that comes from Context Provider Agent to the remaining agents found under the Agent Manager (AM). Other agents including the Knowledge Agent, the Application Agent and the Extended Agent will capture the publishe context information and wrap against the high level context information for better characterization and processing of the low level contextual data, as already discussed in the section given above.

The Extended Agent

The Extended Agent (EA) is supposed to support extensibility of the proposed multi-agent approach. Other agents such as the Service Agent (SA), Object Based Environment for Urban Simulation (OBEUS), Multi-Agent Simulation Suit (MASS) and the like stated and discussed in [14] can possibly be registered to the AP and contribute the context level agreement/negotiation at the second layer of CADB. The EA agent is a way to introduce agents that are not found in our designing and actually assume to be an extension effort of the proposed CADB four layered architecture.

4.5 Summary of CADB Architecture

The architecture of CADB discussed and elaborated in this chapter explains how diary writing process is possible using CADB-Core Services (***CADB-CS***) in collaboration with CADB - Multi

Agent System (***CADB-MAS***). As already explained diary writing procedure undergoes through four distinct stages stating from context acquisition up to application layer. The ***Context Acquisition*** will be achieved by means of static user context and dynamic sensor based context provider components. Actually it is the latter component that provides the current user context data. The static user context component will be used in order to predict and support associated context information to the current user context data being identified by the dynamic context component.

The acquired context data though the context acquisition layer will be characterized by the ***Context Aggregator*** components. The pre-processor component will map the raw context information to atomic location data. This will be directly injected to the CADB-MAS components by means of context negotiator component. Low-level raw context agreement/negotiation regarding current user context will be meet by six agents under the multi-agent manger component. The multi-agent core component (i.e., the agent manager) will pass the pre- processed raw context data to all agents found under it and return agent aggregated value. The feature extractor and context classifier component will characterize the raw context data based on the level of aggregation depicted and summarized in table 5.1.

Once the context aggregation is complete the next activity will be more of reasoning. Components found under the third layer of CADB-CS (i.e., ***Context Reasoner*** components) are means to start the diary writing process. The reasoner component communicates with and extracts vital information from the knowledge and rule repositories to predict current user activity. Once the user activity is determined the time information will be plastered to the activity by means of the time stamp component, followed by storing the user activity persistently in the diary storage component. The last layer of CADB architecture: diary A***pplication*** will read/write user activity diary from/to the diary storage via diary extractor component.

In conclusion the four layered architecture of CADB discussed and explained in this chapter introduce a novel way of achieving low level raw-context agreement/negotiation to be accomplished by means of six agents presented in this study. The architecture introduced and used in this research work will be mapped into client/server architecture. Detail of why and how to implement CADB will be explained in the next chapter.

5 CADB Implementation and Discussion

This chapter describes about the implementation issues of CADB. The chapter is structured into nine sections in which the first Section deals with the implementation plan of CADB which covers the tools and technologies used and the mapping of CADB proposed architecture in to client/server architecture. Implementation scenario of CADB is the issue of the second section. The third section will explain the core services of CADB algorithm in a more generalized way. The fourth section will explain implementation detail of CADB reasoning. Section 5.5 describe how Multi-Agent System in CADB is implemented. The remaining section will give details of CADB environment simulator, prototyping and validation of CADB, CADB demonstration and implementation summary of CADB.

5.1 Implementation Plan

This section describes the tools and technology used in CADB and the mapping of the CADB architecture to Client/Server architecture.

5.1.1 Tools and Technology used in CADB

The following list of software tools and technologies were used for the full development and implementation of CADB:

- ***WAMP*** (Windows Apache MySQL PHP) Server Version 2.0i was used to connect client Mobile device with Diary database of CADB Server.
    - ***Apache*** 2.2.11 - an open-source HTTP server for modern operating systems.
    - ***MySQL*** 5.1.36 – used for persistent data base storage at the different level of CADB architecture.
    - ***PHP*** 5.3.0 - general-purpose scripting language originally designed for web development, to produce dynamic web pages.
- ***TopBriad*** composer ME (Maestro Edition) offers comprehensive support for developing, managing and testing configurations of knowledge models and their instance knowledge bases. Composer provides a flexible and extensible framework with a published API for developing semantic client/server or browser-based solutions that can integrate disparate

applications and data sources [72]. We have used TopBraid to develop the knowledge base representation of CADB (i.e. Ontology representation of CADB entities identified in figure 4.5)

- ***Jena*** [65] is an open source Java framework for building semantic web applications. It provides a programmatic environment for RDF, RDFS and OWL, SPARQL and includes a rule-based inference engine. We have used the Jena API for reasoning on user's activity by combining the ontology and rules of CADB.
- ***Microsoft Windows*** operating system is used for CADB server
- ***NetBeans IDE***(Integrated Development Environment) which supports most of the Java platforms
  - *Product Version:* NetBeans IDE 6.5.1 (Build 200903060201)
  - *Java:* 1.5.0_06; Java HotSpot(TM) Client VM 1.5.0_06-b05
    - Java 2 Micro Edition (J2ME) version 2.5.2 Sun Java Wireless Toolkit for CLDC (Connected Limited Device Configuration) of the MIDP (Mobile Information Device Profile) for mobile device simulator development.
    - Java 2 Standard Edition (J2SE) version 1.5.0_06 is used for developing the desktop machine edition of the prototype.

The proposed algorithms are deployed using NetBeans IDE, Jena and Weka libraries were also imported for the full development of CADB prototype.

- ***HTTP*** (HyperText Transfer Protocol) - used for PDA and CADB server communication. This is possibly enabled via CDMA communication scheme.
- ***RFID*** (Radio Frequency Identification) Reader/Tag – for the identification of user and diverse resource in pervasive environment. It is simulated.
- ***WiFi*** technology, the circuits could be considered to describe WLAN products based on IEEE 802.11 standards, using one or more Access Points (APs) as "hot spots" and various numbers of clients [74]. It is simulated.

- ***Weka***- is a collection of machine learning algorithms for solving real-world data mining problems. It is written in Java and runs on almost any platform. The algorithms can either be applied directly to a dataset or called from your own Java code [66]. We have used weka for atomic location mapping and identification under the preprocessor component; it helps us to characterize the simulated location signals.

5.1.2 Mapping of CADB to Client/Server Architecture

The architecture proposed and discussed under chapter 4 will be mapped to a client /server architecture while implementing CADB. The upper most and the lower most layers (i.e. the Acquisition Layer and the Application layer) reside in the client side or more specifically in the hand held devices (PDAs or Smart Phones). On the other hand, the two middle layers of CADB (i.e. the Context Aggregation and the Context Resoner Layer) will reside in the server side. Figure 5.1 will provide a detailed view on how the CADB architecture is mapped to the corresponding client/server architecture.

Even though the nature of pervasive computing requires pee-to-peer setting, the trend of using client/server architecture has a number of advantages as indicated by [1, 27]. Some of the advantages are:

- A possibility to locate and establish communication with devices that are online in a specified domain;
- An administrative tools will interact with server were deployment kits are stored;
- Hide details and minimizing the risk of wrongdoing;
- Every device (i.e., client side devices) is threaded uniformly; and
- It is more secured than peer-to-peer environment. As devices are controlled and configured at the notion of their owner.

According to [5] client-server technology has the following two critical drawbacks:

- Uncontrollable pressure on server and the risk of bottlenecks; and
- Denial of Service

With the indicated limitation, we prefer and propose client/server architecture for the realization of CADB due to the following two reasons: i) resource limitation in PAD/Smart phone, storing day-to-day activities on hand held devices is not realistic totally; and ii) there are a number of online diary service providers, thus existing online diary service providers can improve their service provision by blending context awareness to the existing client-server architecture they have used.

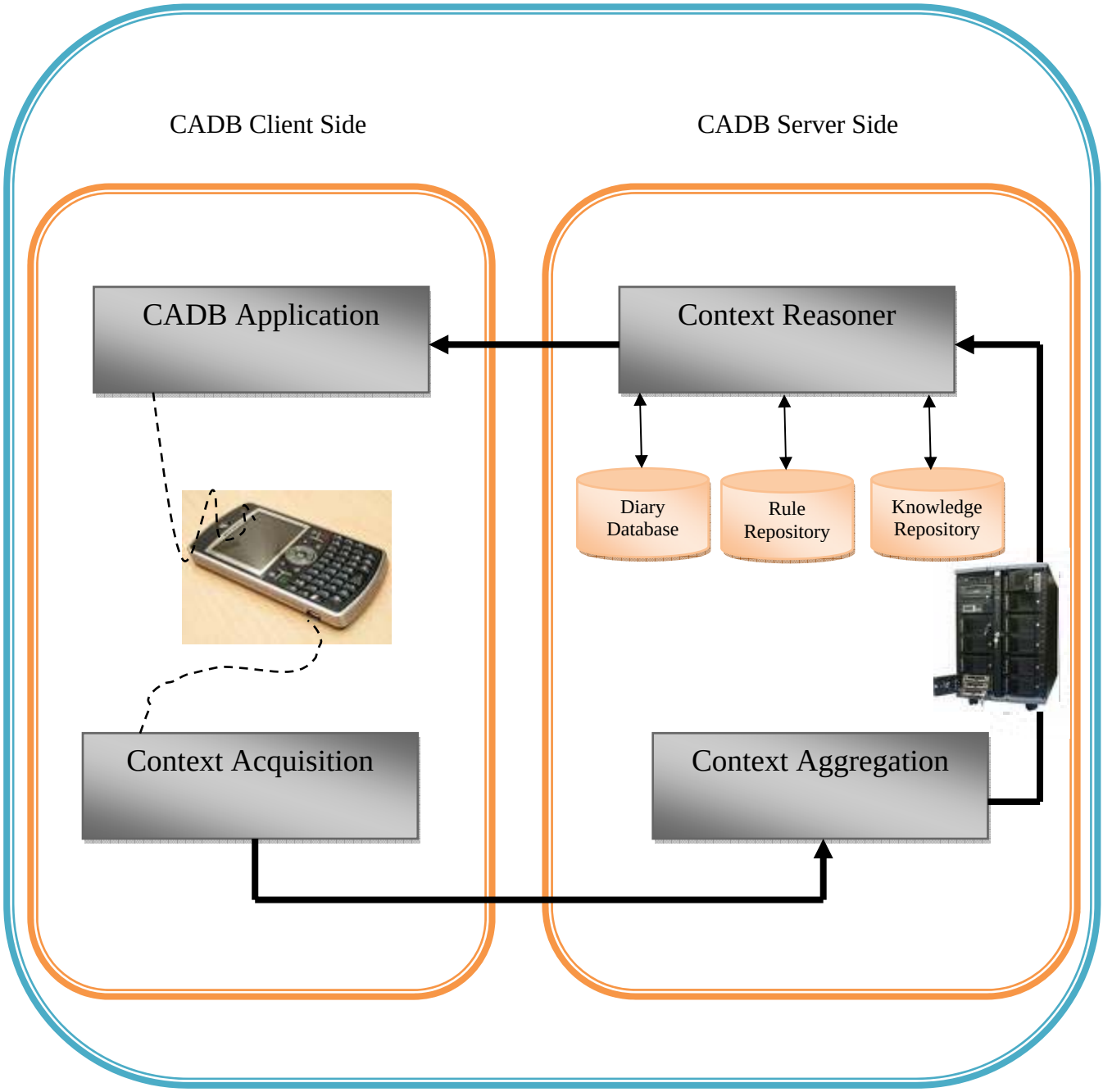


Figure 5-1: Mapping of CADB to Client/Server architecture

5.2 Implementation Scenario of CADB

Pervasive computing projects can be viewed as specific solutions to a particular scenario problem [6]. A number of pervasive computing scenarios seem to envision a unified system of software that acts according to the user's desires and intensions. Researchers also identified that the vision of mobile computing will be enabled by distributed networking, the flourish of mobile technology, and middleware infrastructure. Consider the following Campus student scenario to envision how a Context Aware Diary Builder (CADB) comes into reality in pervasive computing environment.

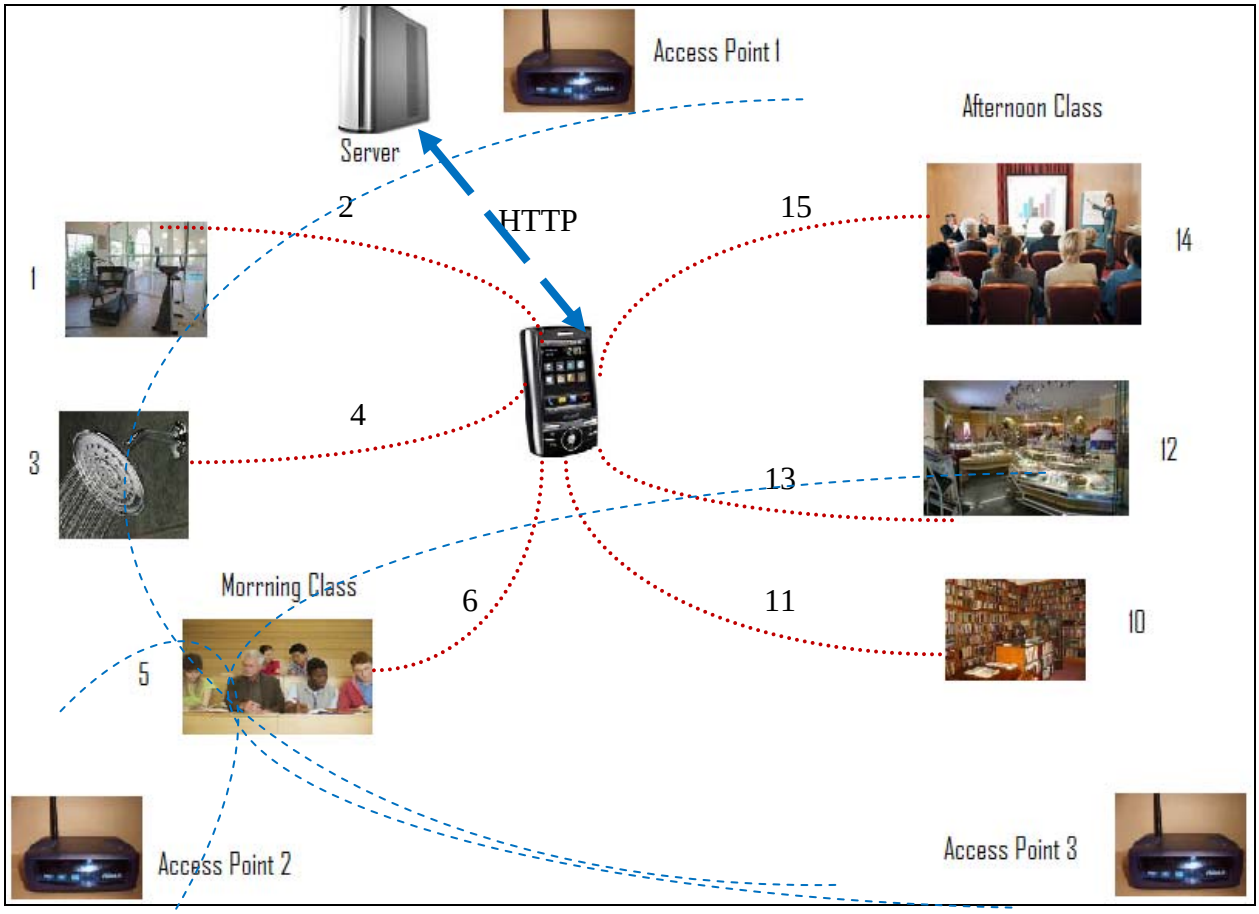


Figure 5-2: Pictorial description of Smart Campus Scenario for CADB

Assumption: Eyob is a university student having a smart phone/PDA enabled with GPS/RFID sensors. The campus environment is surrounded with several sensors, and equipment within the campus is tagged with RFID. Figure 5.2 elaborate the smart campus scenario for the better insight of CADB implementation environment.

The following scenario description explains the possible course of action for figure 5.2 indicated above.

"1) Eyob wakes early in the morning and attempts a daily physical exercise. This event is automatically recorded as he enters to the campus Gym room; 2) The system named as "CADB" records the start and end time of Eyob's exercise. (If Eyob is equipped with some wearable devices it is also possible to measure temperature, blood pressure, etc for the determination of the desirable calories he wish to burn through) ; 3) After completing the exercise he took a daily shower for about 15 minute; 4) This event could be recorded and called directly from the system envisaged to be developed; "

"5) Eyob is in usual lecture room from 8:00-10:00 AM; 6) The system knows Eyob is engaged in class activities by associating the phone of instructor, the functional projector, Eyob's schedule and a collection of class mates. 7) After the class Eyob talk with one of his friend for 25 min; 8) The system lets to record Eyob about the event after the discussion in a very few words; 9) Eyob records about the event or might attempt at his convenient time by clicking on remind me latter button on the display. (The System will remind Eyob at his convenient time, when he is not in a library, or out of any class activities)"

"10) He goes to the library till lunch; 11) The system records title, author, volume, etc of the book from the tagged book and through the RFID reader embedded in the environment; 12) After lunch he went out of campus for a cup of tea with his friends. 13) Social aspect of Eyob is recorded by the system; 14) Eyob returns back to the campus and attend two consecutive classes; 15) The system identifies all the events and records as indicated case in step 7; etc."

5.3 Implementation Detail of CADB Core Service

The implementation of CADB requires the use of a number of technologies and tools as already indicated in Section 5.1 of this chapter. The core service implementation issue will be discussed in this section together with associated software tools and technologies used.

The algorithm given in figure 5.3 is used in the course of diary writing process. The algorithm explains how the entire diary write-up process will be completed in a very precise way using the indicated steps that we developed and discussed in the subsequent section of this chapter.

The input to CADB algorithms (line 1-5 of figure 5.3) includes: the offline user profiles (such as Agenda, Calendar, etc), the current context of the user (such as the location, Activity, Event etc), the current date and time information (i.e., the start and end time of user activity).

The output of CADB algorithm (line 8) is the diary of the user (mainly a summary of daily activity of the user). The remaining lines of algorithm explain the steps required in the process of diary building. The Join function (line 13) will link and analyze the user context and user profile. If there is an intersection between the user current context and the user profile, the Join function will return the number of intersections. One or more intersection is an indication of association of the user previously maintained information with the current user context. Thus, prioritization of user profile will take place in order to simplify and facilitate the diary writing process.

```
1.  Input:  p (Set of user profile i.e. Agenda, Calendar, To do list…)
2.          c (Set of current user context i.e. Location, Activity, Event, Emotion, …)
3.          d (Current date)
4.          st (Activity start time)
5.          et (Activity end time)        //will be append as the context changes
6.  Output: D (User Activity summary as a diary) //aggregate information of c,p,d,st,et
    //steps required to write a Context Aware Diary
7.          d ← current_date              // initialization of system date information
8.          ci ← user_context             // initialization of dynamic user context (Via Sensors)
9.          p ← user_profile              // initialization of  static user profile data
10.         st ← Start_Time               // initialization of  current System time
11.
12. For every ci do
13.         If Join (User Context(ci), User Profile(p)) >=1        // if  any intersection
14.                 Prioritize User Profile          // See figure 5.4
15.         Else
16.                 Call Context Aggregator          //Refer to Annex A1, for ContextAggrigator code
17.          End if
18.         et ← End_Time                 //Activity accomplishment or Context change time
19.         return Di                     //Return aggregate user activity diary
20. End do
```

Figure 5-3: Core Service of CADB Algorithm

As explained in many part of this documentation a user context can be considered as a set that contains {“Location”,”Activity”,”Event” …}. On the other hand, a user profile could also be considered as a set that contains {“Location information”, “Activity description”, “Event explanation”, “Date/Time information ”,….} which could be extracted from user agenda, calendar, reminder, … using a data mining tools. How to bind sensor information with user static context is not the issue of this study, but we highlight how dynamic and static context take part for the overall context awareness provisioning.

The algorithm given in figure 5.4 illustrates how user history prioritization will be accomplished by considering the current context of the user. As the user context drift based on the activity and location of the user, so the profile has to be reorganized to support and facilitate the prediction of user activity.

As long as line 6 of figure 5.4 (No Context Change) is true subsequent lines of steps up to line 11 will be evaluated and executed iteratively. If there exists a context change the function contextChange (c) will return true, otherwise it will return false (i.e., No Context Change will be true). The argument c is an aggregated user context that refers to location, activity, event or any other user context information. Change of user context is possible if there is an alteration of user location, activity, and event.

As the context of the user changes (i.e., context change(c) is true) line 9 will take part in finding and searching the current user context. It will correlate the most probable previous activity of user context with existing profile. Thus line 9, handles the task of binding user context with user profile. It blends user agent and sensor agent data to accomplish the task of prioritization. Detail algorithm of search for existing context is reveal in figure 5.5.

```
1.  Input:  p (set of user profile)
2.          c (set of user context)
3.  Output: PriotizedUserContext
4.          P ← read(UserProfile(p))
5.          C ←Null
6.  While (No Context Change) do
7.                  D ←Write(UserProfile(p))
8.          If Context Chage(C) = True
9.                  Search for Existing Context
10.                 Exit do
11.         End If
12. End do
```

Figure 5-4: CADB Prioritize Algorithm

To search for existing context of user's static data the aggregated dynamic context must be serialized and evaluated by the algorithm described in figure 5.5. The algorithm suggested in figure 5.5, will be realized by the four steps explained and discussed below:

Step 1: After reading user context including location, activity, event, profiles and other relevant context information. For each and every identified user context information do step2;

Step 2: Check if the current user context has at least one common intersection with the user profile information. If there is an intersection at least one to any of the user context (i.e., location, activity, event, etc);

2.1 Initialize the variable Temp by user profile appropriate instance to the user current context.

2.2 count the profile instance

Step 3: If the profile instance count are greater than 2;

3.1 Build a decision tree to identify the most relevant profile instance.

3.2 Copy the predicated user profile.

Step 4: Write relevant user profile to the variable p and return it.

```
1.  Input:  p (Set of user static data including; Agenda, Calendar, To-do list…)
2.          L (Location)
3.          A (Activity)
4.          E (Event)
5.          O (Other Context)
6.  Output: P (UserProfile) //any relevant user profile information from user Agenda, calendar, …

            //steps required to Search the most relevant user context form static source

7.          Li← read(Location(l))              //read user location
8.          Ai ← read(Activity(a))             //read user activity
9.          Ei ← read(Event(e))                //read User event
10.         Oi ← read(OtherContext(o))         //read other user context, if any
11.         pi ← read(userProfile(p))          //read the user static data

12. For every Li,Ai,Ei,Oi, and pi do
13.         Serialize (Li,Ai,Ei,Oi)            //send user context one after the other
14.         If ( Li n pi >=1 || Ai n pi>=1 || Ei n pi >=1 || Oi n pi >=1 )
15.                 Temp[]← pi
16.                 N← Count(Temp[])           //Count function count  the No of pi instances
17.             If (N >=2)
18.                 Build_ID3(Temp[])                  //Construct a decision tree
19.                 Pi ← Predicted(UserProfile(p))     //Prioritize the result of ID3
20.         End IF
21.                 P ←write(UserProfile(pi))
22.         End if
23.         return P                           //Return most relevant user Profile history
24. End do
```

Figure 5-5: CADB Static and Dynamic Binding Algorithm

5.4 Implementation Detail of Reasoning in CADB

Reasoning in CADB mainly implemented at the third layer as already discussed in Section 4.3.3. The capability of Jena ontoModel includes a simplified serialization of Resource Description Framework (RDF) together with rule based reasoning, which provides greater extent for the determination of user activity in our study. Under this section we describe how the ontology being developed to realize CACB using TopBriad is imported and combined with rules that we proposed in Section 5.4.2 using the Jena Model Factory class. This section contains and explains CADB Ontology, CADB Rules and CADB Reasoning strategy while producing CADB.

5.4.1 CADB Ontology Implementation

This section explains and considers a segment code of a class description of our ontology while developing CADB together with its instances. Figure 5.6 contains two RDF of OWL-ontology segment code for CADB. Line 5 up to 10 contains a description of PDA456, which is an instance of PDA subclass that belongs to the Device class (refer figure 4.5). The status and label of **PDA456** descriptions are indicated at line 7 (**on**) and line 9 (**PDA456**), respectively.

On the other hand, line 12 onward describes **Digital Library** which is an instance of Indoor Location subclass that belongs to the Location class of Diary. The label, default activity, and location name of the Digital Library resource description are indicated at line 14(**Digital Library**), Line 16(**Study in Library**) and line 17 (**D-Library**), respectively.

Basically all the classes of diary indicated in Section 4.3.3 and corresponding instances of classes are described in terms of the predefined properties that we propose and used through the development of CADB ontology. This description will be accessed and updated through the process of context aware diary building.

```
1.  <?xml version="1.0"?>
2.  <rdf:RDF
3.     xmlns:CADB="http://www.ContextADB.com/MyCADB.owl#"
……..
4.    xml:base="http://www.ContextADB.com/MyCADB.owl" >
5.    <rdf:Description rdf:about="#PDA456">
6.      <CADB:hasStatus rdf:datatype="http://.../2001/XMLSchema#string">on</CADB:hasStatus>
7.      <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">PDA456</rdfs:label>
8.      <rdf:type rdf:resource="#PDA"/>
9.    </rdf:Description>
10.   <rdf:Description rdf:about="#DigitalLibrary">
11.     <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"></rdfs:label>
12.     <rdfs:label rdf:datatype="http://.....XMLSchema#string">Digital library</rdfs:label>
13.     <CADB:DefaultActivity rdf:resource="#StudyInLibrary"/>
14.     <CADB:locationName rdf:datatype="http:/..Schema#string">DLibrary</CADB:locationName>
15.     <rdf:type rdf:resource="#Library"/>
……..
16. </rdf:RDF>
```

Figure 5-6: Segment Code of OWL Ontology used for CADB

5.4.2 CADB Rules Implementation

The rules found in figure 5.7 run over the ontology resource description OWL file that we described in Section 5.4.1. We have used two user location rules and one activity assertion rule. We describe the user location rule 2 and user activity rule 1 in the paragraphs given below.

UserLocationRule2 description:

" If a device D(i.e., PDA123)is held by a person P (Eyob);

If a device D is connected with another device d (i.e., PC-123);

If a device D is found In atomic location AL1 (i.e., Reading Room);

If atomic location AL1 is a subset of Location L1

Then the rule concludes that P (Eyob) is located in L1 (Library)."

User Activity Rule1 description:

" If a person P (i.e., Eyob) owns a device D (i.e., PDA123);

If PDA123 is connected with another device d (i.e., Projector-456);

If the two devices D and d are found in set of the same atomic location AL2;

If AL2 is subset of Location L1 (i.e., Room224, which is a class room);

If L1 default activity is A (i.e., Learning);

Then the rule concludes that P (Eyob) is engaged in A (Learning)."

```
#Context_Aware_Diary_Builder rule file
#Author Beza Mamo
#Date Sept/01 2009
@prefix CADB: <http://www.ContextADB.com/MyCADB.owl#>
@prefix owl: <http://www.w3.org/2002/07/owl#>
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
# ============================
# CADB:User_Location_Rules
# ============================

[UserLocationRule1:

(?CADB:p CADB:Owns ?CADB:D)

(?CADB:D CADB:foundIn ?CADB:Al)

(?CADB:Al CADB:subsetOf ?CADB:L)

→(?CADB:p CADB:LocatedIn ?CADB:L) ]

[UserLocationRule2:

(?CADB:D CADB:holdBy ?CADB:p)

(?CADB:D CADB:Connected ?CADB:d)

(?CADB:D CADB:foundIn ?CADB:Al)

 (?CADB:Al CADB:subsetOf ?CADB:L)

→(?CADB:p CADB:foundIn ?CADB:L) ]

# ============================
# CADB:User_Activity_Rules
# ============================

[UserActivityRule1:

(?CADB:p CADB:owns ?CADB:D)

(?CADB:D CADB:Connected ?CADB:d)

(?CADB:D CADB:foundIn ?CADB:Al)

(?CADB:d CADB:foundIn ?CADB:Al)

 (?CADB:Al CADB:subsetOf ?CADB:L)

(?CADB:Al CADB:DefaultActivity ?CADB:A)

→(?CADB:p CADB:EngagedIn ?CADB:A) ]
```

Figure 5-7: Sample CADB Rules for user Location and Activity

5.4.3 CADB Reasoning Implementation

This section explains the reasoning procedure followed by CADB. The explanation depicted from step 1 up to step 6 is based on figure 5.8 segment code given in below.

Step1: The first step in reasoning strategy followed by CADB is to acquire the stored knowledge concepts of CADB-Ontology. Line 4 load the ontology we developed in this work and explained in Section 5.4.1 using the loadModel Jena class. Parts of the imported ontology (OWL-file) were already indicated in figure 5.6.

Step2: The rules given in Figure 5.7 were saved with an extension of *rule* and loaded as indicated in line 7 of Figure 5.8.

Step3: The ModelFactory class will merge the ontology and the rules being imported as indicated in line 9.

Step4: Name space definitions and usage in CADB were indicated from line 11-19, which could be vital for wrapping the ontology owl files.

Step5: The output of the Jena reasoner will be saved on the same imported ontology file after reasoning as indicated in line 23.

Step6: Resource, property and literal value accessing and creation will be followed as indicated from line 27 onward of figure 5.8.

```
1.  public class DiaryReasoner {
2.          private static final String Owns = null;
3.      //Loading Imported OWL File
4.          static Model model=FileManager.get().loadModel("file:MyCADB.owl");
5.      //Loading Imported Rule File
6.          static GenericRuleReasoner reasoner=new
7.  GenericRuleReasoner((List)Rule.rulesFromURL("file:CADBRules.rule"));
8.      //Merging Rules into Ontology
9.          static InfModel infModel=ModelFactory.createInfModel(reasoner,model);
10.     //CADB Name Space Defination
11.         static String headerString="prefix CADB: <http://www.ContextADB.com/MyCADB.owl#>" +
12.         "prefix owl: <http://www.w3.org/2002/07/owl#>"+
13.         "prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>"+
14.         "prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>"+
15.         "prefix rdfs: <http://www.w3.org/2000/01/XMLSchema#>"+
16.         "prefix swrl: <http://www.w3.org/2003/11/swrl#>";

17.         static OutputStream output=null;
18.       static String NS = new String("http://www.ContextADB.com/MyCADB.owl#");
19.       static String rdfStr="^^http://www.w3.org/2000/01/rdf-schema#";

20.         public static void main(String[] args)
            {

21.                 try
                    {
22.     //Owl File Path to save the File after reasoning
23. output=new PrintStream(new fileOutputStream("C:/Users/workspace/MYCADB/MyCADB.owl"));
                    }
24.                 catch(Exception err)
                    {
25.                         System.out.println("File related exception !");
                    }
26.     //Resource and Property calling together with Literal definition code
27.                 Resource myUser=model.createResource(NS+"ClassRoom");
                    ……
28.                 model.add(myInstance,RDF.type,myUser);

29.                 Property p1=model.getProperty(NS+"Contains");
                    …….

30.                 Literal l1=model.createLiteral("PC21");
                    …….
31.                 model.add(myInstance,p1,l1);
                    ……
32.                 model.write(output);
                    }
}
```

Figure 5-8: Segment Code for CADB Reasoner using Jena Framework

5.5 Implementation Detail of Multi-Agent System in CADB

The CADB uses multi agent approach for context aggregation to be deployed at the second layer of the proposed architecture. Agents in CADB response a Boolean value (i.e., True (1) or False (0)) based on the existing knowledge of the user context. The return value of the multi-agent system can vary based on the type of application ought to develop.

Accordingly, for this work we consider the learning and intelligence of user environment is highly facilitated by the Knowledge Agent response that is accumulated in the knowledge base. The Knowledge Agent (KA) always wraps the knowledge repository of CADB; we propose and provide a Boolean response value of True (i.e., 1), if knowledge about the user context exists. Otherwise it will return False (i.e., 0).

On the other hand, the Application Agent (AA) monitors the available services in the user environment and responds accordingly. If the desirable service (applications) in the user environment exists, the Application Agent will return a Boolean value of True (i.e., 1). Otherwise it will return 0.

Finally, the Extended Agent (EA) is suppose to return a Boolean value of True (i.e., 1), if some other agent is being registered in the multi agent environment is active and participates for the ultimate intelligence of the user context. Otherwise it will return 0.

Note: The EA is important for the extensibility of our approach. How to integrate EA with our proposed CADB multi-agent system is not the concern of this study.

Table 5.1 summarizes the expected behavior of CADB with respect to the proposed multi agent approach. Consequently, agent responses determine the level of context aggregation to be accomplished at the second layer of CADB. The response level is also an indication of context level agreement/negotiation about the current user context. As can be seen from table 5.1 the response of every agent will be captured followed by deterring the aggregate response value in decimal. The aggregated response of the three agents enables to determine the context aggregation level which can be used for ultimate intelligence of CADB.

Thus, if an environment has **n** possible agents and **x** possible level of responses, then the total number of responses(**r**) could be calculated as:

$$r=x^n$$

Agent responses in the case of our approach can be either Yes (which return a Boolean value of True or 1) or No (in which case the value is False or 0). Thus accordingly using the three higher level agents (i.e. AA, OA, and EA) there is 2^3=8 possible cases or possible combination of responses.

According to this study, the possible set of responses is highly based on the Knowledge Agent and Application Agent response levels. The existence of Extended Agent optional and depend on the pervasiveness of the environment. Basically the knowledge agent response and negligence determine the required level of context agreement/negotiation among all the agents considered in this research work.

Optimum level of context aggregation occurs, if both the Knowledge Agent and the Application Agent respond positively to the current user context (i.e., the response is 1). Average level of context aggregation occurs, if only the Knowledge Agent response is optimistic. On the other hand, if the Ontology response for the current user context is pessimistic, then the level of context aggregation is supposed to be minimum.

The two level responses (1 or 0) used for the three agents considered in this study and can be varied based on the type of application and level of required intelligence that a system suppose to incorporate. The intelligence of our system is highly based on the knowledge agent, followed by the application agent responses. The level of responses for this study is very general in such a way that fixed positive value of 1 returned, if knowledge about the user context exists. Otherwise zero is returned.

Note: As already discussed, the two level responses can be considered as Boolean value (i.e., return “true” or “false”). Additionally this study shows highlight that indicates the agent response level can possibly be more than two levels and to the detail based on the accumulated knowledge of the agent. Thus, agent response handling beyond two levels will be considered as part of our future work.

Table 5-1: Summary of Multi-Agent response in CADB

Possible Cases	Agents	Response	Remark	Aggregation Level
Case 1	KA	1	All agents respond about user context. Little or no effort is required for diary write up process.	***Optimum level of aggregation among the three agents.***
	AA	1		
	EA	1		
Aggregate Response		**7**		
Case2	KA	1	Context information is agreed by two dominant agents (i.e., KA and AA). Little effort is required.	
	AA	1		
	EA	0		
Aggregate Response		**6**		
Case 3	KA	1	KA and EA agent respond for current user context. But the AA responds 0, which implies that the requested application is not available or not known by the CADB.	***Average level of aggregation among the three agents, additional feature extraction about user context requires***
	AA	0		
	EA	1		
Aggregate Response		**5**		
Case 4	KA	1	Only the knowledge about the current user context is known other agents return 0, which requires extracting some features in the user context for the existing knowledge	
	AA	0		
	EA	0		
Aggregate Response		**4**		

Possible Cases	Agents	Response	Remark	Aggregation Level
Case 5	KA	0	The response value of the ontology agent is 0, which is an indication about knowledge of user environment context is not known.	***As the KA response is zero in all the cases. It is an indication for building knowledge of user context (Minimum Aggregation)***
	AA	1		
	EA	1		
Aggregate Response		**3**		
Case 6	KA	0	The same as in the above case, learning about user environment should be taking place by the involvement of the user.	
	AA	1		
	EA	0		
Aggregate Response		**2**		
Case 7	KA	0	The knowledge and service of user context are not known and require adaptation of the environment by the user.	
	AA	0		
	EA	1		
Aggregate Response		**1**		
Case 8	KA	0	This is similar to the initial state of CADB, and requires user involvement for adaption of the use context. High level of user distraction.	
	AA	0		
	EA	0		
Aggregate Response		0		

The three aggregation level summarized in Table 5.1 is implemented in MAS component systematically. Figure 5.9 provides MAS algorithm that explains how the multi agent system can be implemented under the Multi Agent Manger component.

```
1.  Input:   OAR (An integer value of Ontology Agent Response)
2.           AAR (An integer value of Application Agent Response)
3.           EAR (An integer value of Extended Agent Response)
4.           AR (Aggregate integer value of Agent Response)
5.           c (Set of current user context i.e. Location, Activity, Event, Emotion, …)
6.  Output: D (User Activity summary as a diary)
    //steps required to write a Context Aware Diary using a MAS Approach
7.           Advertize Context  To All Agents        //Publish context information to all agents
8.           ORA← Knowleg_Agent_Response(C)
9.           AAR← Application_Agent_Response(C)
10.          EAR← Extended_Agent_Response(C)
11.          AR← Agent_Response_Handler(ORA,AAR,EAR)
12. If AR >=Optimum_Trishold(C)
13.          D←Write(Activity_Description(C))      //  Case1 and Case 2 of Table  5.1
14.  Else IF AR>=Average_Trishold(C)
15.          Search For New Context_Feature(c)
             a. If Context Class Exists
16.                 Assign Existing Context Class
             b. Else
17.                 Crate New Context Class
             c. End If
18.          Call Reasoner
19.          Wrap Against Ontology(c)
20.          D←Write(Activity_Description(c))
21.          Update Agent Information
22. Else
23.          Alert User
24.          Provide Diary Interface
25.          Build New Feature(c)
26.          Crate New Context Class(c)
27.          Create Instance Of Ontology(c)
28.          Build Ontology(c)
29.          D←Write(Activity_Description(c))
30.          Update Agent Information
31. End IF
32. Return D
```

Figure 5-9: Multi Agent System Algorithm implemented under Agent Manger Component

The steps given below will summarize how agent agreement/negotiation take place based on the algorithm given in figure 5.9.

Step 1: Raw context information comes from Context Provider Agent will be advertize to all the remaining agents via the AP. (i.e., to KA, AA, and AA of CADB-MAS)

Step 2: All agents will return its response based on the knowledge and service running in the user environment, followed by calculating aggregated agent response (line 8-11)

Step 3: If aggregate response of agent is greater than or equal to optimum threshold (i.e., 6). Writing user diary is simple as all user context information will be inferred from existing system knowledge

Step 4: Else if aggregate response of agent is greater than or equal to average threshold value (i.e., 4). Search for new context-feature

4.1 If context class for searched feature exists then assign existing context class; otherwise

4.2 Create new context class.

Step 5: Otherwise, the aggregate response of agent is minimum (i.e., below 4). Then adaption of user context will require the involvement of the user. Thus alert the user to confirm and validate the correctness of the context information.

5.6 CADB Environment Simulator

The pervasive computing environments for CADB were simulated using Java codes. This study considers three RFID readers to provide the location of user using the most known method of triangulation. We use RFID-tag for fixed and movable resources (including the user) in CADB environment. The CADB user is characterized by high mobility in simulated environment of CADB; the three RFID-readers identify the location of the user with respect to the signal strength of the radio frequency emitted from each reader.

In this research, Java codes were used while simulating CADB environment. The FRID reader signal ranging from -40 up to -90 db were used for this specific study. Accordingly, Signals near to -40 is assumed to be strong (resources near to RFID-readers). On the other hand, resources far

away from the reader will have a value near to -90. In actual case as indicated by [21] the RFID reader signal values are between 0 to -100 db. The **getRandomizeRFID (RFIDT)** function given below (see Figure 5.10) shows how six different RFID values are generated randomly. In which the argument RFIDT indicate the tagged resource.

```
public static int  getRandomizeRFID( RFIDT )
  {
    String[] rfid1Value={"-40","-50","-60","-70","-80","-90"};
    int randomRfid1Index=(int)(new Random().nextDouble()*6);
    String rfid1Selected=rfid1Value[randomRfid1Index];
    int RFID=Integer.parseInt(rfid1Selected);
    return RFID;
  }
```

Figure 5-10: Get Randomize RFID Function Implementation Code

The function given above can be called during system start up by the three RFID readers simultaneously. Once the value of the three readers initialized the first value of the readers, the function named as **getRandomMovement(RFIDT)** will randomly define the possible course of user movement in a given atomic location. Five possible directions will be randomized using the code given below (see Figure 5.11).

```
public static String  getRandomizeMovement( RFIDT)
  {
    String[] strOperator={"N","E",”W”,”S”,”!”};
    int randomOpIndex=(int)(new Random().nextDouble()*5);
    String selectedOperator=strOperator[randomOpIndex];
    return selectedOperator;
  }
```

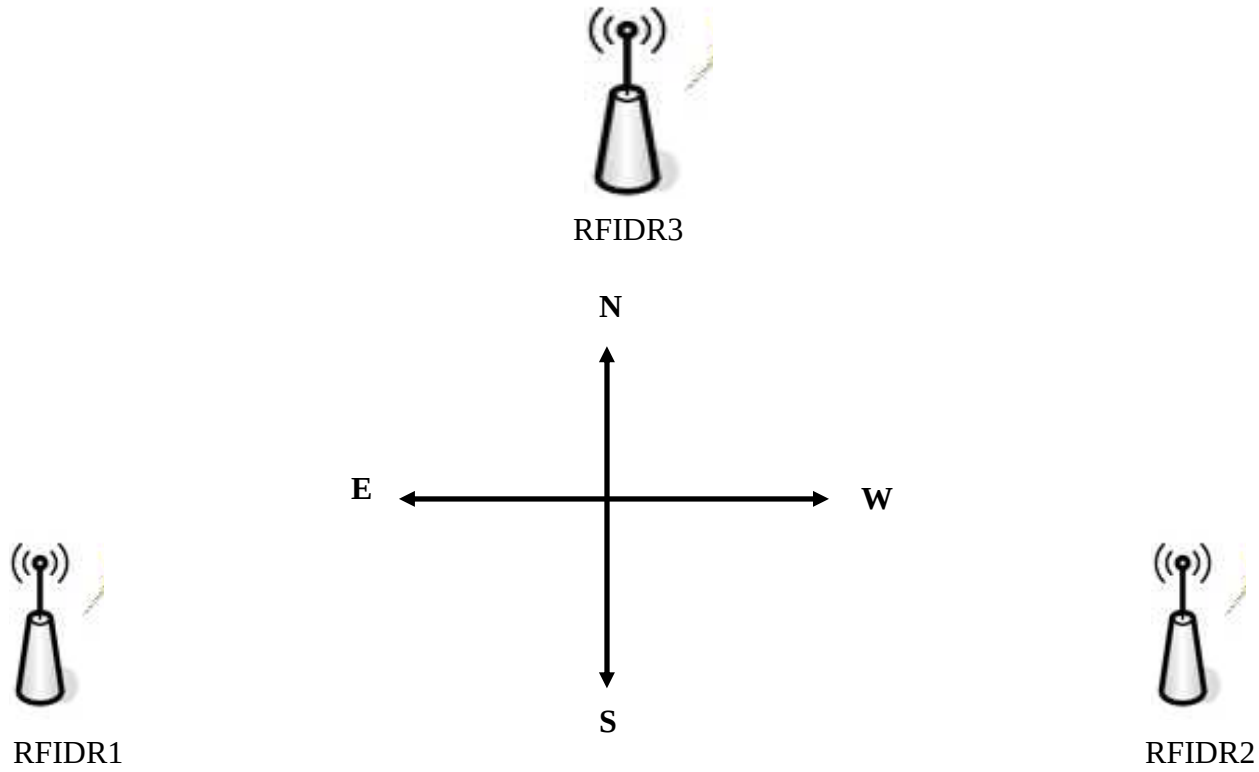
Figure 5-11: Get Randomize Movement Function Implementation Code

The North (N) and the South (S) random directions highly characterized and influenced by the third RFID reader as indicated in figure 5.12. If the value of the signal strength towards RFIDR3 is strong from its current position the user or the resource might move or migrate to the north direction. Otherwise the user will move to south.

On the other hand, East (E) and West (W) random movements are characterized by ether of the first and the second RFID reader (figure 5.11). If the signal strength of RFIDR1is strong (i.e., RFIDR2 signal is weak) then the user is moving towards east. Otherwise the user is moving to the west. No movement are expressed by the exclamation sign (!) in which the readers information are repeated or remains the same.

Once the movement of the user is defined via the getRandomizeMovement function the next task is to bias the reading of the RFID-reader initial value based on the selected direction. For example, if the return of getRandomizeMovement is E (i.e., the user move toward east) the value of RFIDR1 should be Strong, while the value of RFID2 should be weak. Varying the signal strength of the readers can be accomplished by adding or subtracting a common integer value to simulate the movement of a user.

In the course of this work, five distinct user movements were considered in a given atomic location. In reality movements could not be bounded to five directions instead it is arbitrary (such as NE, NW, SE, SW, and other complex combination of directions).



Sample RFID-Reader Value for eleven atomic locations

Atomic-L / Reader	AL1	AL2	AL3	AL4	AL5	AL6	AL7	AL8	AL9	AL10	AL11
RFIDR1	-40	-45	-50	-55	-60	-65	-70	-75	-80	-85	-90
RFIDR2	-90	-85	-80	-75	-70	-65	-60	-55	-50	-45	-40
RFIDR3	-65	-60	-55	-50	-45	-40	-45	-50	-55	-60	-65

Figure 5-12: CADB environment RFID –reader placement and 11 atomic locations

The RFID reader values (in figure 5.12) are considered as a major source of atomic location information. The sample RFID reader's atomic location values were considered as a training data set for preprocessing of sensor data for weka machine learning software (refer appendix C).

5.7 Prototyping and Validation of CADB

NetBin 6.5 environment which blend J2ME and J2EE and allowing to importing the Jena and Weka API were used to develop and test CADB. To implement algorithms proposed so far with the exception of addressing static context binding parts. All the code we write makes use of Java programming languages and run in a modular way.

The simulator developed provides a random location points pumped in 10 second interval and saved in MySQL database environment. The raw context data from the three access points were pre-processes by a machine learning tool, namely Weka. Weka contains and provides a Java API for a number of machine learning algorithms. A decision tree algorithm was used to identify the corresponding atomic location of the raw context. After identifying the atomic location, the raw context will be published to knowledge agent, application agent and extended agent to support context prediction and simplify diary building mechanisms.

All agents registered to the agent platform (including KA, AA, and EA) will wrap the published context information against their respective information base and provide a two level response as explained in Section 5.5.

Agent response will be evaluated and returned as context agreement/negotiation among the agents and followed by optional activity of feature extraction and classification of existing context (see table 5.1).

Reasoning of context data is a vital process to avail high level context information in the course of building a context aware diary. Once the high-level context information is determined (in our case user activity is known such as reading books in the library, attending a lecture in a class room, etc). The next immediate step is to stamp the date and time information before storing the activity information together with other description to the diary storage.

The proposed CADB application interface extract the user activity diary via the diary extractor component as already discussed in Section 4.3.3.

The activity diagram given in figure 5.13 illustrates how the diary write-up process takes place at every component of the CADB architecture. The figure presents the activities undertaken at each layer of CADB. The prototype that we developed to comprehend context aware diary builder will be validated against the activity diagram we presented.

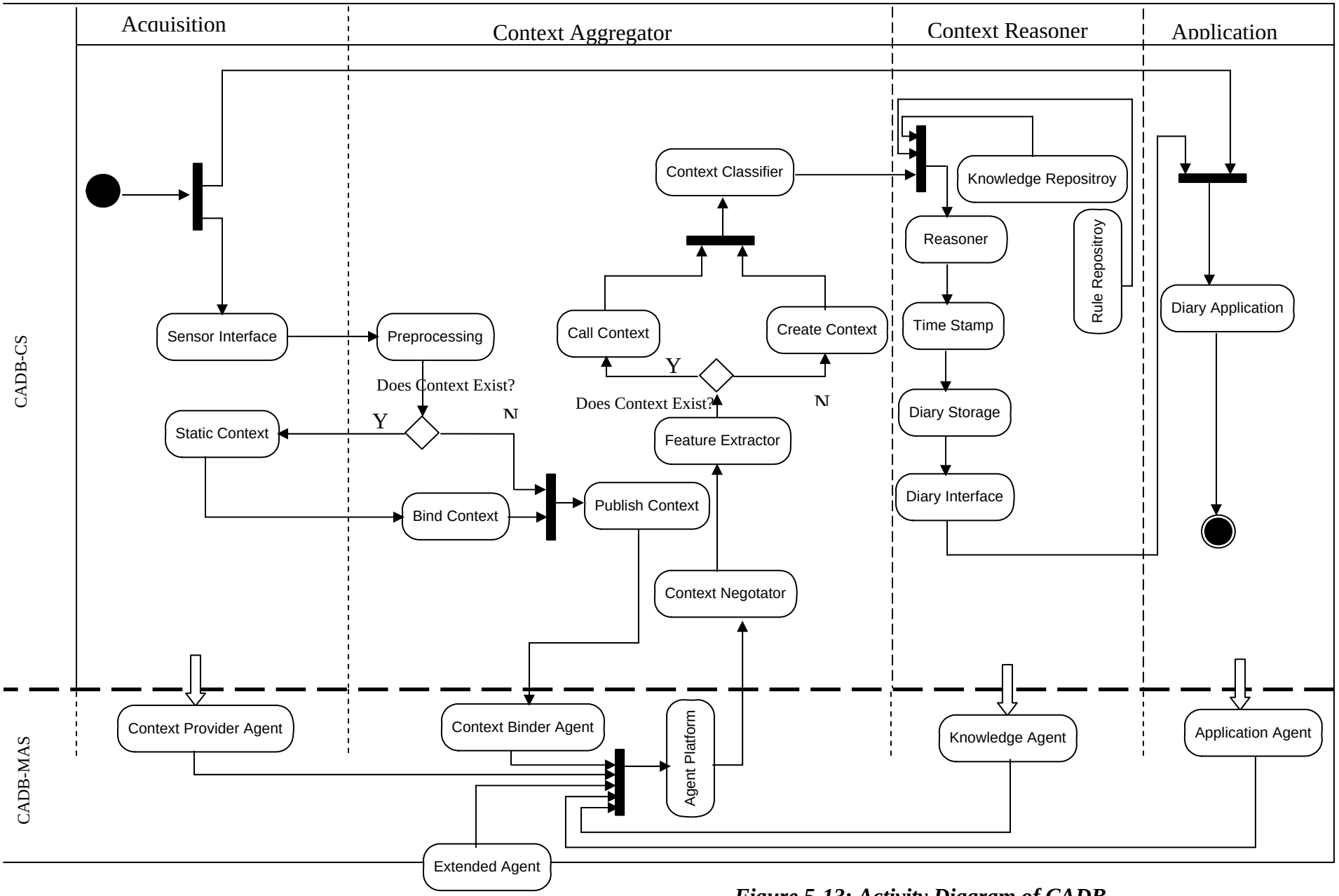


Figure 5-13: Activity Diagram of CADB

5.8 CADB Demonstration

The CADB developed and deployed in this thesis work has a number of user friendly interfaces. Figure 5.14 (a) shows the CADB main application interface that appears to the user at the first glance. Figure 5.14(b) shows the CADB user interface that is used to adapt user environment during system adaption or when the environment is not smart space.

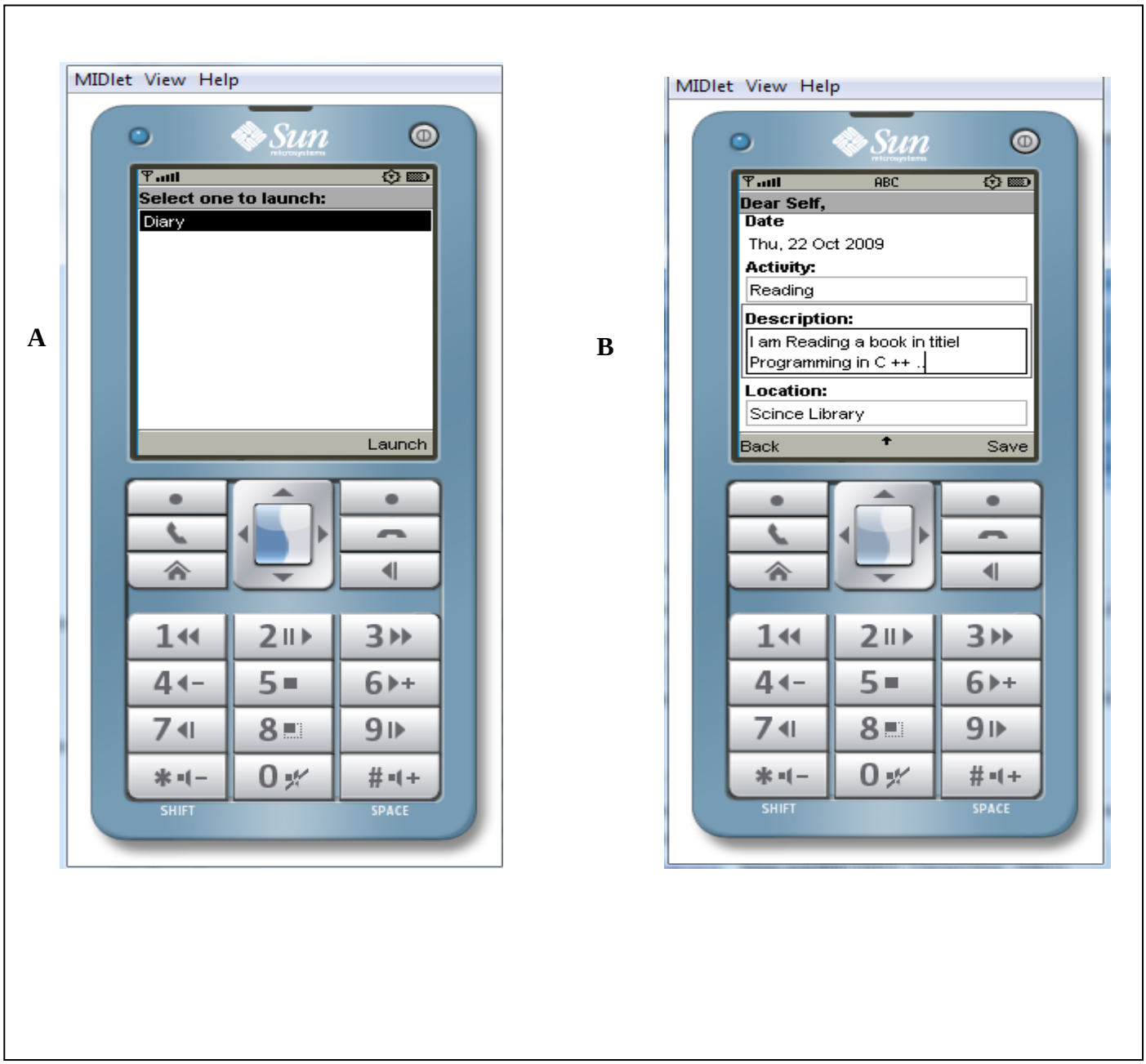


Figure 5-14: CADB Application interface a) Main and b) Adaption

The interface given on figure 5.15 allows the user to select specific date information whenever required and start to compose daily activity.

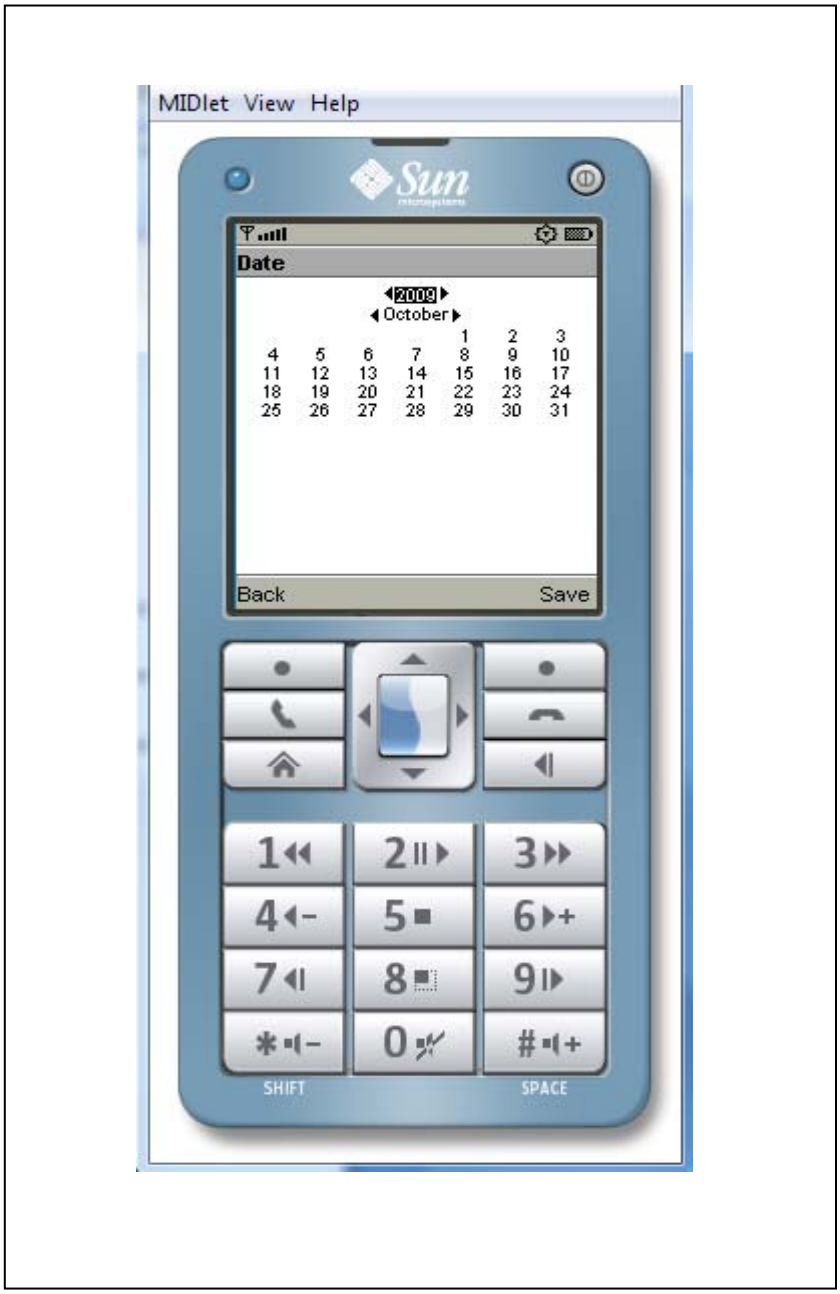


Figure 5-15: CADB Application interface used date selection

Figure 5.16 contains the navigation and edition window. Figure 5.16 (a) enables the user to view user past recorded diary annexed with date and activity list. Figure 5.16 (b) elaborates the CADB application interface after the user complete editing his/her diary.

A

B

Figure 5-16: CADB Application interface a) Diary Viewer and b) Editor

5.9 CADB Implementation Summary

In this section offers a brief summary regarding CADB Implementation and also provides discussion and comparison of CADB with other diary writing approaches that we already discussed and explained in Section 3.3.

This chapter covers a number of issues on how to implement CADB. It includes the assessment of tools and technology for the attainment of context aware diary builder followed by why and how CADB architecture is mapped to client/server architecture as explained in Section 5.1.2. While implementing CADB we have proposed a general implementation campus scenario, this scenario is a more generalized scenario that our system assumes and narrate in many part of this thesis work. The CADB architecture implementation detail is also segmented into two parts namely, the CADB-CS and CADB-MAS.

The three CADB-CS algorithms together with the reasoning strategy followed by this study for the determination of user activity that comprise of CADB ontology, CADB rules and CAD reasoning are also explained by considering a sample scenario and steps whenever required. The CADB-MAS implementation strategy indicates how context level agreement/negotiation will be achieved among a number of agents. The CADB environment simulation techniques and strategy for the purpose of demonstrating our work is also realized under section 5.6. The chapter is concluded by considering the prototype we have developed based on the components we already explained so far and also demonstrating CADB functionality by presenting a serious of screenshots to illustrate the viability of this research work.

Finally, this section specifies how CADB improves existing approaches using the same parameters that we have used in table 3.1. Thus table 5.2 depicts a more detailed explanation how CADB avoids the indicated gaps and introduce pervasiveness to diary writing approaches. From the table we can deduce that CADB satisfy almost all of the parameters that we set as evaluation criteria for desirable diary building approaches. CADB provides and support the very critical concept of pervasive computing "*anywhere-anytime*". The only observable limitation of CADB is lack of satisfying distraction free diary writing (i.e., user involvement on using CADB). Users are forced to review their diary in much less time as compared to existing diary writing approaches. User involvement in case of CADB is observed while narrating emotional expression regarding daily experience of an individual. Extraction of emotional expression is very much complex, subjective,

cannot be generalized and requires a sophisticated sensor network implementation which is not covered by this study.

Table 5-2: Comparison of CADB with existing diary writing mechanisms

Approaches	Parameters							
	Any-where	Any-time	Sharing	Searching/ sorting	Security mechanism	Privacy	Multimedia Integrations	Distraction Free
MD	-	-	-	-	Physical Security	?	-	--
CBD	-	-	-	++	Use computer security	?	+	--
OD	+	+	++	++	Use web security	?	++	--
CADB	++	++	++	++	Pervasive computing	?	++	+
(Key : ++:= Comprehensive +:= Partial ?:= Unknown -:= None)								

6 Conclusion and Future work

6.1 Conclusion

These days, the mode of life of the society is highly dependent on technology. Peoples of the early days make use of manual diary to capture their daily activities. As a result of desktop computer and Internet service, introduction diary writing shifts towards digital information processing and storage. Currently, a number of digital diary software that can be installed and run on standalone computers are available freely. Some service providers also start to offer diary writing and processing online. Pervasive computing introduces a new paradigm of diary writing, which is fully context aware as indicated in the analysis. The notion of this research work is on how to build user activity diary with less distraction of the user (i.e., user's effort to build a diary is just to annotate their emotional expression about the phenomenon in their day-to-day activities).

In this study, the researchers has proposed and developed a model for Context Aware Diary Builder (CADB). It has also been develop a prototype of our proposed model. CADB demands a number of technological issues including smart space (which is equipped with sensor and actuators), advanced mobile devices (which has a capability of receiving diversify sensor information such as RFID, GPS, Bluetooth, etc), and intelligent software system.

Context awareness of CADB is aided by passing through a four layered approach. The acquisition layer obtains knowledge about the environment in relation to the user experience using two possible sources (i.e., the sensors and the user history or profile). After capturing and binding sensor information with user history, the next activity under context aggregation was to pre-process raw context data. The context which was being preprocessed was characterized by feature extractor and context classifier component at the second layer.

The context reasoner layer provides high-level context information by combining ontology and rules on the Jena Java API. Once the high-level context information is identified it can be stored on diary storage after having date and time signature.

In the courses of all the above steps, the researcher introduced how multi-agent system assist and aid the diary building process by manipulating raw context data against their information base.

In summary, CADB covers a number of core issues in pervasive computing including: context acquisition, context aggregation, context reasoning and multi agent system coordination for context agreement/negotiation to be deployed while attempting context aggregation.

6.2 Summary of Contribution

It is believed that the research work plays a role in issues raised under pervasive computing. Below is a list of contribution obtained from this study:

Propose a new approach for diary writing: It was tried to propose a novel approach of context aware diary. Thus, individuals and online diary service diary providers can make use of the effort to simplify and facilitate a context aware diary builder.

Suggest a new context-aware architecture together with its components: The architecture that was proposed and used while producing CADB can fit to a number of context aware application development. Thus, application developer in the area of pervasive computing can make use of the effort and might add their-own component as there is more roof of extensibility in layer approach.

Propose systematic way of static and dynamic context binding: User existing data or profile regarding a location and an activity is helpful in determining the most likely subsequent activities. Even though there are research finding and publication in the aspect of context prediction, to the best of the researchers' knowledge, most of the researches do not provide insight on how historical data bind with sensor information systematically but this research try to give a highlight on how static information and dynamic information could be broken down and aggregated at large.

Context agreement/negotiation for raw context data: A number of researches that uses multi-agent system in pervasive environment strived to attain context level agreement/negotiation at application layer. This study introduced and showed how context level agreement/ negotiation can be realized by means of multiple agents that we have already mentioned and discussed. Moreover, the researcher summarized how agent response will be aggregated and used in developing intelligent context aware systems. Agent response is highly dependent on the accumulated knowledge; our study provides a formula that explains how to determine response level of multi-agent environment in broad.

6.3 Future Work

As pervasive computing is in its infant stage, there could be a number of potential research issues not covered yet. This research identified the following research problem as a feature work:

Increasing precision of dynamic context data: Sensors provide inaccurate and uncertain information in relation to an entity in pervasive environment. In addition, there could be several sensors in user environment that can provide actual and wrong information about the status and location of specific entity. In our case the diary application users might be exposed to an area where the environment is not having a smart space. In such cases the highest priority should be given for GPS reading at least to identify where the user is by associating street name, city name, etc. Even with a smart space environment, there could be a need of identifying the best sensing mechanism such as RFID reader, Bluetooth, GSM, etc. and selecting the best mechanism for the determination of exact context of users. For instance, in a class room environment, GSM or Bluetooth could be suffice and provides more tangible information/data to identify social aspect of the user, as in the case of our work. Identification of sensor matrix by considering the user environment and habitual actions could be one issue to be considered as future research work.

Event and location Catching: Even if pervasive computing environment is characterized by high mobility of resources and users, there are fixed computational assets (devices) such as RFID-reader, standalone computers and server machines. Building knowledge about diversified movable resources together with their metadata may be addressed during the first instance with some configuration specification. Catching location and associated events of movable devices increase the ultimate of context awareness. Questions like: *"for how long the information kept in such devices and what form of security and trust should be developing among diversified devices?"* are considered as future work.

Security and Trust: Security strategy proposed and used by stand-alone computer and small networks (such as authentication and access control) are not sufficient to secure pervasive environment and users. Expectation and right of mobile users to access resources and services; anytime and anywhere leading to the consideration of new security approaches. The security implementation among resources in pervasive environment is highly facilitated by trust among the existing resources. The issue of building trust for the realization of securing pervasive environment is considered as a prospect work in this study.

Beyond 2 level agent response handling: If agents have response level of greater than two which cannot be Boolean as in the case of our study. New approach of agent response handling based on the level of accumulated knowledge will be proposed, followed by systematic aggregation strategy which can be vital to determine agent agreement/negotiation. How to address those issues is beyond the scope of this study.

How to use multiple Medias: In the course of this study, we consider only textual information processing and handling in relation to context aware diary builder. Diary building using hand held devices can be integrated and make use of the existing technologies blended with such devices including camera, video and audio recording systems. Most mobile phones are capable of recoding video, audio and also able to capture pictures; how to make use of those technologies and configured based on the user preference is another potential research direction to be considered in future. For example, a video camera take a picture of the user environment if there exist any context change in users' environment such as placement of objects and other contextual dependent phenomenon are occurring.

REFERENCES

1 Andersson J., “A Deployment System for Pervasive Computing”, Proceedings of the International Conference on Software Maintenance (ICSM'00), pp: 262 , 2000.

2 Bottaro A., Gerodolle A. and Lalanda p., “Service Composition in the Home Network”. Advanced Information Networking and Applications, pp 596-603; May 2007.

3 Bradshaw J.M. “An Introduction to Software Agents. In: Software Agents”, pp: 3-12, AAAI Press, 1997.

4 Brdiczka P., Crowley J.L and Reignier P, “Learning situation models in a smart home”. IEEE Transactions on Systems, Man and Cybernetics (part B); special issue on Human-Centered Computing, 39 (1): pp 56-63. February 2009.

5 Burgess M. and Begnum K., “Voluntary Cooperation in pervasive computing Services”, Proceedings of the Nineteenth Systems Administration Conference (LISA XIX), (USENIX Association: Berkeley, CA), pp: 143, 2005.

6 Butler M. Leusschel M, Allsopp D and Beautement P., “Towards a Trust Analysis Framework for Pervasive Computing Scenarios”, 2003, (Unpublished).

7 Carroll, J, “ Jena: Implementing the Semantic Web recommendations”, technical report HPL-2003-146, Hewlett PackardLaboratories Bristol, URL: http://www.hpl.hp.com/techreports/2003/HPL-2003-146.pdf- Last Date Acessed on Oct 22/2009.

8 Charles M. and Michael J., “Tutorial on Agent-Based Modeling and Simulation Part 2: How to Model with Agents”, Proceedings of the 2006 Winter Simulation Conference, 2006.

9 Chen G. and Kotz D., “Context-Sensitive Resource Discovery”. Proceeding of the First International Conference on Pervasive Computing and communications, pp 243-252. March 2003.

10 Lai J., Leavas A., Chou P., Pinhanez C. and viveros M., “BlueSpace: Creating a Personalized and Context-Aware Workspace”, IBM Research Report, IBM Research Division Thomas J. Watson Research Center, RC22281 (W0112-044) December 11 2001.

11 Chung Cindy K, Pennebaker James W, “Variations in the spacing of expressive writing sessions.”British journal of health psychology; 13(Pt 1):15-21, 2008.

12 Cirletta L., Leclerc T., Siebert J., Chevrier V. and Schaff A , “Towards standards for Pervasive Computing evaluation: using the multi-model and agent paradigms for mobility”, inria-00348386, version 1- 18 Dec 2008.

13 Coronato A., Pietro G. De, and Esposito M., “A Semantic Context Service for Smart Offices”, IEEE International Conference on Hybrid Information Technology 2006.

14 Cynthia N. and Gregory M., “Tools of the Trade: A Survey of Various Agent Based Modeling Platforms”, Journal of Artificial Societies and Social Simulation vol. 12, no. 2 2, 31-Mar-2009.

15 Day A. “Understanding and Using Context”. 5 (1), Personal Ubi Comput 2001.

16 Dey A. and Abowd G. “Towards a Better Understanding of Context-awareness”. First International Symposium on Handheld and Ubiquitous Computing (HCC’99), June 1999.

17 Cook D.J, Youngblood M., Heierman E.O., Gopalratnam K. Rao S. Litvin A. and Khawaja F., “MavHome: An Agent Based Smart Home”, Proceedings of the First International Conference on Pervasive Computing and Communications (PerCom’o3), 2003.

18 Dobson S., Stevenson G., Williamson G., Knox S., Stabeler M., Coyle L., Neely S. and Nixan P., “An open-source infrastructure for pervasive computing”, PerAda Magazine, Towards Pervasive Adaption, February 17 2008.

19 Edmison J, D. Lehn, M. Jones, and T. Martin, “Users’ Perceptions of an Automatic Activity Diary for Medical Annotation and Analysis”, In Proceedings of International Symposium on Wearable Computers, pp 192-193, October 2005.

20 Edmison J., David L., Jones M. and Martin T., “E-Textile Based Activity Diary for Medical Annotation and Analysis”, In Proceeding of the International Workshop on Wearable and Implantable Body Sensor Network, 2006.

21 Ejigu D., “Contxt Modeling and Collaborative Context-Aware Services for Pervasive Computing”, A thesis for the partial fulfillment of doctorial degree, Submitted to the National Institute of Applied Sciences (INSA deLyon), Defended on December 2007.

22 Ejigu D., Marian Scuturici, and Lionel Brunie, “Hybrid Approach to Collaborative Context-Aware Service Platform for Pervasive Computing”, Journal of Computers, Vol. 3, No. 1, January 2008.

23 Etter, R., Costa P.D. and Broens T., “A Rule-Based Approach Towards Context-Aware User Notification Services”. In: Proceedings of the IEEE International Conference on Pervasive Services 2006, , Lyon, France. pp. 281-284. IEEE Computer Society. ISBN 1424402379, Jun 26-29, 2006

24 Floreen P., Przybilski M., Nurmi P., Koolwaaij J., Tarlano A., Wanger M., Luther M., Bataile F., Boussard M., Mrohs B. and Laus S., “Towards a Context Management Framework for MobiLife“, 14th IST Mobile & Wireless Communications Summit, Dresden, Germany, June 19-23, 2005.

25 Flores-Mendez R. A., “Towards Standardization of Multi-Agent System Framework”, The Guide to Computing Literature, Volume 5 , Issue 4 , pp 18-24, (Summer 1999).

26 Funk C., Shulthesis A., Mitic J., and Kuhmunch C., “Adaption of Composite Service in Pervasive Computing environments” ,Pervasive Services, IEEE International Conference , pp 242-249, 15-20 July 2007.

27 Garlan D., Siewirek D.P., Smailogic A. and Steenkiste P., “Project Aura: Towards Distraction-Free Pervasive Computing”, IEEE Pervasive Computing, Volume 1 , Issue 2 , April 2002.

28 Gellersen H-W, Schmidit A. and Beigl M., “Adding Some Smartness to Devices and Everyday Things”, Proceedings of the Third IEEE Workshop on Mobile Computing Systems and Applications December 2000.

29 Goh E., D., Chieng D., Mustapha A.K., Ngeow Y.C. and Low H.K., “A context-Aware Architecture for Smart Space Environment”, IEEE International Conference on Multimedia and Ubiquitous Engineering 2007.

30 Guinard D, Steng S. and Gellersen H., “RealatedGatwarys: A User Interface for Spontaneous Mobile Interaction With Pervasive Services”. San Jose, USA CHI, 2007.

31 Huang X., Barralet M. and Sharma D., “Accuracy of Location Identification with Antenna Polarization on RSSI”, Proceeding of the International MultiConfrence of Engineers and Computer Scientists 2009 Vol I, MESCE 2009, Hong knong, March 18-20, 2009.

32 Julien C., “Context-Aware Middleware Abstractions for Ad Hoc Mobile Computing”, Washington University Department of Computer Science and Engineering, Doctoral Thesis Proposal, August, 2003.

33 Kataria P., Juric R., Paurobally S. and Madani K., "Implementation of Ontology for Intelligent Hospital Wards", Proceedings of the 41st Hawaii International Conference on System Sciences, 2008.

34 Khedr M., Karmouch A. “Negotiating Context Information in Context-Aware Systems” .IEEE Intelligent Systems 21 - 29 , 19(6), November 2004.

35 Kindberg T. Fox, A. “System software for ubiquitous computing”. IEEE, Pervasive Computing 70- 81, 1 (1), Jan-Mar 2002.

36 Klein K. “A new reason for keeping a diary: Research offers intriguing evidence on why expressive writing boosts health” Journal of Experimental Psychology: General (JEP: General) Volume 32, No. 8 September 2001.

37 Krummenacher R. and Strang T., “Ontology-Based Context Modeling”, In Workshop on Context-Aware Proactive Systems Issue: 2007.

38 Lim B. Y., Zhang D., Zhu M. and Zheng S., “Context-Aware Framework for Spontaneous Interaction of Services in Multiple Heterogeneous Spaces”. Proceedings of the 2007 IEEE International Conference on Multimedia and Expo, ICME 328-331, 2007.

39 Louise B., "Designing Ubiquitous Computing Technologies to Motivate Fitness and Health", In Proceedings of the Grace Hopper Conference 2006, San Diego, CA.

40 Macal C. M. and North N. J, “Tutorial on Agent-Based Modeling and Simulation Part2: How to Model with Agent”, Proceedings of the 38th conference on Winter simulation, Monterey, California, pp: 73 – 83, 2006.

41 Maitland J., Sherwood S. Barkhuus L., Anderson I., Hall M., Brown B., Chalmers M. and Muller H., "Increasing the Awareness of Moderate Exercise with Pervasive Computing", In Proceedings of IEEE PerHealth Conference, Innsbruck November 2006.

42 Mark W., "The Computer for the 21st Century" - Scientific American Special Issue on Communications, Computers, and Networks, September 1991.

43 Mayrhofer R., “Context prediction based on context histories: Expected benefits, issues and current state-of-the-art,” in Proc. ECHISE 2005: 1st International Workshop on Exploiting Context Histories in Smart Environments (T. Prante, B. Meyers, G. Fitzpatrick, and L. D. Harvel, eds.), May 2005. part of the Third International Conference on Pervasive Computing

(PERVASIVE 2005).

44 Mayrhofer, “An architecture for context prediction,” in Advances in Pervasive Computing (A. Ferscha, H. Hörtner, and G. Kotsis, eds.), vol. 176, pp. 65-72, Austrian Computer Society (OCG), April 2004. part of the Second International Conference on Pervasive Computing (PERVASIVE 2004).

45 Miller N., Judd G., Hengartner U., Gandon F., Steenkiste P., Meng I-H. Feng M-W., and Sadeh N., “Context-Aware Computing Using a Shared Contextual Information Service”, Pervasive 2004, "Hot Spots (2004)

46 Minas P and Nicolas T, “ An Intelligent Agent based Approach for Service Discovery in Wireless Networks”, Advances in Communications and Software Technologies, World Scientific and Engineering Society Press, pp. 134-139, 2002.

47 Miraoui M., Tadi C. and Amar C., “Architectural Survey of Context-Aware Systems in Pervasive Computing Environment”, Ubiquitous Computing and Communication Journal (UBICC), Vol. 3, No. 3, 2008.

48 Moha Khedr M., Karmouch A. . “ACAI: Agent-based Context-aware Infrastructure for Spontaneous Applications”. Journal of Network and computer Application 19-44,28,2005.

49 Petteri Martin N. M., and Flanagan J. A., “Enabling Proactiveness through Context Prediction” University of Helsinki, June 17th 2005, Helsinki, Finland , 2005.

50 Nurmi P., Martin M. and Flanagan J.A., “Enabling Pro-activeness Through Context Prediction”, publisher Nokia Research center, 2005.

51 Pham Ngoc K. A., Lee Y-K., and Lee S-Y., “Context Knowledge Discovery in Ubiquitous Computing”, OTM Workshops 2005, LNCS 3762, pp 33-34, 2005.

52 Progoff, I. At a Journal Workshop. New York: Dialogue House Library, 1975. P 87.

53 Przybilski M., Nurmi P., "An Architecture to Enable Remote Context Reasoning", Proceedings of the 2005 International Conference on Pervasive Systems and Computing, PSC 2005, Las Vegas, Nevada, June 27-30, 2005 2005.

54 Raphiphan P., "Context Aware Traffic Congestion Estmimation to Compensate Intrmittently Available Mobile Sensors", Tenth Intrnational Confrence on Mobile Data Management: System, Services and Middleware, May18-20 2009.

55 Shehzzd A., Ngo H.Q., Lee S.Y. and Lee Y-K., "A comprehensive Middleware Architecture for Context-Aware Ubiquitous Computing Systems", Proceedings of the Fourth Annual ACIS International Conference on Computer and Information Science, 251 – 256.

56 Huang S. and Mangs J., "Pervasive Computing: Migrating to Mobile Devices: A Case Study", Systems Conference, 2008 2nd Annual IEEE, April 2008.

57 Slatcher Richard B; Pennebaker James W, "How do I love thee? Let me count the words: the social effects of expressive writing.", Psychological science : a journal of the American Psychological Society / APS ;17(8):pp 660-4, 2006.

58 Tsang1 S. L. and Clarke S., "Mining User model for Effective Adaption of Context-aware Applications", Presented at The 2007 International Conference on Intelligent Pervasive Computing (IPC-07).

59 Tsegn V. S-M. and Lin K. W-C., "Exploiting Data Mining Techniques to Support Context-aware M-services in Mobile Web Service Systems, International Computer Symposium, Taipei, Dec. 2004

60 Wilson D.H., Consovo S., Fishkin K and Philipose M., "In-Home Assessment of the Activities of Daily Living of the Elderly", In Extended Abstracts of CHI 2005: Workshops - HCI Challenges in Health Assessment, pp 2130, Portland, OR, April 2005.

61 Yamin A. C., Augstin I. Barbosa J.L.V. and Geyer C.F., "ISAM: a Software Architecture for Pervasive Computing", CLEI Electronic Journal, Volume 8, Number 1, Paper 3, August 2005.

62 Yusheng XU., Zhixin MA., Xiaoyun C. and Lian L., "A Composite Sensor-based Context Modeling Method for Context-aware Pervasive Computing", Proceedings of the International MultiConference of Engineers and Computer Scientists 2008 Vol II, IMECS 2008, 19-21 March, Hong Kong, 2008.

63 http://digital-diary.software.informer.com/download/ - Last Date Accessed on Sept/25/09.

64 http://en.wikipedia.org/wiki/Online_diary - Last Date Accessed on Sept/25/09.

65 http://jena.sourceforge.net/ - last date accessed Aug -17/2009.

66 http://sourceforge.net/projects/weka/, -- accessed on Aug 17/2009.

67 http://wiki.answers.com/Q/List_the_major_advantages_with_layered_approach_to_protocols -last date accessed Aug/13/2009.

68 http://www.124diary.com/diary-journal-tips.asp- last Date Accessed on Oct -03/2009.

69 http://www.compudiary.googlepages.com/ – Last Date Accessed on Sept/25/09.

70 http://www.deardiary.net/ - Last Date Accessed on Sept/25/09.

71 http://www.edailydiary.com - Last Date Accessed on Sept/25/09.

72 http://www.franz.com/agraph/tbc/- last dates accessed Aug-13/2009.

73 http://www.freedownloadscenter.com/Information_managment/ Diary_Keeping_Tools/ Diary_Keeping_Toos/Computer_Diary_2007.html – Last Date Accessed on Sept/25/09.

74 http://www.freshpatents.com/-dt20090625ptan20090160715.php, accessed on Aug 17/2009.

75 http://www.inboxjournal.com/what-is-a-diary-or-journal.php - last date accessed on Aug/17/2009.

76 http://www.onlinediary.net/ - Last Date Accessed on Sept/25/09.

77 http://www.opendiary.com - Last Date Accessed on Sept/25/09.

78 http://www-distance.syr.edu/journal1.html - last date accessed on Aug/17/2009.

Appendices

APPENDIX A: List of Main Java Classes used for CADB:

No	Class Name	Description
1	GregorianCalendar( )	Is a concrete subclass of Calendar and provides the standard calendar used by most of the world. Defined under java.util.GregorianCalendar
2	BufferedReader( )	Read text from a character-input stream, buffering characters so as to provide for the efficient reading of characters, arrays, and lines. Defined under java.io.BufferedReader
3	BufferedWriter ( )	Write text to a character-output stream, buffering characters so as to provide for the efficient writing of single characters, arrays, and strings. Defined under java.io.BufferedWriter
4	J48( )	Classification via Decision Trees in WEKA. Defined under weka.classifiers.trees.J48
5	Instance ( )	Class for handling an instance. All values (numeric, nominal, or string) are internally stored as floating-point numbers. Defined under Weka java API (i.e., weka.core.Instance)
6	setClassIndex( )	Sets the class index of the set. If the class index is negative there is assumed to be no class. (ie. it is undefined). Defined in 4.
7	setAttributeIndices ( )	Sets which attributes are to be Discretized (only numeric attributes among the selection will be Discretized). Defined in 4.
8	Remove ( )	An instance filter that deletes a range of attributes from the dataset. Defined under weka.filters.unsupervised.attribute.Remove.
9	FilteredClassifier( )	Class for running an arbitrary classifier on data that has been passed through an arbitrary filter. Defined under weka.classifiers.meta. FilteredClassifier
10	Model ( )	An RDF model is a set of Statements. Methods are provided for creating resources, properties and literals and the Statements which link them, for adding statements to and removing them from a model, for querying a model and set operations for combining models. Defined under com.hp.hpl.jena.rdf.model.Model
11	GenericRuleReasoner ( )	A reasoner interface that is able to invoke any of the useful rule engine combinations. The rule set can be set after the reasoner instance is created. The mode can be set to forward, backward or hybrid. com.hp.hpl.jena.reasoner.rulesys.GenericRuleReasoner
12	startApp ( )	Signals the MIDlet that it has entered the Active state. In the Active state the MIDlet may hold resources. The method will only be called when the MIDlet is in the Paused state. Defined under javax.microedition.midlet.*

APPENDIX B: A Mysql database ConnectToMYSQL class to saving access point Data

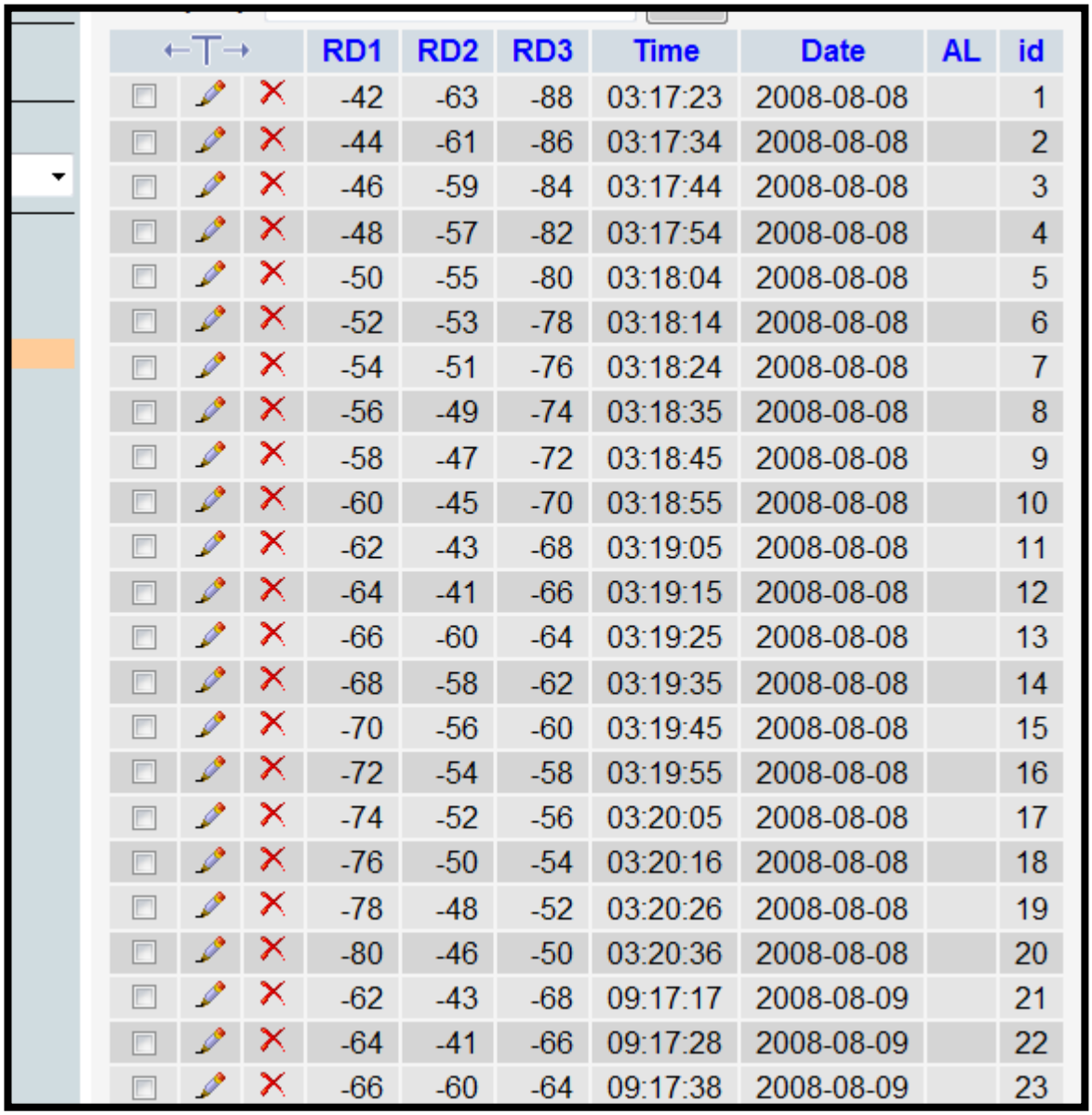
```
package contextawarediarybuilder;
import java.sql.*;
/**
 *
 * @author Beza Mamo
 */
public class ConnectToMYSQL {
        Connection con=null;
        Statement st=null;
        ResultSet rs;
        ConnectToMYSQL(){

try{
        Class.forName("com.mysql.jdbc.Driver").newInstance();
        con=DriverManager.getConnection("jdbc:mysql://localhost/useraccount", "beza","ad");
        }catch(SQLException ex){
        System.out.println(ex.getMessage());
        }catch(Exception e){
        System.out.println(e.getMessage());
        }
        }
        public void createStatement ()throws SQLException{
        st=con.createStatement();
        }
        public void runQuery(String q)throws SQLException{

                st.execute(q);
                rs = st.getResultSet();

                    }
public void closeConnection() throws SQLException{
        con.close();
}
}
```

Mysql view of Access point data of three Readers (RD1,RD2, andRD3) together with date and time information :



RD1	RD2	RD3	Time	Date	AL	id
-42	-63	-88	03:17:23	2008-08-08		1
-44	-61	-86	03:17:34	2008-08-08		2
-46	-59	-84	03:17:44	2008-08-08		3
-48	-57	-82	03:17:54	2008-08-08		4
-50	-55	-80	03:18:04	2008-08-08		5
-52	-53	-78	03:18:14	2008-08-08		6
-54	-51	-76	03:18:24	2008-08-08		7
-56	-49	-74	03:18:35	2008-08-08		8
-58	-47	-72	03:18:45	2008-08-08		9
-60	-45	-70	03:18:55	2008-08-08		10
-62	-43	-68	03:19:05	2008-08-08		11
-64	-41	-66	03:19:15	2008-08-08		12
-66	-60	-64	03:19:25	2008-08-08		13
-68	-58	-62	03:19:35	2008-08-08		14
-70	-56	-60	03:19:45	2008-08-08		15
-72	-54	-58	03:19:55	2008-08-08		16
-74	-52	-56	03:20:05	2008-08-08		17
-76	-50	-54	03:20:16	2008-08-08		18
-78	-48	-52	03:20:26	2008-08-08		19
-80	-46	-50	03:20:36	2008-08-08		20
-62	-43	-68	09:17:17	2008-08-09		21
-64	-41	-66	09:17:28	2008-08-09		22
-66	-60	-64	09:17:38	2008-08-09		23

APPENDIX C: CADB Preprocessing Demonstration using Weka:

Sample training data used by Weka and saved with an extension of arff

@relation contextProvider

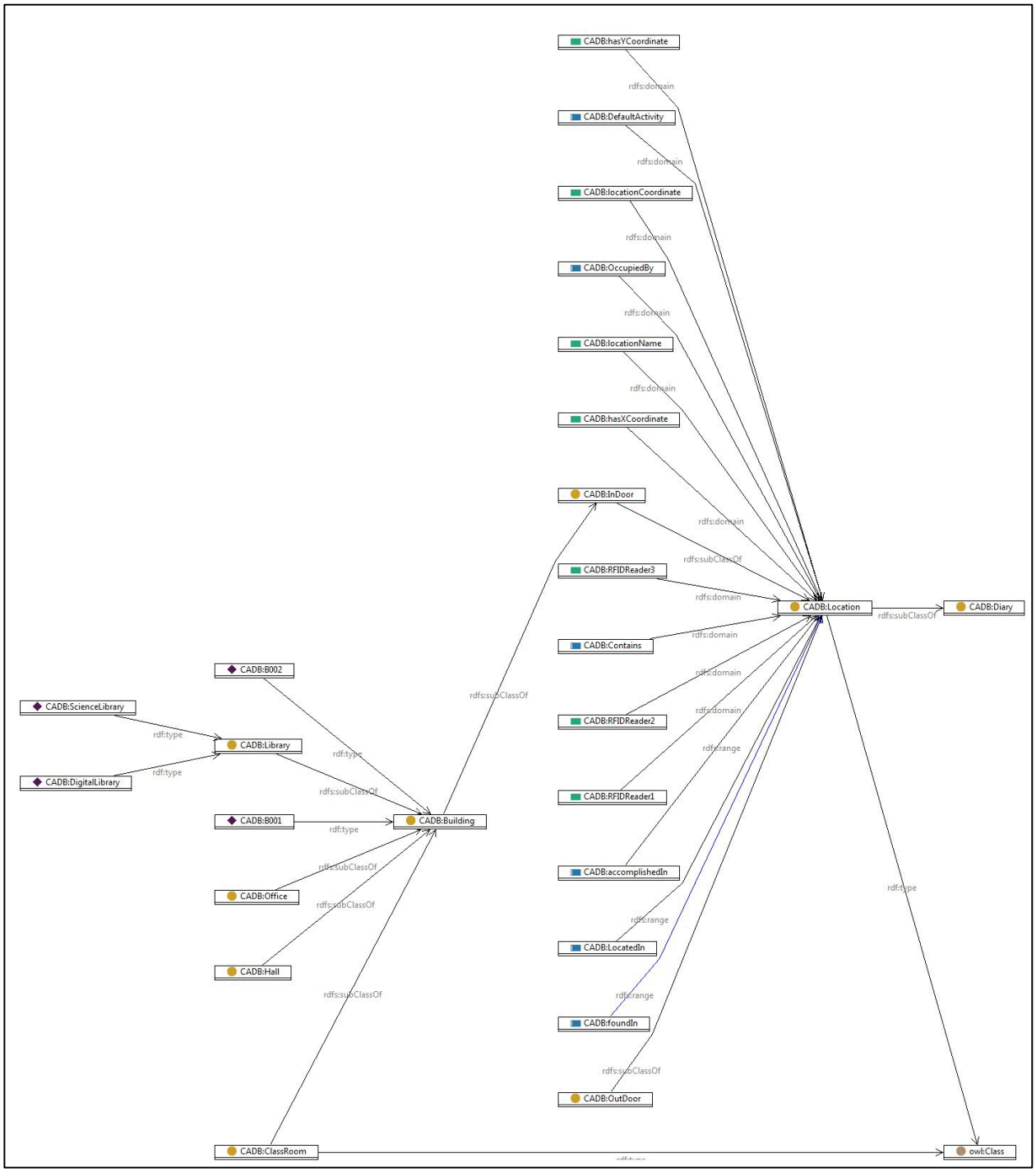
@attribute FRID1 real
@attribute FRID2 real
@attribute FRID3 real
@attribute attomicLoc {circ,re,pr,au}
@attribute Location {lib,hal,class}
@attribute activity {reading,borrowing,attending,speaking,un}

@data
40,90,60,circ_lib
41,89,61, circ_lib,
42,89,61,circ_lib
43,90,60,circ_lib
42,90,60,circ_lib
45,89,62,circ_lib
46,90,62,circ_lib
55,85,75,circ_lib
60,90,40,reading_lib
61,89,41, reading_lib
61,90,41, reading_lib
61,89,40, reading_lib
75,85,55,reading_lib,
90,40,60,audience_meeting
…

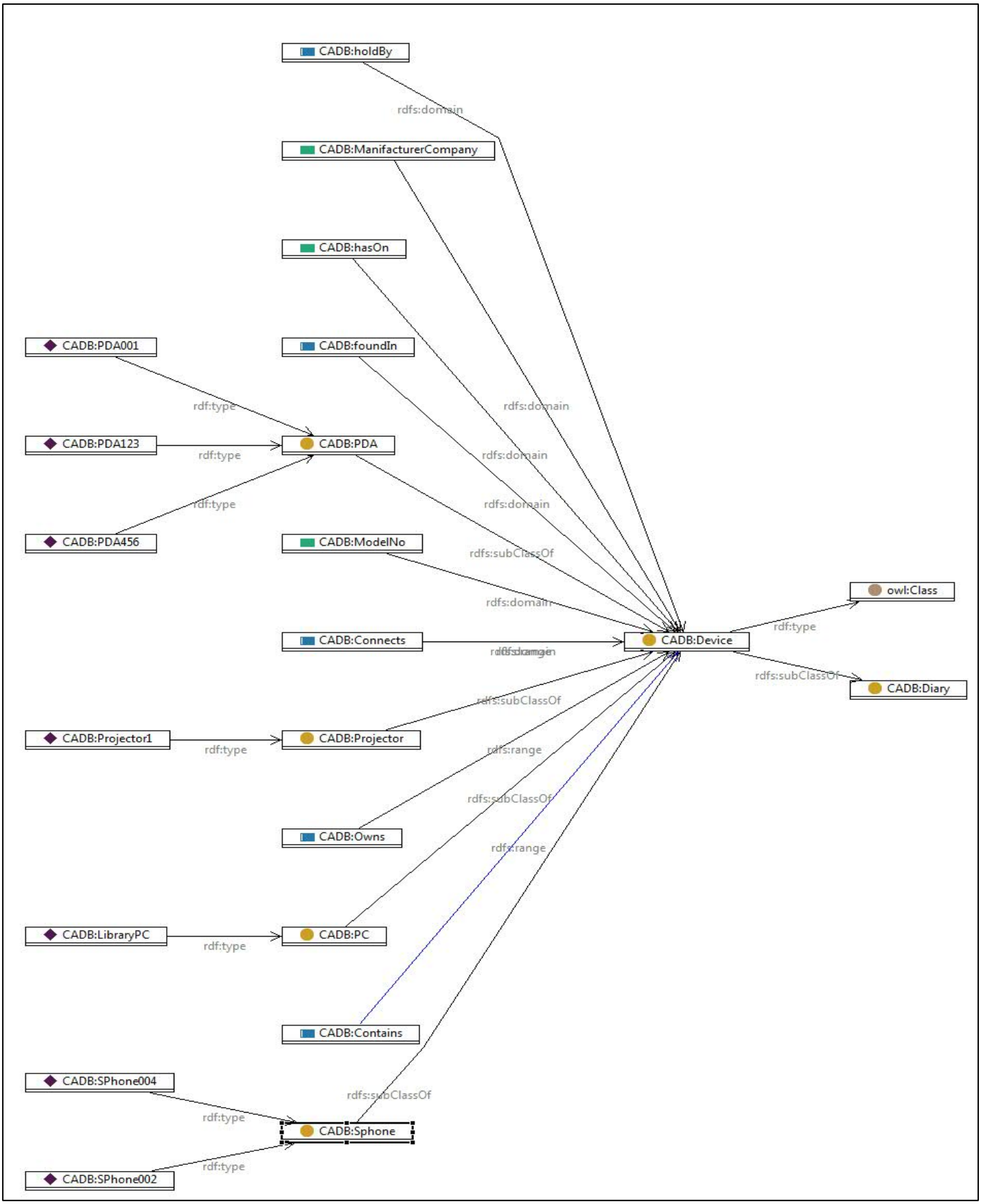
85,55,75,audience_meeting
…
42,89,61,presenter_meeting

…..
54,84,74,presenter_meeting

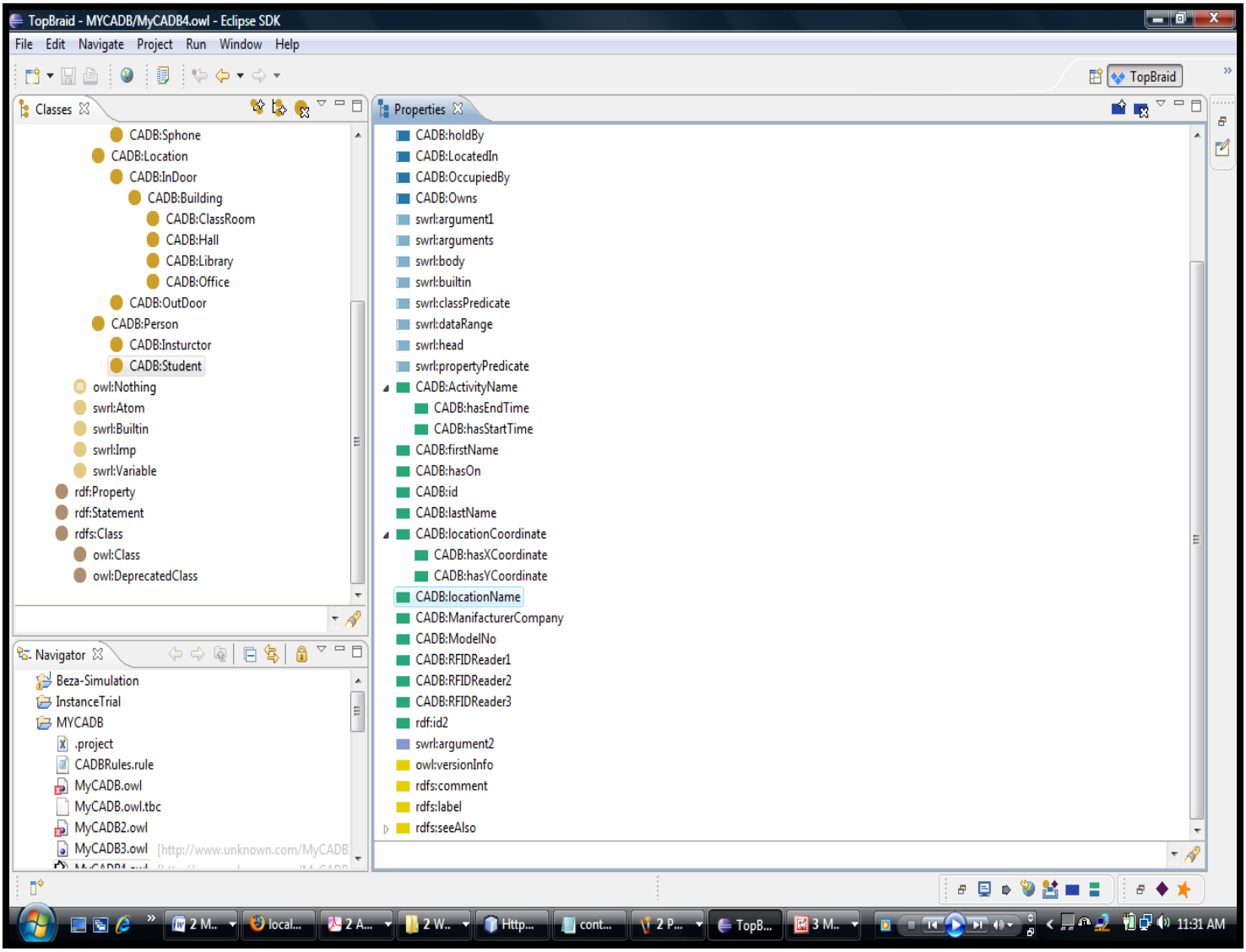
APPENDIX D: CADB Demonstration Sample Ontology:



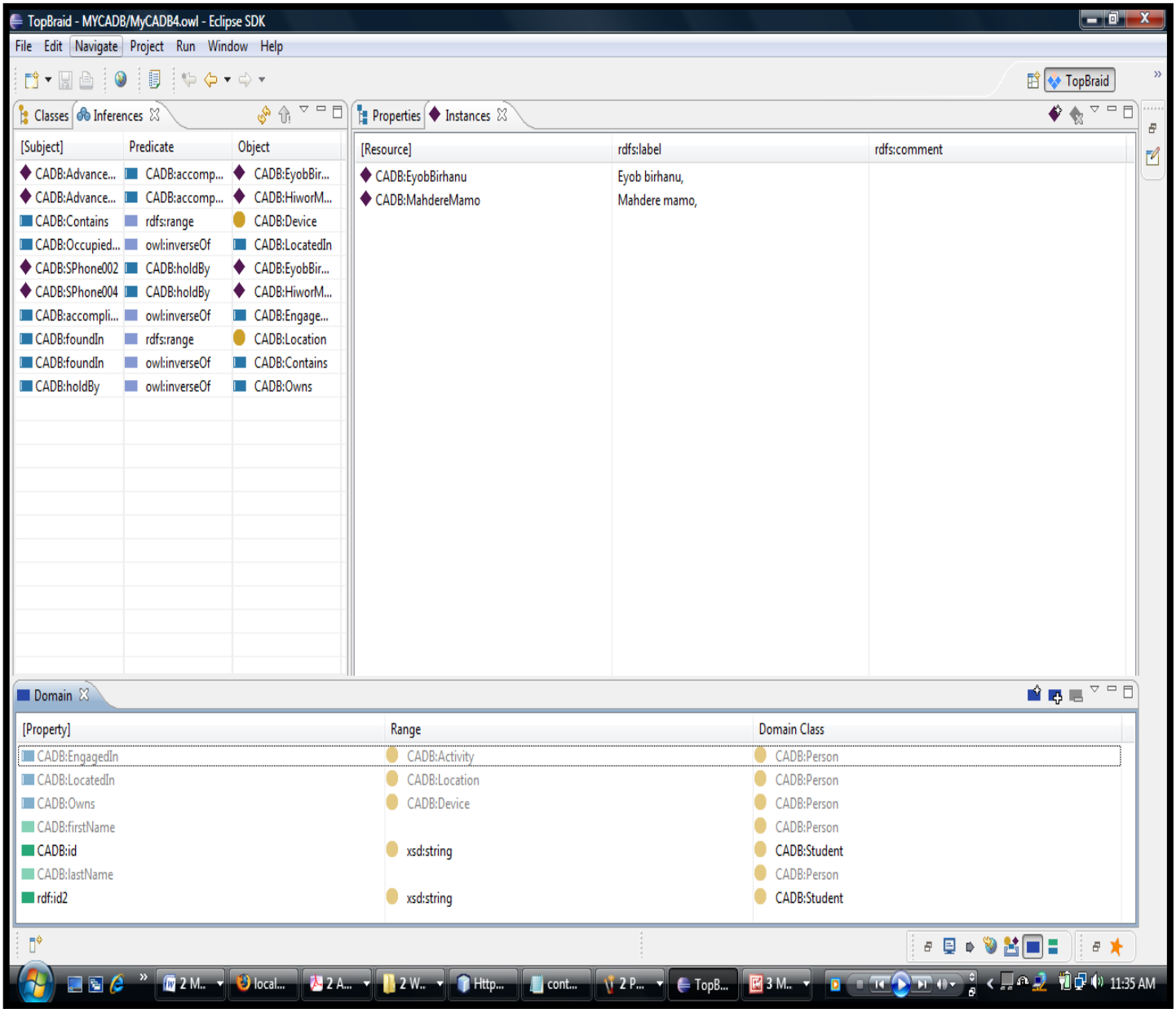
Ontololgy graph up to 4th Level Description for Location Class



Ontololgy graph up to 4th Level Description for Location Device



Sample TopBraid Screenshot of classes and properties for CADB ontology definition



Sample TopBraid Screenshot of Domain, Inference and Instance for CADB ontology definition

APPENDIX E: CADB Demonstration Class for Application Interface using J2ME:

```
package MyPackage;

import java.io.*;
import javax.microedition.io.*;
import javax.microedition.lcdui.*;
import javax.microedition.midlet.*;
import java.util.*;

public class MyDiary extends MIDlet implements CommandListener, Runnable {
   private Display display;
   private TextField location;
   private TextField activity;
   private TextField description;
   private Form form;
   private Command exitcmd;
   private Command savecmd;
   private String date;
   private String time;
   public MyDiary() {
      display = Display.getDisplay(this);
      //date = new DateField("Date",DateField.DATE);
      //time = new DateField("Time",DateField.TIME);
      location = new TextField("Location:", "", 20, TextField.ANY);
     activity = new TextField("Activity:", "", 50, TextField.ANY);
      description= new TextField("Description","",80,TextField.ANY);
      form = new Form("Dear Self...");
      exitcmd = new Command("Exit", Command.EXIT, 2);
      savecmd = new Command("Save", Command.OK, 2);
   }

   public void startApp() {
      Calendar calendar = Calendar.getInstance();
    String h= Integer.toString(calendar.get(Calendar.HOUR)+3);
      String mn=Integer.toString(calendar.get(Calendar.MINUTE));
      String se=Integer.toString(calendar.get(Calendar.SECOND));
    time= h +":"+mn+":"+se;
// obtain the total and free memory, and convert them to strings
        String d=Integer.toString(calendar.get(Calendar.DAY_OF_MONTH));
        String m=Integer.toString(calendar.get(Calendar.MONTH));
          String y=Integer.toString(calendar.get(Calendar.YEAR));
        date= y+"-"+ m+"-"+d;
      HttpConnection c = null;
      InputStream s = null;
      display = Display.getDisplay(this);
```

```
        form.append(new StringItem("","Time :"+ time + "\n"));
        form.append(new StringItem("","Date :"+d+"\\"+m+"\\"+y+"\n"));
        form.append(location);
        form.append(activity);
        form.append(description);
        form.addCommand(exitcmd);
        form.addCommand(savecmd);
        form.setCommandListener(this);
        display.setCurrent(form);
    }

   public void InsertRec() {
        Thread t = new Thread(this);
        t.start();
     }
public void Notify()
{
    Alert notify = new Alert("You save a Record", "Please View Record", null, AlertType.INFO);
    notify.setTimeout(Alert.FOREVER);
}
    public void run() {

        String loc = location.getString();
        String act = activity.getString();
        String d = date;
        String t=time;
        //String da= date;
        StringBuffer b = new StringBuffer("");
        HttpConnection hc = null;
        InputStream in = null;
        String parameter;
        String url = "http://127.0.0.1/insertrec.php?";
       parameter = "location=" + loc;
        parameter += "&activity=" + act;
         parameter += "&date=" + d.replace(' ', '+');
         parameter += "&time=" + t.replace(' ', '+');
        System.out.println(url+parameter);

        try {
            hc = (HttpConnection)Connector.open(url+parameter);
            in = hc.openInputStream();
            int ch;
            while((ch = in.read()) != -1) {
                b.append((char) ch);
            }
```

```
        } catch (IOException e) {
            e.printStackTrace();
        }

      try {
            if(in != null) {
                in.close();
            }
            if(hc != null) {
                hc.close();
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
}

    public void menu() {


    }

    public void tryAgain() {
        Alert error = new Alert("Login Incorrect", "Please try again", null, AlertType.ERROR);
        error.setTimeout(Alert.FOREVER);
        activity.setString("");
        location.setString("");
        display.setCurrent(error, form);
    }

   public void commandAction(Command c, Displayable d) {
        String label = c.getLabel();
        InsertRec();
        if (c==exitcmd)
        {
            notifyDestroyed();
        }
         else if (c==savecmd){
            Notify();
        }
    }

    public void pauseApp() { }
    public void destroyApp(boolean unconditional) { }
}
```

APPENDIX F: CADB Demonstration of PHP files to insert Diary of a user:

```
<?php
$loc=$_GET['location'];
$act=$_GET['activity'];
$date=$_GET['date'];
$time=$_GET['time'];
$link = mysql_connect('localhost','beza','ad')
or die('Could not connect: ' . mysql_error());
//echo 'Connected successfully';
mysql_select_db('CADBDB') or die('Could not select database');
// Performing SQL query
$query = "INSERT INTO diary(No,Location,Activity,Date,StartTime,EndTime)
VALUES(NULL,'$loc','$act','$date','$time','')";
$result = mysql_query($query) or die('Query failed: ' . mysql_error());
echo "OK";
mysql_free_result($result);
// Closing connection
mysql_close($link);
?>
```

APPENDIX G: CADB Demonstration of PHP files to Select Diary of a user by Date:

```
<?php
$link = mysql_connect('localhost','beza','ad')
or die('Could not connect: ' . mysql_error());
//echo 'Connected successfully';
mysql_select_db('CADBDB') or die('Could not select database');
// Performing SQL query
$query = "SELECT * FROM diary ORDER BY RAND()";
$result = mysql_query($query) or die('Query failed: ' . mysql_error());
while($row = mysql_fetch_array($result)){
echo "Date   :".$row['Date']."\n\n";
echo "Time  :".$row['StartTime']."\n\n";
echo "Activity  :".$row['Activity']."\n\n";
echo "Location  :".$row['Location']."\n\n\n";

}
echo "OK";

mysql_free_result($result);
// Closing connection
mysql_close($link);
?>
```

Declaration

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

Declared by: Name: Beza Mamo

Signature: _____
Date: _____

Confirmed by advisor: Name: Dr. Dejene Ejigu

Signature: _____
Date: _____

Place and date of submission: Addis Ababa University October 2009.