A Knowledge-Oriented, Context-Sensitive Architectural Framework For Service Deployment In Marginalized Rural Communities

A thesis submitted in fulfillment of the
requirements for the degree of

DOCTOR OF PHILOSOPHY

of

RHODES UNIVERSITY

by

Mamello Thinyane

January 2009

Abstract

The notion of a global knowledge society is somewhat of a misnomer due to the fact that large portions of the global community are not participants in this global knowledge society which is driven, shaped by and socio-technically biased towards a small fraction of the global population. Information and Communication Technology (ICT) is culture-sensitive and this is a dynamic that is largely ignored in the majority of ICT for Development (ICT4D) interventions, leading to the technological determinism flaw and ultimately a failure of the undertaken projects. The deployment of ICT solutions, in particular in the context of ICT4D, must be informed by the cultural and socio-technical profile of the deployment environments and solutions themselves must be developed with a focus towards context-sensitivity and ethnocentricity.

In this thesis, we investigate the viability of a software architectural framework for the development of ICT solutions that are context-sensitive and ethnocentric[1], and so aligned with the cultural and social dynamics within the environment of deployment. The conceptual framework, named PIASK, defines five tiers (presentation, interaction, access, social networking, and knowledge base) which allow for: behavioural completeness of the layer components; a modular and functionally decoupled architecture; and the flexibility to situate and contextualize the developed applications along the dimensions of the User Interface (UI), interaction modalities, usage metaphors, underlying Indigenous Knowledge (IK), and access protocols.

We have developed a proof-of-concept service platform, called KnowNet, based on the PIASK architecture. KnowNet is built around the knowledge base layer, which consists of domain ontologies that encapsulate the knowledge in the platform, with an intrinsic flexibility to access secondary knowledge repositories. The domain ontologies constructed (as examples) are for the provisioning of eServices to support societal activities (e.g. commerce, health, agriculture, medicine) within a rural and marginalized area of Dwesa, in the Eastern Cape province of South Africa. The social networking layer allows for situating the platform within the local social systems. Heterogeneity of user profiles and multiplicity of end-user devices are handled through the access and the presentation components, and the service logic is implemented by the interaction components. This services platform validates the PIASK architecture for end-to-end provisioning of multi-modal, heterogeneous, ontology-based services. The development of KnowNet was

[1] In this dissertation, the word "ethnocentric" is used to mean "with a focus on a specific culture, culturally informed and culturally aware". It is not to be understood to imply any cultural superiority or any notion of ethnic assimilation.

informed on one hand by the latest trends within service architectures, semantic web technologies and social applications, and on the other hand by the context consideration based on the profile (IK systems dynamics, infrastructure, usability requirements) of the Dwesa community. The realization of the service platform is based on the JADE Multi-Agent System (MAS), and this shows the applicability and adequacy of MAS's for service deployment in a rural context, at the same time providing key advantages such as platform fault-tolerance, robustness and flexibility. While the context of conceptualization of PIASK and the implementation of KnowNet is that of rurality and of ICT4D, the applicability of the architecture extends to other similarly heterogeneous and context-sensitive domains.

KnowNet has been validated for functional and technical adequacy, and we have also undertaken an initial prevalidation for social context sensitivity. We observe that the five tier PIASK architecture provides an adequate framework for developing context-sensitive and ethnocentric software: by functionally separating and making explicit the social networking and access tier components, while still maintaining the traditional separation of presentation, business logic and data components.

Acknowledgements

I would like to thank everyone who has contributed, in many different ways, to the success of this research.

I am greatly indebted to my supervisors Prof. Alfredo Terzoli and Dr. Peter Clayton for their invaluable insight, support and direction in this research. I have definitely seen further by standing on the shoulders of these two giants :-)

I thank my friends and family for being in my life, and for their encouragement.

I am grateful to my wife, my partner, Hannah Thinyane for her unconditional love and support. Thank you.

My utmost gratitude to God, who holds everything together [Col 1:17].

I would like to thank the Dwesa community for welcoming us into their lives and allowing us to ”grow together”. I also thank the Siyakhula Living Lab research group.

The financial assistance from the Andrew Mellon foundation towards this research is hereby acknowledged. Opinions and views expressed are those of the author and do not reflect those of Rhodes University or the Mellon Foundation.

This research was undertaken within the Telkom Center of Excellence (CoE) at Rhodes University.

The endless cycle of idea and action,
Endless invention, endless experiment,
Brings knowledge of motion, but not of stillness;
Knowledge of speech, but not of silence;
Knowledge of words, and ignorance of the Word.
All our knowledge brings us nearer to our ignorance,
All our ignorance brings us nearer to death,
But nearness to death no nearer to God.
Where is the Life we have lost in living?
Where is the wisdom we have lost in knowledge?
Where is the knowledge we have lost in information?

[T. S. Eliot, 1934]

Contents

List of Figures

List of Tables

List of Acronyms

AA Aspectual Analysis

AAA Authentication, Authorization and Accounting

ACL Agent Communication Language

AMS Agent Management System

ASR Automatic Speech Recognition

BPEL Business Process Execution Language

CAPEX Capital Expenditure

CBD Component Based Development

CBR Case Based Reasoning

CKML Conceptual Knowledge Markup Language

CL Content Language

CPE Consumer Premises Equipment

CORBA Common Object Request Broker Architecture

CSS Cascade Style Sheet

DAE Distributed Agent Environment

DAML DARPA Agent Mark-up Language

DBMS DataBase Management System

DCMI Dublin Core Meta-data Initiative

DCOM Distributed Component Object Model

DF Directory Facilitator

DL Description Logics

DTD Document Type Definitions

EDI Electronic Data Interchange

FIPA Foundation for Intelligent Physical Agents

FOAF Friend of a Friend

FOL First Order Logic

FOSS Free Open Source Software

GFP Generic Frame Protocol

GGG Giant Global Graph

HCI Human Computer Interaction

HTML Hyper Text Markup Language

HTTP Hyper Text Transfer Protocol

ICT4D ICT for Development

ICT Information and Communication Technology

IEEE Institute of Electrical and Electronic Engineers

IK Indigenous Knowledge

IP Intellectual Property

JADE JAVA Agent Development Framework

JRMI JAVA Remote Method Invocation

KB Knowledge Base

KBA KB Agent

KBLA KB Local Agent

KBONTA KB ONTology Agent

KIF Knowledge Interchange Format

KQML Knowledge Query Manipulation Language

KR Knowledge Representation

KRS Knowledge Representation System

LEAP Lightweight Extensible Agent Platform

MAF Multi Agent Facility

MASIF Mobile Agent System Interoperability Facility

MAS Multi-Agent System

MMIA Multi-Modal Interaction Agent

MPA Media Presentation Agent

NLOS Non Line Of Sight

OAA Open Agent Architecture

ODBC Object Database Connectivity

OIL Ontology Inference Language

OKBC Open Knowledge Base Connectivity

OLPC One Laptop Per Child

OMG Object Management Group

OML Ontology Markup Language

OOP Object Oriented Programming

OPEX Operational Expenditure

OWL Web Ontology Language

PBX Private Branch eXchange

PDA Personal Digital Assistant

PPPoE Point-to-Point Protocol over Ethernet

PWSA PIASK Web Services Agent

RDF Resource Description Framework

RDF(S) RDF Schema

RDQL RDF Query Language

RMA Remote Management Agent

RSS Really Simple Syndication

SECI Socialization, Externalization, Combination and Internalization

SHOE Simple HTML Ontology Extensions

SIA Socially Intelligent Agents

SIP Session Initiation Protocol

SIPA SIP Agent

SIPMA SIP Management Agent

SKIF Semantic Knowledge Interchange Format

SMTP Simple Mail Transfer Protocol

SMTPA SMTP Agent

SNA Social Networking Agent

SOAP Simple Object Access Protocol

SOA Service Oriented Architecture

SPARQL Simple Protocol and RDF Query Language

SUMO Suggested Upper Merged Ontology

SW Semantic Web

SWSA Semantic Web Service Architecture

TMA Theory of Modal Aspects

TTS Text-To-Speech

UDDI Universal Description Discovery and Integration

UI User Interface

URI Universal Resource Identifier

VOIP Voice Over Internet Protocol

VSAT Very Small Aperture Terminal

VXML Voice XML

WSDL Web Service Description Language

WSIG Web Services Integration Gateway

WSMF Web Services Modeling Framework

WS Web Service

WSA Web Services Architecture

XML eXtensible Markup Language

XMPP eXtensible Messaging and Presence Protocol

XMPPA XMPP Agent

Chapter 1

Introduction

> "I know it's a foolish question," says Jones, "but did you ever by any chance run into a fellow named Ben Arkadian? He's an old friend of mine, manages a chain of supermarkets in Detroit..."
>
> "Arkadian, Arkadian," the Englishman mutters. "Why, upon my soul, I believe I do! Small chap, very energetic, raised merry hell with the factory over a shipment of defective bottlecaps."
>
> "No kidding!" Jones exclaims in amazement.
>
> "Good lord, it's a small world, isn't it?"

The excerpt above is taken from a 1967 publication by Stanley Milgram which set out the results of an empirical study into the degrees of separation between any two random people in the world [1]. Milgram came to the conclusion that there were, on average, six hops between any two people. The publication paved a way to a number of inquiries and discussions: the small world problem, the six degrees of separation problem and the idea of a connected world. The fascination with the idea of a connected world has been heightened by the developments in Information and Communication Technologies (ICTs). The Internet provides an information and knowledge interconnection network among different individuals, diverse geographical locations and various people groups. The Internet has, in a sense, become a platform for the realization of a connected world and a global village, echoing Milgram's sentiments.

The reality is however far removed from the ideas of connectedness and ubiquity as is sometimes suggested [2]. The reality is that while there is a growing proliferation of ICT and a more pervasive integration of computing into the everyday lives of many people, there is at the same time a large number of people who are not citizens in the global *eVillage*, and who are not

participants in the Internet economy. So, while the Internet has bridged many barriers (e.g. geographic, economic, and communication), there are still people who are excluded from the global *eVillage*, hence the term *digital divide*. The issue of digital divide has received attention from governments, research institutions and non-governmental organizations (NGOs). Central to the discussion on the digital divide is the quest to understand the causes and the effects for a community being excluded from participation in the global information society and also, in the context of social development, to understand the ways in which ICT can be used to aid development. The former concern has been the focus of research in the field of ethnocomputing, while the latter has been at the core of ICT for Development (ICT4D) research. At the core of ethnocomputing is the acknowledgment that computing is not culture agnostic and that ICT cannot be distanced and separated from the ethnographic and anthropologic considerations relating to the context in which it was developed or in which it is being implemented [3].

In the context of ICT4D it is important that ICT solutions take into consideration the need for deep localization [4]. This should consider not only the linguistic and thematic flexibility of customizing application interfaces but also the cultural and ethnographic factors within a society. These are the factors that influence the values, aesthetics, usage patterns and metaphors, attitudes, and perceptions of a society and are inevitably and subtly present and carried through in every ICT solution [4]. These are also the factors that are inherent within the HCI domain, at a deeper level than user interfaces. For example, the context that has influenced today's computer usage (i.e. desktop with folders and files, office-suite) is one of offices and desks, and working with files, folders, and filing cabinets. These cultural influences can also be seen in how the reductionist view of knowledge and reality within the Western culture, in which ICT and computing have their roots [3], has influenced the ontological and epistemological underpinnings of knowledge based applications and systems [5].

The requirement for culture sensitive and ethnographically informed ICT solutions necessitates the investigation into the models and frameworks that would guide the implementation of ICT solutions. This is especially important in the light of the wide adoption and deployment of ICT as an enabler for community development. Governments, NGOs and research institutions are exploring the ICT4D landscape as one of the means towards development for the currently marginalized, digitally excluded third world communities. Most of the undertaken ICT4D interventions fall prey to technological determinism, which assumes that using technologies and solutions that have been successfully implemented in one cultural setting would achieve the same success in a different environment [6]. These models and frameworks would need to be intrinsically flexible to be informed by the social dynamics, the knowledge systems, values, ap-

preciations, ideologies, beliefs, and aesthetics of the community where they are implemented [3].

Great effort and work has gone into situating ICT4D solutions within the specific cultural contexts where they are implemented, and some of this has been along the lines of linguistic localization of Human Computer Interaction (HCI) components, provisioning of socially relevant services (e.g. eHealth, eLearning, and eGovernment), and contextualizing usage modalities [7, 8, 9] to the skills and abilities of the communities. The focus of this thesis is on the contextualization of ICT solutions from the Indigenous Knowledge (IK) point of view and on situating the applications and services developed within the knowledge systems of a community, such that they encapsulate not only the IK but also the social dynamics. The ethnographically informed, locally situated, IK-based service development framework and architectures would mitigate the information and knowledge marginalization by allowing for the deployed solutions to be relevant and familiar, easy to use, and with a high buy-in from the end-users.

1.1 Technical context

The information revolution, driven by the advancements in technology, has introduced new information usage paradigms and challenged the traditional notions of knowledge production and consumption. The Internet is a key component of this revolution and it facilitates a distributed hosting of heterogeneous information, accessible to a multiplicity of devices through various protocols and standards [10]. This paradigm shift can be observed in the proliferation of Web 2.0 tools, which are enabling and facilitating the authoring of information and production of knowledge by the average Internet user, versus the expert knowledge users or the large media houses. Statistics published in 2006 estimated the number of new blogs created every day at 12 000, and that approximately 400 000 blog posts were made every day [2]. Technologies and standards used in Web 2.0 applications are Atom/RSS, AJAX, Simple Object Access Protocol (SOAP), Friend of a Friend (FOAF) and eXtensible Markup Language (XML) (which typically underlies these technologies) [11]. Besides a technological revolution, Web 2.0 encouraged new paradigms and dynamics: a more user centered, decentralized approach to deploying services (BitTorrent); rich user interfaces (AJAX); user participation and a *trust your users* attitude (Wikis); community belonging, participation and sharing (FOSS, Social Networking); scalability, innovation and constant re-engineering (i.e. the notion of *the perpetual beta*).

Inherent in Tim Berners Lee's vision of the Internet is the idea of a ubiquitous knowledge repository that encapsulates not only the data but also the associated semantics [12]. The ad-

vancements in Knowledge Representation (KR) languages (e.g. Resource Description Framework (RDF), Web Ontology Language (OWL), DARPA Agent Markup Language + Ontology Inference Language (DAML + OIL)), knowledge processing heuristics and knowledge modeling tools and applications, have brought the vision of the Semantic Web (SW) closer to reality [13, 14]. These advancements in knowledge modeling and representation have heralded an era of intelligent, knowledge-based applications that are making their way into various application domains: medical applications [15], financial brokering systems [16], and airplane control systems [17]. Knowledge-based applications can exhibit human-like problem-solving capabilities for a particular domain. These applications depend on large knowledge repositories that encapsulate the expertise within a domain. The building of these knowledge repositories is often through a structured, top-down process of ontology design followed by ontology population or through an organic, evolutionary process of folksonomy development, also known as social or collaborative indexing [18]. The success of knowledge-based systems and SW technologies is evident from the wide scale adoption and successful deployment of semantic technologies based and social media based services: Flickr [19], del.icio.us [20], facebook [21], Orkut [22], MySpace [23], Amazon [24], and eBay [25].

Alongside the advancements in content representation, there have been new developments in web services deployment architectures. Web services are primarily distributed computation components that communicate via standard protocols and specifications. Web service solutions and architectures have evolved from Electronic Data Interchange (EDI), Common Object Request Broker Architecture (CORBA), Remote Procedure Call (RPC), Java Remote Method Invocation (JRMI), Web Services Architectures (WSA) to Semantic Web Services Architecture (SWSA), which is a web services architecture that seeks to enable maximum automation in services discovery, selection, negotiation, invocation, monitoring and recovery [26]. The automation (intelligence) in web services is achieved through the use of semantic knowledge representation. As a result, languages including DAML-S and OWL-S have come to the fore [27, 28]. These developments are aligned with the vision of a semantic Internet, a distributed, networked repository of machine processable knowledge.

These technical developments are paving a way for the realization of the Internet of the future. A prediction by Gartner is that by 2011, 63% of products in the software infrastructure, and 56% in the software application market will support web services and Web 2.0 technologies [29]. These are the technologies that are encapsulated in the ”version” of the Internet referred to as Web 3.0.

1.2 The problem areas

The benefits of all these advancements in ICT (i.e. knowledge representation tools and languages, service deployment architectures, Web 2.0 and Web 3.0) are not experienced globally. As mentioned before, the main reason for this limitation is a disparity in societies, usually referred to as the digital divide, which leads to information marginalization and participatory exclusion of individuals and societies from the Internet driven knowledge economy [6]. This marginalization has direct ramifications to the societal well-being of communities due to the direct correlation between access to information and the access to economic opportunities, a concern raised by the United Nation's Administrative Committee on Coordination (UNACC) [30]. The digital divide is caused, among other things, by the cultural and ethnographic mis-alignment of the deployed ICT solutions with the societies in which these solutions are being implemented. ICT solutions and computing should therefore encapsulate the cultural considerations of the local society [3].

The need to adapt and localize computing solutions is one that has been addressed extensively in the discipline of HCI, with solutions predominantly adapting the linguistic, thematic and stylistic features of computer interfaces to the locality where the solution is being deployed. This addresses only a few of the factors in a cultural make-up of a society. It does not address, for example, the values of a society as far as their epistemological and ontological perspectives are concerned, nor the relevance of the knowledge encoded in the solution. It is therefore necessary that tools and mechanisms are developed to enable a deep level HCI intervention (versus user interface level adaptation) of technology in the local culture, context and environment.

Another ramification of the digital divide is the perpetual innovation lag that the marginalized third world communities experience behind the technologically developed communities. This phenomenon is sometimes termed *technology dumping* and is usually used in the context of old (and sometimes obsolete) computing hardware being transferred to poor communities to start them off in using the technology [31]. This process automatically sets these communities off a few years (and therefore older technologies) behind their technologically advanced counterparts. There is therefore the need for the solutions and technologies that are deployed in ICT4D context to integrate and encapsulate the new developments in computing with the view of being forward compatible with the technology of the future. This can be achieved by ICT4D interventions that are guided by the global trends in technology (as mentioned in Section 1.1) and also by activating and empowering (technologically and otherwise) marginalized communities to be participants in the evolution and in the shaping of the Internet. A trend towards this process of activation and empowerment has already been experienced through Web 2.0 tools and the proliferation of more user-centric, user-driven and user-controlled service deployment architectures. Web 2.0 tools

shifted the traditional information exchange paradigm from the users being just the consumers of information, to enabling the users to be both authors and consumers of information. This makes the Internet more relevant. This level of relevance needs to go beyond just content, to modeling and reflecting local knowledge system dynamics and internal system relationships.

This thesis explores these problems in detail and addresses the challenges inherent in these problems, with a view to providing part of the solution that indeed needs to be multi-disciplinary, multi-angled and multi-faceted.

1.3 Objectives

The over-arching objective of this research is the investigation of an architectural framework for context-sensitive, ethnocentric, and knowledge based software applications that are deployed in a heterogeneous context (typified by the ICT4D context). This is with a long-term view to the establishment of a knowledge society in rural, marginalized communities by activating them to participate in the global knowledge economy. The following is a detailed breakdown of the research objectives:

1. Objective 1 - *Investigate and formalize an architectural framework for software that is sensitive to the socio-technical context of deployment*: positioned within the social dynamics of the community; flexible to operate in an environment characterized by multiplicity of end-user devices, diversity of users (different preferences and capabilities), and heterogeneity of interaction modalities; and encapsulating the local knowledge in the community.

2. Objective 2 - *Explore the knowledge system dynamics and the social context in marginalized rural communities,* taking into consideration the nature of their IK (by exploring current knowledge repositories, current traditional knowledge systems, and accessibility of the knowledge) and the usage characteristics and patterns associated with the IK.

3. Objective 3 - *Develop a proof-of-concept knowledge platform based on the architecture.* The platform should handle heterogeneity as far as the following are concerned: sources of the information, format of the content, usage modalities and access methods to the information. This also entails deriving relevant schematic encoding for the ontologies that encapsulate the local knowledge.

4. Objective 4 - *Undertake a pre-validation of the architecture and the developed platform* from the point of view of being sensitive to the context of deployment, and functionally and technically adequate.

5. Objective 5 - *Develop tools and services* that use the implemented knowledge platform. These tools will be in support of community eServices within the context of an ICT4D intervention.

1.4 Research approach

This research in undertaken within a context of an ICT4D project that has been running for the last three years [32]. The project site is a rural marginalized community in South Africa. Both the ICT4D project and the community provided the context for this research. Based on our experiences in the project and on the exploration of literature in the ICT4D domain, we identified the limitations and challenges associated with current ICT4D interventions. The overall limitation of lack of context-sensitivity and ethnocentricity in the developed solutions was then developed as a core research problem which is investigated in this research.

At the conceptual level we explored the formalization of ICT4D interventions primarily as knowledge networking solutions (i.e. that the ICT solutions developed for community development should primarily facilitate the processing of the IK within the community and the integration into the global knowledge systems). The integration of IK is also, and almost automatically, one way of contextualizing ICT solutions to the environment of deployment. Other approaches for contextualizing software (e.g. user interface customization and localization) have been proposed and implemented in different projects [33]. Traditionally most of these approaches have been undertaken in a disconnected manner (i.e. as an added feature) from the rest of the software development. This research is focused on investigating the viability of an architectural framework that specifically addresses the need for context-sensitivity in applications (particularly within the ICT4D domain).

In exploring the problems and viability of the framework discussed above, the following activities were undertaken:

1. Explored the literature within the ICT4D domain to understand the limitations and challenges in the current ICT4D interventions.

2. Investigated the knowledge systems dynamics in rural communities and the requirements as far as utilization of knowledge in these communities is concerned.

3. Conceptualized and formalized a new architecture to address the need for context-sensitivity and ethnocentricity (with a particular focus on rural marginalized communities).

4. Undertook the evaluation and validation of the proposed solutions.

1.5 Significance of research

The work undertaken in this research has direct impact within the ICT4D domain and the following are the motivating issues under consideration:

1. ICT and connectivity to the Internet have the potential to foster development within communities. Effective adoption of the Internet can be realized through: the provisioning of semantic capabilities by encoding the information to be meaningful, both to the end users and to the information processing agents; the encapsulation and integration of local IK; and the implementation of context-sensitive services that provide direct benefits to the communities in which they are deployed. The investigation of a framework towards greater context-sensitivity in software applications paves a way for realizing more effective ICT4D interventions.

2. The ability to process and distribute information effectively via the Internet has heightened the pervasiveness of knowledge in the economy. Knowledge has always played a crucial role in economies, however, in the knowledge economy it takes a more central role where ”the principal component of value creation, productivity and economic growth is knowledge” [34]. Implementing and deploying ICT based knowledge systems and applications for marginalized communities activates them for participation in the knowledge society.

3. The Internet is envisioned as a network of distributed and ubiquitous knowledge that is not encapsulated in the domain expert applications but rather ingrained in the machine-processable meta-data via suitable ontologies [12]. The lack of these underlying knowledge repositories for marginalized communities essentially means that the applications and systems that are developed are not able to use and exploit local IK to provide relevant contextualized systems. This is a problem that might not be too costly currently but in view of the evolution of the Internet (i.e. Web 2.0, Web 3.0 - SW) might be a huge limitation and a further cause of marginalization at a later stage. Developing local ontologies for these communities mitigates against this problem (current and future).

4. Adapting and situating computing systems and applications within the cultural and environmental context of deployment is one of the goals within HCI research. Much work has been undertaken with a focus on the user interface aspect of HCI, and this has been in

terms of the themes, styles, and the interface language of the developed applications. This research explores the contextualization of ICT4D systems at a deeper social and cultural level and in so doing makes a contribution to the HCI domain.

1.6 Proposed framework

The solution proposed in this thesis provides an architectural framework to facilitate the implementation of knowledge-based systems in marginalized rural communities. The overall framework, named PIASK, is a service deployment architecture that addresses the requirements and the problems mentioned in Section 1.2. PIASK addresses the need for the knowledge available on the Internet to be locally relevant by using ontologies for specific local domains (e.g. health and medicine, agriculture). It addresses the need to model the community knowledge system dynamics by implementing an explicit social networking layer in the architecture. PIASK makes a provision for the utilization of the technologies and tools that are prevalent and that are likely to form and remain at the core of what the Internet evolves into. These are the SW tools (Web 3.0), Web 2.0, knowledge representation and authoring technologies, multimedia tools and specifications to handle multiplicity and heterogeneity in the deployed ICT systems.

The PIASK architecture is realized and validated through the implementation of KnowNet[1], a knowledge platform which has been developed as a Multi-Agent System (MAS). At each of the different layers of the architecture, agents collaborate to provide the functionality defined for that layer. KnowNet enables the deployment of knowledge rich eServices and exposes the available agent behaviours through standard Web Services (WS) technologies. The role of the platform is to handle community users' knowledge requirements and the provisioning of services for WS clients (i.e. software entities).

1.7 Research contributions

This thesis is primarily positioned within the ICT4D domain, but it also addresses issues that pertain to software development, HCI and ethnocomputing. The main conceptual contributions made in this thesis are:

1. *A knowledge-centric paradigm for ICT4D interventions* (Section 5.1)- undertaking an ICT4D intervention within a paradigm that emphasizes the centrality of the local knowl-

[1]The realization of PIASK as a knowledge platform is called KnowNet. However, other references used in the thesis include: the knowledge platform, PIASK platform or the KnowNet platform.

edge and facilitates the processing and exchange of the knowledge. This is an additional and a different approach to the traditional paradigms (i.e. technological, cultural, economic and structural) (Section 2.1.2).

2. *Web 3.0 for the third world contexts* (Chapter 4) - exploring the utilization of latest, high-end Web 3.0 (i.e. SW) technologies for the ”low-end” third world ICT4D contexts. This challenges the traditional *technology dumping* attitude that is often found in ICT4D interventions.

3. *An architectural framework towards context-sensitive and ethnocentric ICT solutions* (Chapter 5) - investigating the viability of a software architecture for applications that are inherently flexible and sensitive to the environment of deployment, with a strong community orientation. This builds on the principles within HCI and ethnocomputing to provide highly contextualized and tightly situated applications within the socio-technical context of deployment.

4. *A formalization of knowledge system dynamics in marginalized rural communities* (Section 4.5) - allowing for a profiling of different communities and their associated IK along the dimensions of ownership, social advantage, confidentiality and accessibility.

These and other technical contributions are further discussed in Section 9.3.

1.8 Thesis structure

In this chapter we have provided an overview of the problem areas that are addressed in this research, highlighting the key aspects of the solution that has been investigated and the contribution that the research makes. In Chapter 2, we explore the area of ICT4D with a specific focus on the role of knowledge in these interventions. This is followed, in Chapter 3, by a discussion of the current technological developments and architectures for knowledge based application development. Chapter 4 contextualizes this research, by giving an overview of the community that forms the field site for the research. We discuss the characteristics of this community that have a bearing on the development of knowledge based ICT4D solutions. This includes the knowledge systems dynamics and the infrastructural constraints and challenges within this community. This chapter crystallizes and formulates the key problems and the requirements that inform the development of the PIASK architecture and the KnowNet platform. Chapter 5 elaborates on the conceptualization of the PIASK architecture. The design and development of the KnowNet

platform as a MAS, specifying the contributions that have been realized in its development, is discussed in Chapter 6. This is followed directly by the validation of the platform for technical and functional adequacy in Chapter 7 and the evaluation of the social context adequacy of both the PIASK architecture and the KnowNet platform in Chapter 8. The thesis concludes in Chapter 9 with an overall discussion of the research: addressing the research objectives, the contributions made, and the future research directions.

Chapter 2

Knowledge and Development

> ”Once a new technology rolls over you, if you are not part of the steam roller, you are part of the road.” (Steward Brand)

The work undertaken in this research builds on extensive prior knowledge in the area of ICT4D. This chapter provides the contextual under-girding for this dissertation by exploring the key theoretical developments around formalizing the problem of underdevelopment, and the role of ICT towards community development. The key role of ICT in development is the facilitation of exchange and usage of information and knowledge, and therefore a large portion of the discussion is devoted to the consideration of knowledge in communities: the nature of knowledge from a multi-discipline perspective, including the consideration of the sociological theories and the philosophical underpinnings of knowledge; indigenous knowledge and its role in the life of communities; and the role of ICT in the knowledge economy and some of the paradigm shifts associated with the knowledge revolution. This chapter positions the research within the context of related work that has previously been done and the work that is currently being undertaken.

2.1 ICT for community development

ICT4D explores the ways in which ICT can be used in the context of community development. In addressing the role of ICT in development, it is imperative that the underlying theories and conceptualization of both the nature of underdevelopment and the nature of the interaction of ICT solutions in a community are understood and articulated.

2.1.1 [Under] Development

Inherent in the idea of social development is the notion of a need for improvement and embetterment of the conditions of life within a community. Social development used to be equated to social survival, but over the last number of year it has become equated with improving the living standards of people [35]. Social development has therefore taken a multi-dimensional approach to the improvement of life. It considers the meeting of people's needs along different dimensions: heath, food, shelter, security, dignity and significance needs. Social development in this day and age is determined by the ability to create positive interactions between the technological developments and human values, resulting in organizations and communities that "create feedback loops between productivity, flexibility, solidarity, safety, participation and accountability" [35].

2.1.1.1 Rurality and underdevelopment

Traditionally the bulk of economic development and the hubs of industrial activities have been the urban cities where industries have thrived and skilled labour has migrated. This shift in economic dynamics has meant that urbanization and development have been minimal in rural communities with the net effect that rural communities become the least developed. The resultant socio-economic dynamic that occurs in these communities is termed the rurality penalty. One of the primary dimensions of the rurality penalty is the low population density, and therefore low density of most markets. In turn, this results in a great distance to the markets and to information, knowledge and resources [36].

Drabenstott identifies 5 critical challenges in shaping the rural economic outlook: tapping digital technology, encouraging entrepreneurs, leveraging new agriculture, improving human capital and sustaining the rural environment [37]. Success of economies, especially in this digital era, depends on the role of entrepreneurs in these rural communities, and therefore it becomes a human capital issue and a social capital issue. Traditionally entrepreneurship in rural communities was hindered by low levels of education in these regions and the migration of skilled human labour to the large urban cities. These issues around entrepreneurship in rural communities are complicated by a number of phenomena: the return migration of skilled locals, the growing appeal of rural communities for young retirees, and the general structural support for entrepreneurship in these rural communities [38].

The adoption of ICT in rural communities is a phenomenon with pronounced variability and one which is still not thoroughly understood [39]. It is therefore still not very clear why some rural firms adopt ICT and why others are resistant to the integration of ICT into their operations.

However, research has found that some of the leading reasons for rural communities adopting ICT are more social than economic. These rural communities adopt ICTs due to the ease of access to information and ease of communication that they provide. As such, the widely used applications are email applications and Internet search engines [38].

It is within this landscape of rurality that ICT interventions are explored to facilitate development and enhancement of life in rural communities. This environment of rurality presents unique challenges and opportunities which are explored in detail later in this chapter.

2.1.1.2 Sustainable development

The question of the sustainability of development projects is a concern that arose primarily out of evidence of the resultant environmental degradation and the persistence, or worsening, of poverty in areas where such projects were implemented [40]. Subsequent efforts were undertaken by the United Nation (UN) under the auspices of the Brundtland Commission (formerly the World Commission on Environment and Development (WCED)) to address the issues of ecological and cultural sustainability in development projects. The idea of sustainability has since moved from a focus on ecological consequences of development projects to encompassing a sustainability of wider aspects within the socio-political systems of the communities in which the projects are implemented. Gibson *et al* mention the following ten basics that under-gird the full definition and consideration of sustainability [40]: current paths of development are not sustainable; sustainability is about projection and creation; requirements for sustainability are multiple and interconnected; pursuit of sustainability hinges on integration; core requirements and general rules must be accompanied by context specific elaborations; diversity is necessary; surprise is inevitable; transparency and public engagement are key characteristics of decision making for sustainability; explicit rules and processes are needed for decisions about trade-offs and compromises; and the end is open.

Sustainability is a concept that relates to the continuity of economic, social, ecological, cultural, political and structural aspects of a society. It connotes the ability of a society to meets its needs and achieve its greatest potential while planning and acting in a manner that allows it to maintain the ideal for a very long time. According to the Brundtland report [41], sustainable development is defined as:

> "...development that meets the needs of the present without compromising the ability of future generations to meet their own needs."

Implicit in this definition is the awareness of the environmental constraints imposed by the im-

plemented development projects. These are the constraints which include the consideration of ecological factors, the economic viability of the projects and the preservation of the cultural integrity of the communities in which the projects are implemented. The issue of sustainability is one that is context (i.e. time and space) sensitive, and as such the formalization of generic best practices is elusive [40]. There are, however, underlying common characteristics which are exhibited by a sustainable system: Learning, Innovation, Adaptation, Flexibility, and Anticipation [40].

Hietanen [42] defines four dimensions of sustainability whose borderline limits should not be crossed and which should be combined in proportional measures. In other words, a business focused development project that increases income and profit to the entrepreneurs at the expense of violating and exceeding the limits of a community's values and beliefs, would not be sustainable. This notion of balancing the outworking of a solution within its different dimensions is equitable to the *principle of shalom* (discussed in Chapter 8) from Dooyeweerd's philosophical framework of modal aspects[43]. These dimensions that need to be balanced are:

1. Ecological - development interventions and ICT solutions should lead to an overall eco-efficiency within a society.

2. Social - the well-being of communities along the aspects of work, health, safely, regionality, communality, and social well-being should be maintained and strived for in the undertaking of sustainable solutions.

3. Cultural - the cultural dimension of sustainability is the sum of the human, cognitive and spiritual aspects of a society as realized within the 15 modal aspects in Dooyeweerd's philosophical framework [43]. The values, aesthetics, practices, beliefs of a society should be integrated and a proportional balance achieved in the implementation of development interventions [43, 44].

4. Economic - balancing the transition towards the knowledge economy and the digital economy, keeping the profitability motive at the fore.

The diversity of the different environments in which projects are undertaken necessitates a contextualization of these dimensions that have been highlighted to the specific situations of the communities. Different models of sustainable development have been proposed that take into consideration different environment variables within a community: The economist capital model [45]; the Triple Bottom Line model (a.k.a. People-Profit-Planet model) that considers the balancing of social performance, economic profitability and environmental well-being [46]; and the

PRISM model, based on the Brutlandt definition of sustainability, that defines four imperatives for sustainable development (i.e. institutional imperative, economic imperative, environmental imperative and social imperative) [47]. These principles, guidelines, models and patterns towards sustainability can and should inform the decision making in undertaken ICT4D projects.

2.1.2 Paradigms for formalizing development

Poverty and underdevelopment are a complex phenomenon that is articulated and formalized in numerous approaches [48]. The articulation of the problem has direct bearing on the solutions that are proposed. In a number of ways the problem of underdevelopment is akin to the story of the six blind men and the elephant, who in their own right have a conceptualization of a part of the whole [49]. In a similar manner, different paradigms of formalizing poverty target an aspect of underdevelopment which is part of the whole problem, and necessarily not the totality of the problem. The following are some of the paradigms within which poverty can be articulated [6]:

1. Technological - The premise for this view is the correlation, observable in the first world countries, between the use of technological solutions in their primary industries and the level of development of the countries. Technology is seen deterministically as the overall solution that permeates all areas of society. The majority of current ICT solutions for development are based on this premise of transferring the technology from first world countries to the developing countries. As evidenced in technology based development projects that have failed, a technology determinism that is not accompanied by a corresponding societal empowerment in the form of skills transfer is not adequate for mitigating the problem of underdevelopment. Reports are published of agricultural equipment that gathers rust and never gets utilized, or of ICT tele-centers that are never open to the public for efficient utilization.

2. Economic - In this view, poverty is articulated in monetary terms. Economic metrics such as Gross Domestic Product (GDP), Balance Of Payment (BOP), and GDP per capital are used as measurements of poverty. The role of the international financial institutions (e.g. the world bank and the International Monetary Fund (IMF)) become paramount and emphasized in the alleviation of poverty.

3. Cultural - In this view poverty is seen as a result of a society's culture and the social values that a community holds. Culture is the learned behaviour that is prevalent within that society and hence under-girds the society (i.e. it influences the nature of social institutions, social activities, role players in the community, society's attitudes towards change

and progress). This view identifies the fact that ideas have consequences, that actions are a result of an interplay of different ideas, thoughts, opinions and influences, and subsequently that the *right* values and worldviews can lead to favourable behaviour that alleviates poverty.

4. Structural - The political and social systems within a community are also highlighted as the main causes of poverty within the structural paradigm analysis of poverty. This view places the responsibility of addressing poverty in the hands of the government, as the institution of social development. The traditional power-play by the society's elite and rulers and the resultant increased poverty and dependency of the masses highlight the legitimate concerns in this view.

Most ICT4D interventions subscribe to a particular paradigm (i.e. a technological, economic, structural or cultural paradigm), or a mix of the different perspectives [6]. An intervention undertaken within an economic paradigm would predominantly tend to develop eCommerce solutions. A project undertaken within a structural paradigm would focus on eGovernment solutions within the paradigm, and eLearning and media-based solutions would be proposed. It is important that the adoption of ICT4D interventions embrace a holistic, flexible, and adaptable approach in aligning their development efforts with a particular paradigm. Within each of the different paradigms, ICT is significant in the realization of the development goals. In the next section we articulate the role of ICT within the community development landscape.

2.1.3 Role of ICT in development

ICT indeed provides the opportunities for dealing with rural poverty, inequality and degradation and in many ways it is challenging the traditional paradigms of doing business, delivering services to citizens and running societal institutions [48]. The key question at the center of all policy makers and governments is whether to invest in developing ICT capacities in the rural communities or to focus on the provision of other basic services (e.g. schools, hospitals, and government services). At the core of this question is the need to understand the role that ICT can play in the development of a society, and the benefits and limitations that can be expected from undertaking an ICT based approach to development.

Technology, in and of itself, is not a panacea for the underdevelopment woes of communities. It is, however, a prerequisite for social development in this day and age [35]. Neither is technology a target for community development and social well-being, but rather a tool for facilitating the achievement of a desirable future for a society: well-being, health, peace, and communality

[44]. To a large extent, human activity depends on information and therefore a synergistic interaction of technology and information leads to a competitive advantage for societies [35]. ICT also increases information share-ability within communities and therefore can positively impact the provision of that information for commercial benefit, based on the differing valuation of the information to different people [50]. The application of ICT in development can be broadly categorized into the following[48]: decision support systems for administrators; improving service delivery to the citizens; and empowering citizens with access to information and knowledge.

There is a cyclic dependency between demand for technology and the advancements in technology, in that a strong demand for advanced technology produces innovation and higher levels of service, which in turn leads to more demand [38]. Investment in infrastructure is therefore the initial step in the realization of effective ICT for development. This has to be accompanied by services and applications that meet the needs of the communities in which the ICT solutions are implemented. The communities themselves must adopt and use the implemented technology and translate that into a competitive advantage within the context of their economies and their development environments [51]. While there is a complex relationship of cause and effect resulting in direct and indirect costs and benefits, ICT can be expected to play a number of different roles in sustainable community development [52]:

1. Direct technology benefits due to increased efficiencies
2. Indirect benefits through changes in behaviour of individuals and organizations
3. Improving the overall decision-making capabilities of a society

An extensive study was undertaken by Wellman *et al.* to determine the effect of the Internet on the social capital of communities along the dimensions of network capital, participatory capital and community commitment [53]. The conclusion from the study is that Internet supplements the network capital, increases the participatory capital and decreases the commitment to online community. These effects can therefore be harnessed within the context of development to achieve a desired outcome in a community.

As far as the economic development of a community is concerned, there are two potential gains from implementation of ICT based solutions. The first gain is with regard to static and dynamic efficiencies [50]. Static efficiency gains are once off gains associated with efficient use of resources that results in higher consumption. Dynamic gains happen as a result of higher growth in the economy of the region, which leads to an overall increase in future demand and consumption within that community. The second gain associated with ICT is as a result of a

reduction in the economic inequality within a community to the extent that it is a social benefit within it [50].

The dynamics associated with ICT in development are therefore twofold: on one side it is able to leap-frog community development based on the ability of the society to synergistically embrace the technology (e.g. the Asian economies: Hong Kong, Taiwan, and Singapore); and on the other side, ICT can further the marginalization, participatory exclusion in global economies and retardation of communities that are not able to integrate technology into their life-systems [35].

There are potential benefits of ICT to facilitate social development, but the biggest challenges to development are those associated with shortages of human capital in these under-developed communities [38]. ICT is therefore a necessary but not a sufficient condition for community development [54].

2.2 Knowledge marginalization and the digital divide

There runs within societies a demarcation line between the *haves* and the *have-nots* (i.e. the different class structures and class hierarchies). The ramifications of these societal classes are prevalent in all spheres of society: education, governance, and social activities. This delineation in societies introduces a paradoxical consideration as far as information and knowledge are concerned. Feather articulates this paradox as follows [55]:

> "technology has made more information more available to more people than at any time in tens of thousands of years of human history. But the same technology has made access to it more difficult"

Access to information and the contribution to information creation has been a privilege reserved for the *haves* of societies: those who have the appropriate education qualifications, financial means, or social standing. This gap, as far as information and technology is concerned (i.e. the digital divide) is both the cause and the effect of social delineation, social class hierarchies and marginalization. With regard to the effects of technology and globalization, Castells defines the following relationships of consumption: inequality - unequal and mis-apportionment of resources; polarization - the marginally faster growth of the top and bottom sectors of a society than the middle sector; poverty - a level of living below the socially accepted well-being standard; and misery - the lowest standard of living as per defined levels of well-being [35]. Castells then observes the following social trends in the world [35]:

1. There is an increasing inequality between countries worldwide.

2. Polarization is increasing globally. The ratio of the income to top 20% to the bottom 20% in the world increased from 30:1 in 1960, to 78:1 in 1994, and personal assets of the top 385 richest people in the world is higher than the annual income of countries that represent 45% of the world population.

3. The number of people living in poverty has increased across the world.

4. People leaving in misery are some of the fastest growing groups of poor people in the world.

The causes of this marginalization and digital divide have been the subject of numerous discussions, and in reality these are as numerous and diverse as the associated communities and environments.

2.2.1 Connectivity, affordability and capability (CAC)

Traditionally, information marginalization and the widening digital divide were understood to be a result of three factors:

1. Connectivity - public service delivery is typically very poor in rural marginalized communities, and as a result there are numerous infrastructural constraints. Basic connectivity to wired telecommunication networks is often non-existent in these communities [56]. A number of ICT4D projects have therefore predominantly focused on addressing this factor. Some projects have implemented wireless networking solutions: Very Small Aperture Terminal (VSAT), WiFi, WiMAX, and wireless broadband technologies [32]. One other infrastructural barrier to ICT proliferation in rural communities is lack of electricity, and this has sometimes motivated the use of solar energy as an alternative power source [57].

2. Affordability - the costs associated with acquisition of hardware for Internet provision are sometimes prohibitive for poor communities, and as a result become a barrier to access to ICT solutions. There are projects that address the barrier posed by the costs of technological equipment, by developing low-cost, entry-level devices for deployment in rural areas. One Laptop Per Child (OLPC) [58] and Aleutia [59] are examples of such projects. Other projects minimize the costs associated with acquisition and use of software by adopting Free and Open Source Software (FOSS) solutions [60].

3. Capability - once connectivity and the associated costs are provided, there still remains a limitation of the capability of the communities to use the technology. This is directly associated with the skills of the rural communities and their literacy levels. Therefore there are ICT4D projects that focus on addressing this barrier by undertaking training and skills development for rural communities [61, 62].

Besides implementations that address these specific barriers to ICT access, some ICT4D projects provide *ad hoc* solutions to specific social institutions: eLearning solutions for education institutions and social centers [8, 9]; eHealth for clinics [63, 64, 65]; and eCommerce for entrepreneurs [66, 67].

2.2.2 Digital barriers, and enablers

The digital divide also has its roots in the deeper societal systems and structures. There are factors which contribute to the propagation of the digital gap and those that contribute to minimizing it. These are considered below:

2.2.2.1 Access to information

1. Cost - the commoditization of information has meant that access to information has an associated cost to it (i.e. the cost of buying a newspaper, television, computer, and cell phone). These costs associated with acquiring information are sometimes prohibitive for the very people (i.e. the socially marginalized and the poor) to whom information and access to it might have far-reaching benefits.

2. Usability - the skills and abilities necessary to engage in information acquisition are not always commonplace. This manifests in terms of technical limitations (i.e. inability to use/operate a television set, or a computer) and also in terms of Human Computer Interaction (HCI) interfaces that pose limitations to the user (i.t.o. usability, intuitiveness, and aesthetics).

2.2.2.2 Cultural boundaries

1. Ethnographic considerations - these are the engraved ethnographic tenets of a culture or a society that clash or conflict with specific factors in the implemented information systems (i.e. clash of worldviews, clash of cultures). A example that is regularly cited is that of the office metaphor that is predominantly adopted on computing systems.

2.2.2.3 Structural limitations

1. *Lingua Franca* - English has been the *lingua franca* of the information age. The inability to communicate in English traditionally meant an inability to access the tons of information encoded in the English language, or the tools that used English as the interface language.

2. *A voice* - There are underlying structural dynamics that entail that some people have more of *a voice* than other people, in terms of being heard and/or having an opportunity to articulate information. These dynamics are associated with educational credentials, individual charisma, social class structures, financial status or social status.

2.2.2.4 Digital enablers

The digital landscape has been wrought with interplays between forces that increase the digital divide and the forces that decrease it. Some of the factors that facilitate the decrease are:

1. Empowerment - as more individuals are empowered with necessary skills and abilities, they are able to engage and use the information processing tools and related technologies. One of the avenues for empowerment of communities is education and practical training in the use of ICT.

2. Technological innovations

   (a) Devices - the innovations as far as devices are concerned have introduced smaller, more powerful devices that offer the required capabilities to handle information processing requirements of an average person. Mobile telephones are now manufactured with capabilities to access information on the Internet, to communicate information and to do minimal storage of information on the device. These innovations as far as devices are concerned are also reshaping information processing and distribution paradigms that were initiated in the print media era.

   (b) Standards - parallel innovations are happening as far as standards, protocols, frameworks, and methodologies related to information processing are concerned. These are enabling wider access to information and facilitating the societal engagement through various diverse avenues (e.g. Wireless Application Protocol (WAP) or Hyper Text Transfer Protocol (HTTP) to access a hypertext resource).

3. Ubuntu - there is more alignment of efforts towards the attainment of the common good for all. The realization of the injustices of poverty and underdevelopment are driving efforts

towards their alleviation and the empowerment of individuals. The *Ubuntu* philosophy encourages the collectivist ideology where individuals choose actions that have a greater communal good than the individualistic ideologies.

4. Cost reductions

   (a) CAPEX - the cost of capital investment associated with information processing has always been one of the aspects that limits participation (e.g. printing machines, and expensive distribution networks) [50]. The information revolution associated with computing has introduced drastic reductions in the related hardware costs. The costs of computers are decreasing by the day thus enabling marginally more people to participate in the information landscape.

   (b) OPEX (Software) - some other aspects that limit access to information participation are related to the running costs of processing information. The FOSS movement has brought paramount cost savings related to the use of relevant software to facilitate the operation of information processing hardware.

2.2.3 The paradox of restricted and limited interventions

One of the limitations in ICT4D projects is a restricted and decontextualised strategy for Internet connectivity. Such interventions may work well within the constraints of their implementation, but limit the participation in the global eSociety for the target communities. This perpetuates information marginalization and exclusion from the full advantages of the Internet. To illustrate the point: a project could be undertaken that seeks to provide ICT solutions for rural health systems; for an example refer to [64]. In the deployment of such a project a LAN between the clinics in the community and a back-haul link to the supporting hospitals would typically be deployed. The clinics would then have a communication channel between themselves and also with the supporting hospitals. Such interventions do not take into consideration the possibility of using the deployed LAN to support other societal activities, and remain constrained within the health domain. This constrained and tightly focused intervention means that the nature of the deployed infrastructure is tied to the usage context envisaged by those who deployed it. This would limit its use in different usage contexts, thus limiting its potential. So in the community health system scenario, a low bandwidth network, sufficient within the health sector could be deployed, which in turn would not be sufficient to support a community entertainment network, or a telecommunication grid. Typically such an intervention would result in individuals (e.g.

nurses and health workers) within the community who are vested in the knowledge of using the health system, but not in a community that is generally eLiterate and that can be involved as peers and participants in the global eSociety.

2.3 Ethnocentric ICT for development

One of the primary tenets of ethnocomputing is the realization of the culture specific influences on computing and subsequently on the Internet [3]. The Internet was born within a specific worldview, culture and social system, and over the years the interaction of different individuals and groups on the Internet has resulted in the synthesis of a new culture, an eCulture. To highlight one such influence of culture on computing, the current metaphor of desktop, folders, files, and documents to refer to the organization of bits and bytes on a computer is one that was birthed in a context and domain of office usage, where files and documents are organized in folders, and the work involves the use of a desk. This is also evident in that the predominant positivist notion in computing is a Western conceptualization of reality, and this imposes an assumed dominance of science and ICT at the expense of various other socially ingrained realities [68]. At the same time there are other areas which have been birthed from other cultures: for example, fuzzy logic has been predominantly formulated and contributed to from the Eastern cultures which have more tolerance of truth values that are neither totally true nor totally false [69]. Suffice to say, the ethnographic influences from different cultural contexts are evident in computing, and the most effective usage of those tools and constructs would be within the context of communities that have a similarly aligned cultural orientation.

It is important at this stage to delineate the different levels of granularity and heterogeneity that come into play in the consideration of differences and variations in the realization of computing tools and technologies. Tedre *et al.* define four different levels of computing which are realized differently within different cultures [68]: the structures and models used to represent information (i.e. data structures); the ways of manipulating the information (i.e. algorithms); the mechanic and linguistic ways of realizing the data structures and algorithms; and lastly the application of the above three levels. Tedre *et al* also identify three levels of uniqueness in the human enterprise, depicted in Figure 2.1 [68].

At the lowest of the levels is the general human nature that is common to all human beings; the influences that emanate from this level are universal and are appreciated globally. The next level is the cultural level of uniqueness, which encapsulates values, ideas and preferences that have been learned over time and that are part of an identity of a specific group or category of

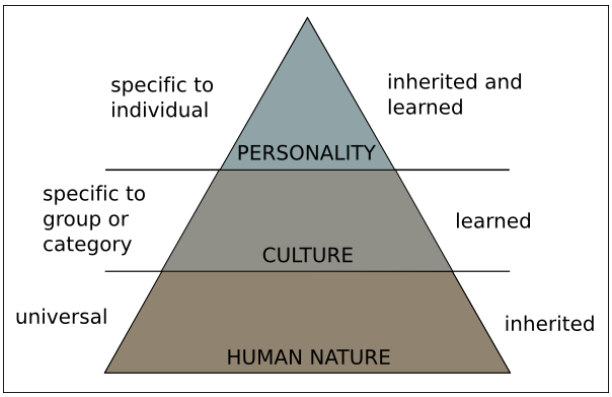


Figure 2.1: Levels of human behaviours [68]

people. The topmost level, and the smallest, is the personality level. At this level, the variation is as large as there are people on earth, and it is the level at which the individuality of human being is expressed in terms of values, beliefs, and preferences.

In the consideration of ICT in development, the focus is at the level of a culture, the middle level in Figure 2.1. The aspects of computing that are universal and aligned with human nature are easily transplanted from one community to another. Aspects that are cultural, however, need to be considered, adapted and validated for adequacy within the environment where they are being implemented [4]. ICT4D should therefore take into consideration the ethnographic considerations and expressions of a community to avoid the technology determinism flaw that has plagued many ICT4D projects [5].

The realization of an ethnographic ICT for development intervention must address the issues at the interface between technology and the culture of a community. More appropriately, it has to address the encoded cultural expressions in the technology and their interaction with the culture of the community in which the intervention is being undertaken. This intervention strategy is positioned within the premise that for rural communities to be active participants and peers in the global eSociety, adaptation mechanisms and interfacing strategies must be implemented. These adaptation mechanisms facilitate the interaction between the rural communities and the Internet in a manner that is immediately relevant and positioned within the cultural and cognitive framework of the rural, marginalized communities (Fig. 2.2) [4].

The nature of the adaptation is bi-directional in that certain aspects of the Internet are adapted and situated within the usage framework of the rural communities, and inversely there is culture mobility within the rural communities due to social and anthropological influences from the Internet. The following are some of the under currents in this intervention strategy:

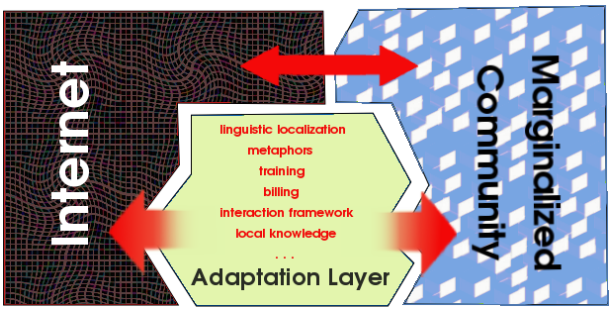


Figure 2.2: Adaptation of the Internet

1. Consumer/Producer Actualization - the nature of the interaction on the Internet is such that individuals play the role of both consumers and producers of knowledge. The majority of Internet users are traditionally consumers, but there is a move towards more people producing knowledge, through Web 2.0 tools.

2. Openness - the Internet is open and accessible to anyone with connectivity.

3. Distributed - the Internet is by nature distributed and the full advantages of the Internet are realizable owing to its being decentralized.

4. Diversified - the Internet fosters an environment for diversity that extends not only to its protocols and standards but also to the demographic make-up of its users.

5. Ethnocentric computing - cultural framework and background extensively influence the nature of the interaction and the experience on the Internet.

6. Dynamic discovery and appropriation of technology - once technology is deployed, it is used based on the user's needs and the value they derive from it. New uses of the technology are also discovered within the context of its being available.

This adaptation is specific to each rural community in terms of the nature of the components that make up the interface. The level of specificity in the implementation is due to the varying profiles of different communities, due to sociological, cultural, demographic and anthropological considerations. The following are some of the components for adaptation [4]:

1. Linguistic localization - this is the undertaking of contextualizing the applications user interfaces into the language that is relevant for the users. This eliminates the need for the users to learn a foreign language, an unnecessary barrier to computing, and enables them to use the tools in a language that they are already proficient in.

2. Cultural localization - this involves the adaptation of the Internet tools and applications to fit the cultural framework of a rural community. This is along such dimensions as values, aesthetics, interaction modalities and beliefs so that the implemented solution is aligned with the culture of that community.

3. Application themes - different cultures conceptualize the perfection of form and structure differently. This goes to the extent that colours, shapes, positioning, alignment and use of symmetry for interface components differs for different people groups.

4. Metaphors - the use of metaphors as representation and articulation of reality permeates every aspect of computing, in some aspects more subtly than others. Identifying the key metaphor on the Internet and substituting those with relevant user-centered metaphors is one of the ways of adapting the Internet and computing.

5. Interaction framework - the primary methods and means of communication within a culture influence how that culture approaches interaction with a computer. Some cultures are more predisposed to communicate through music, dance or songs, while others communicate predominantly through written stories. In order for the Internet and computing to be culturally contextualized and adapted, provision has to be made for the interaction modalities that are native within the rural community. A *put that there* input modality might be more appropriate than a text based input modality for a community that is traditionally and predominantly audio and music based [70].

6. Indigenous knowledge integration - the advantage of the Internet to a community is predicated on the possibility for relevant knowledge being accessed and disseminated, from and through the Internet respectively. In a rural community, the Internet infrastructure must explicitly and structurally enable and facilitate the indigenous knowledge capturing, codifying, preserving and sharing.

7. Contextualized strategic training - in addressing the capability, or skills deficiency barrier, training needs to be focused on activating self-driven learning, and encouraging a culture of learning within the community. This ensures that equipping for eLiteracy filters organically through the community, and also allows for learning that is specific to the interest area of individuals in the community.

2.3.1 The human computer interaction perspective

In essence the adaptation mechanisms highlighted above can be informed by the body of knowledge within the HCI domain. These mechanisms facilitate the interation between individuals and the computing agents, and as such can be understood from the theoretical basis of HCI.

The focus of HCI has, in recent years, been on the interface between the individuals and the computer systems. This has been due in many ways to the fact that developing the user interface components is a large fraction of the overall system development effort, and also to the fact that much of HCI is embedded within the cognitive theory approach [71, 72]. In this approach, HCI is a loop of interaction between two information processing agents, where the output from one agent is processed as the input of the other. The limitations that Kaptelinin identifies with this approach are [72]:

1. The fact that the information loop is a closed system, which makes it difficult to take into consideration the phenomena outside the loop.

2. The approach does not allow for an understanding of computer use in the social, organizational and cultural context.

A number of propositions for the theoretical basis of HCI that allows for greater ”ecological validity” have been made, and these include: situated action models, distributed cognition and activity theory [72]. These propositions allow for a consideration of factors within the human computer interaction space in a manner that is sensitive to the social, cultural and environmental context. As such the mechanisms undertaken towards ethnocentric ICT4D are essentially embedded within the deep level HCI that goes beyond the user interaces.

2.4 The knowledge economy

Core to human life is the processing of information that is acquired through the senses (i.e. sight, smell, taste, touch and hearing). Subsequent to the acquisition of information, human interaction is then facilitated by the communication of that information from one individual to another in different forms, formats and media. Efforts to facilitate the communication of information can be traced back from the pictographs of ancient civilization and the Egyptian hieroglyph, and to the Latin alphabet that has permeated most 21st century cultures and civilizations [73]. These efforts aim to facilitate and emulate the natural processing of information in human beings (i.e. acquiring, storing and communicating information). From the early ages of information processing, three paradigms have been prevalent.

1. Cultural - information is an avenue through which cultural and traditional artifacts are encapsulated (e.g. the historical accounts of ancient cultures, or way of life).

2. Economic - from the earliest times when information could be stored and the libraries birthed, information and access to information became of value within societies and cultures, and thus began the commoditization of information. In the 21st century the commodity of highest prevalence is information and its communication - economic empires are built on the creation, storage and communication of information.

3. Political - the invention of writing and printing facilitated the mass distribution of information and subsequently propaganda and political agenda. Those who controlled the printing press controlled the information distributed to the masses and hence the ability to shape the views and opinions of people. To this day, those who control information have the potential to exploit the power associated with access to it.

Knowledge in society is therefore not only proliferating as more knowledge intensive industries come to the fore, or as the economy becomes more knowledge driven, but as every aspect of societal life become facilitated and under-girded by knowledge, hence the term *knowledge society* [74]. The relationship between this term and the closely related *information society* and *knowledge economy* is as follows: the *information society* is the building block of the *knowledge society*, and the *knowledge society* includes the dimensions that are social, cultural, economic (hence the *knowledge economy*), and political [75]. It is generally agreed that we are in a transition from an industrial society to an information and ultimately knowledge society, where the knowledge economy replaces the production economy and knowledge becomes a central economic commodity [48, 32]. Knowledge has always been at the core of industrial activities as dictated by the economics of innovation which require businesses and companies to acquire new knowledge and exploit the current knowledge they possess. The role of knowledge and its centrality in the economy has been accentuated by the ability to process information and knowledge in an efficient and effective manner, and to be able to feed this back into the production system. This dynamic is the key competitive advantage afforded by ICT solutions [35]. Smith identifies four common paradigms and approaches for understanding the changed role of knowledge in the economy [74]: first, is the view that knowledge has become quantitatively and qualitatively more important as an input into the economy; second, is the notion that knowledge has become more important as a product in the economy based on the observation of increases in trading of knowledge products or knowledge based services; third, is the elevation of codified knowledge to a more important position and role than tacit organization knowledge; and fourth, is the acknowl-

edgment of the dependence of economies on ICT, since these technologies facilitate efficient and effective processing of knowledge. Each of these approaches defines a dynamic within the knowledge economy that is contributing towards the formation of this new economic structure. Houghton identifies the following as the characteristics of the knowledge economy [76]:

1. Increased codification of knowledge.
2. A relative shortage of tacit knowledge, as a result of the shift in knowledge stocks.
3. A shift in the organization and structure of production.
4. ICT reduces the investment requirement for a given quantum of knowledge.
5. The increasing rate of accumulation of information stocks is positive for economic growth.
6. Reductions in knowledge dispersion (i.e. increased turnover of knowledge stocks) bridging different areas of competence.
7. The importance of the innovation system and the subsequent knowledge distribution power.
8. An increased shift towards tacit skills.
9. Learning is becoming central to people and to organizations.
10. Emergence of different modes of learning: learning by doing, learning by using, and learning by interacting.
11. Networking is becoming crucial for learning organizations.
12. Initiative, creativity, problem solving and openness are becoming increasingly important skills.
13. Transitions to a knowledge-based-system may make market failure systemic.
14. There is a need to re-examine the conventional economic theories due to the fundamental differences of knowledge-based economy.

Different modes of information distribution and communication can be identified which were further propagated along with the technological information processing inventions. The recent innovations in technology have facilitated a truly revolutionary paradigm shift in information processing. Where information and its distribution were once limited to those who could afford

the high CAPEX associated with a printing press, a television studio, or a newspaper agency, access to and collaboration in creation of information has become accessible to the majority of people at a relatively cheaper price. This has been the core driver of the Web 2.0 revolution which has had far-reaching implications for different ICT implementations (e.g. the Next-Generation pedagogical paradigms associated with eLearning 2.0, blogging and RSS syndication associated with social networking)

The ramifications of exclusion of marginalized communities from participation in this knowledge economy are paramount. From the point of view of the three paradigms of knowledge (i.e. cultural, economic, and political), the marginalization of communities by the digital divide has the following effects: since there is a limitation in codification, and therefore preservation of their cultural expressions and artifacts (notwithstanding the fact that some knowledge is passed from generation to generation), the communities are pushed closer and closer to assimilation and cultural poverty; the communities are not able to participate in information and knowledge exchanges that would benefit them economically; and these communities typically remain at the bottom as far as experiencing the empowerment associated with access to knowledge and information is concerned.

2.5 Indigenous knowledge

Despite the fact that there are numerous definitions of IK, at the very basic level, IK is local knowledge. IK forms a crucial component of the culture and the history of a local community [77]. Knowledge always exists in the context of the entities that create, store, and exchange that knowledge. These range from organizations, societal institutions to communities and people groups. The nature of knowledge processed by these entities ranges from tacit, implicit knowledge to explicit and codified knowledge.

A definition of IK is sometimes made through a discussion of some of its characteristics which include the following [78]: the nature of IK is holistic, socio-cultural and sometimes spiritual; it is predominantly tacit knowledge, and is therefore difficult to codify: it is typically encapsulated in the values, practices, beliefs, relationships, ceremonies, and institutions of a community; it is more communitarian in terms of its discovery and experimentation and its mode of transmission and sharing is often collective rather than individualistic; the community is at the center of IK production and discovery and the motivations for knowledge production are not commercialization and the profit motive; it is predominantly transmitted orally and through imitation and demonstration - therefore its codification can lead to loss of its properties [78];

IK is unique to every culture and society; it is experiential, rather than theoretical knowledge - its validity and relevance is tested in the laboratory of survival of local communities [78]; it is learned through repetition: this is one of the key aspects of tradition, that it gets reinforced and retained through repetition [78]; it is constantly changing and evolving with the changing societal and cultural climate; IK represents the collective, historical and conventional wisdom of a community which provides problem-solving wisdom for that community for everyday life.

These characteristics highlight aspects of IK that give it a unique significance in the context of community development initiatives, at the same time highlighting the challenges that are inherent in the nature of IK that make it a subject worthy of further consideration. Of particular interest for ICT4D are the issues associated with the codification, storage, validation of the integrity, and the integration of IK into ICT infrastructures that are deployed in these ICT4D interventions. In the context of community development, the usual processes and steps involved in the use of IK (Fig. 2.3) start with the recognition and identification of IK within a community. This step has its own challenges associated with access to the knowledge that is by nature tacit, widely dispersed, and closely coupled with group membership and social dynamics. Once this knowledge has been identified, the next step is to validate the identified knowledge along the dimensions of significance, relevance, reliability, functionality, effectiveness and transferability. This validation step is necessary and is directly aligned with and influenced by the purposes for which the IK is being collected (i.e. the relevance of one piece of information will be determined by the domain under focus). For example the knowledge about the genealogy of people in a community may not be directly relevant for a knowledge repository that is implemented for agricultural purposes. Once the validity of the knowledge has been established, the next phases are documentation and storage of the IK. These steps also have their own set of challenges associated with codification of information that is tacit, and maintaining the integrity of the knowledge in the process of codification. Once the information is available and has been stored, it can then easily be transferred and disseminated.

2.5.1 Significance of IK

Every community and group of people possess and daily use IK. It has, however, become increasingly important to consider the IK of marginalized communities, especially in view of the fact that this IK is becoming increasingly marginalized and at the risk of possible assimilation. The IK of these communities is an important contribution to the global knowledge systems and is a contribution that is not being realized due to the marginalization and exclusion of these communities from the digital knowledge economy. The world is suffering the possible loss of this

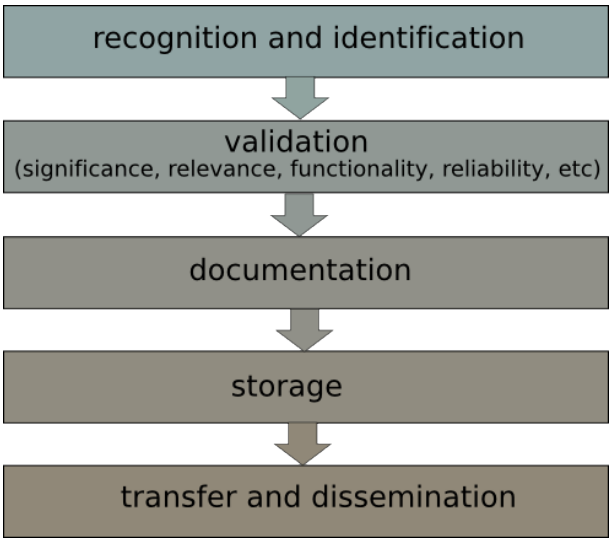


Figure 2.3: IK life cycle [78]

crucial contribution from the sources that have been produced and validated over a long period of time in the lives of local communities.

In the light of the growing number of ICT4D interventions that are being undertaken, IK becomes crucial for the development process because it is part of the social capital of these communities, and therefore part of their means for survival [77]. IK represents the knowledge and wisdom of the local community, and in a sense it acknowledges the value, dignity and essence of that community. The integration of IK into the development processes therefore provides the needed sustainability by being pegged onto the knowledge systems that are already owned by the local community [77]. Building on IK also facilitates in the empowerment of these local communities.

2.5.2 Challenges associated with IK

The integration of IK into the development processes and ICT4D interventions is, however, not without its set of challenges. As highlighted already, some of the basic challenges are with regard to the tacit nature of IK, and the mode of transmission and dissemination does not lend itself to being easily accessible and codifiable. Most of IK is encapsulated in traditions, practices and processes within a community, which are passed from one generation to another through imitation and experiential demonstration. This characteristic of IK makes recognition and identification difficult.

In the context of participating in the global knowledge economy, a number of issues have

to be addressed and taken into consideration. Since IK represents the social capital of the communities, it is important to formulate mechanisms and processes to protect it, with a view to enabling its exchange, promotion, development and validation [77]. One area of concern is with regard to the protection of the intellectual property rights of the IK owners. As a corollary, issues surrounding the ownership of the IK and the implementation of relevant judicial structures to enforce the protection of these rights must be addressed.

The other issue of relevance in the context of exchange of IK is with regard to the protection of the privacy of communities. Local knowledge involves local beliefs and knowledge that may be sacred and belong solely to a certain group of people. The exposure of this information and knowledge to people outside that group may not be entirely appropriate. The measures used to facilitate the exchange of IK should provide mechanisms for its owners to determine the access permissions for different people, and to decide the level of privacy that they want to enforce on their IK.

The dynamics associated with the exchange of IK still have to be thoroughly understood. The opinions associated with this issue include concerns that IK should not be transferred because it is by nature contextualized knowledge and therefore could be irrelevant or misunderstood in a different cultural context. The transfer of IK could also lead to the dis-empowerment of the local communities [78]. At the other end of the spectrum, particularly in the context of ICT4D, is the concern associated with the adequacy, relevance and ability of Western influenced/developed tools to codify IK from different cultures, owing to the different underlying epistemological and ontological foundations. What this alludes to is a simple fact that different people, groups, and cultures have different worldviews which describe and influence their articulation of the nature of reality, their formalization of knowledge and reality, and ultimately the nature of their interaction with reality. These issues are further explored in the next section by discussing knowledge formalization.

2.6 Knowledge formalization

The formalization of knowledge has been a consideration of many academics over a number of years. The diverse forms of conceptualization of knowledge reflect the broad spectrum of issues that are alluded to in seeking to formalize knowledge. Some of these issues include understanding the cognitive and epistemological aspects of knowledge, the role of different entities in a knowledge system and the dynamics of knowledge exchange and communication. The notion of knowledge in many discourses has to do with the concept of *understanding* and the *resolution of*

perplexity [74].

2.6.1 Nature of knowledge and other philosophical considerations

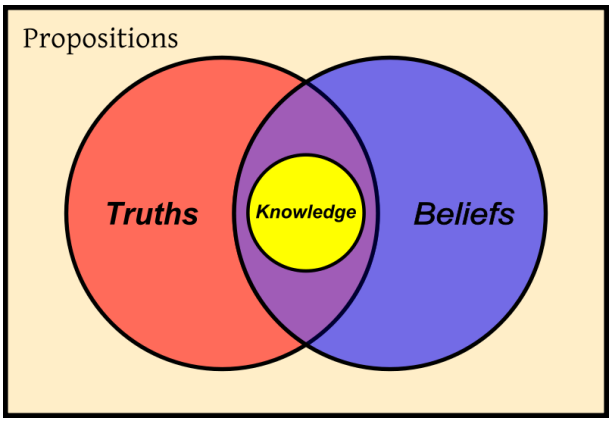


Figure 2.4: Nature of knowledge [79]

The philosophical debates around epistemology, which started several hundred years ago, are still raging today, addressing issues of: the nature of knowledge, the acquisition of knowledge, and the content of what people know. Plato was one of the first philosophers to formulate the nature of knowledge by suggesting that three criteria have to be met for anything to be categorized as knowledge: it must be justified, it must be true and it must be believed (Fig. 2.4), resulting in the Cartesian view of knowledge as the ”justified true belief” [79]. Numerous discussions and critiques have been levelled at this articulation of knowledge. A primary critique of this notion of knowledge was introduced in 1963 by Edmund Gettier in what has come to be known as *Gettier cases*, in which he proposed two counter-examples to the Cartesian view of knowledge [80]. Some of the discussions have been around the notion that a significant distinction has to be made between two main types of knowledge, *a knowledge-that* and *a knowledge-how*, as they represent totally different activities, one being cognitive and the other practical. Needless to say, discussions and theories on epistemology continue to be split along a number of lines: externalism and internalism; empiricism and rationalism; and *a priori* and *a posteriori* knowledge.

Lawson and Plaza argue that the process of acquiring knowledge determines the nature of the knowledge acquired, and in a sense this focuses the question more on the process of knowledge acquisition than on the intrinsic nature of knowledge [81]. Theories around knowledge acquisition are also split between constructivism on one side and realism on the other. For constructivists - ”scientific knowledge is symbolic in nature and socially negotiated. The objects of science are

not the phenomena of nature but constructs that are advanced by the scientific community to interpret nature", and as such within this paradigm knowledge is defined as "temporary, developmental, non-objective, internally constructed, and socially and culturally mediated" [82]. On the other hand realists believe that "the world exists and is organized independent of us, our language, and our methods of inquiry" [83]. Realists view constructivism as [84]:

> "This *(constructivism)* has led to the portrayal of science as a process of constructing and manipulating representations which bear no necessary relation to any ontological reality. In so doing constructivists have forgotten that it is the world that imposes constraints on human thought, and not human thought that imposes constraints on the world."

Few aspects around the discussion on epistemology have the consensus of the whole body of philosophy. One area where there is a general consensus is in the classification of knowledge into tacit knowledge and explicit knowledge [85]. Tacit knowledge (a.k.a. common sense knowledge, or implicit knowledge) is implicitly learned behaviours, while explicit knowledge (a.k.a. scientific knowledge, hard knowledge or codified knowledge) is knowledge that has been encapsulated in external systems and is accessible (cognitively and experientially) to the public [86].

Other classifications along different dimensions include [87]:

1. Meta-knowledge - which entails knowledge about knowledge, and a knowledge that represents a certain awareness and consciousness.
2. Embrained knowledge - which is conceptual skills and abilities.
3. Embodied knowledge - which represents knowledge acquired through practice.
4. En-cultured knowledge - defined as knowledge acquired through socialization.
5. Embedded knowledge - the knowledge that is embodied in the communities' routines and processes.
6. Encoded knowledge - encapsulated in the signs and symbols of a community. The different encapsulations of knowledge include products, routines, and processes.

2.6.2 Knowledge in society

In the previous sections, the importance of the knowledge of societies and communities was enunciated, with a specific focus on IK, its significance and role in societies. While it is important to understand the nature of knowledge, it is also important to understand the processes associated with knowledge creation and knowledge transmission. The work of Nonaka *et al.* stands to provide a framework for the consideration of knowledge in the context of its use in society [87].

Nonaka's framework is built on the following considerations [87]:

1. There are *two forms of knowledge* - tacit knowledge, which is the non-codified knowledge that individuals possess; and explicit knowledge, which is codified knowledge. In Nonaka's framework, knowledge creation is a spiralling process that occurs in the context of the interaction between tacit knowledge and explicit knowledge.

2. There exists *an interaction dynamic/transfer* that facilitates the exchange and the creation of knowledge.

3. There are three *three levels of social aggregation* - individual, group and context

4. The *four knowledge creating processes* (SECI) (Fig. 2.5) [88]: *Socialization* is the process wherein tacit knowledge is shared between the individuals in a community. This happens in the context of joint activities that necessitate close proximity. *Externalization* is the process in which tacit knowledge is explicitly expressed in a manner that is understandable by the rest of the community. *Combination* is the process of generation of more explicit knowledge out of other explicit knowledge; this is done through communication and dissemination of knowledge. Ultimately the *Internalization* process is the conversion of explicit knowledge into tacit knowledge both for the individuals and for the community. This internalized knowledge then becomes practice, habit, and tradition for the individuals and the community.

5. The *space for knowledge conversion* [88] - the concept of *Ba* (Fig. 2.6). Ba relates to the English concept of a place or context, and therefore a space for dynamic creation and exchange of knowledge. This concept recognizes the context-specificity of knowledge, that knowledge has an associated place where it is created, developed and effectively utilized. The four aspects of Ba and their role in the knowledge conversion process are as follows: *Originating Ba* - is the space where feelings, experiences and mental models can be shared between individuals; in this space, the interaction is face to face; *Dialoguing Ba*

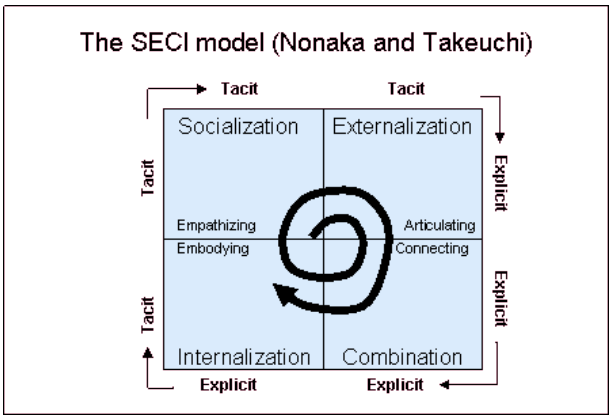


Figure 2.5: Nonaka's SECI model [87]

- is where tacit knowledge is made explicit and this is achieved through dialogue and the use of metaphors; *Systemizing Ba* - a space of collective interaction within a community where new explicit knowledge is created from the existing explicit knowledge; and finally *Exercising Ba* - which is the space where individuals internalize explicit knowledge and convert it into tacit knowledge.

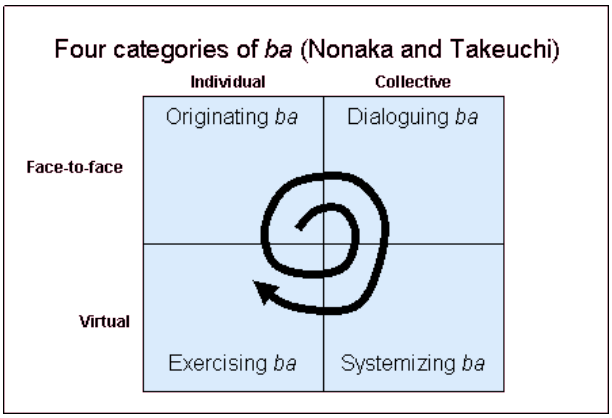


Figure 2.6: Categories of *Ba* [87]

Almost all the knowledge theorists, although differing in their definition and articulation of the different types of knowledge, subscribe to the notion that knowledge exists in a social space [87]. The primary entity in the knowledge environment is an individual and knowledge is constructed within the interactions between the different entities and various contexts. This observation leads to the notion of knowledge as a social construct and that knowledge is an action, hence the activity theory [89]. This observation is akin to Dooyeweerd's articulation of knowledge as the

multi-aspectual knowings (i.e. the action of acquiring knowledge) that individuals engage in within their horizon of experience [43]. The notion of knowledge as an action is one that is supported widely in literature, with some authors arguing that the essence of knowledge ”comes from its exercise, not from its existence *per se*” [90]. The exercise of knowledge is almost always in the context of decision making for individuals, organizations and communities. In Snowden's elaboration of action-oriented knowledge systems, four components are developed [90]: *Explicit/Tacit knowledge* - and he recognizes the role of human beings as carriers or vessels of implicit, tacit knowledge and that external systems and structures encapsulate explicit knowledge; *Knowledge assets,* which consist of both tacit and explicit knowledge; *Trust* is recognized as underlying the knowledge dynamics within a community; and the *Uncertainty* - of decisions with regard to the overall objectives of an organization (or community) and to the cause and effect relations within a knowledge system. A knowledge management ecology within a community comprises a balanced set of these four components.

2.7 Knowledge and ICT

The proliferation of ICT has had significant effects on every day human activity, with these effects extending to the processing of information and knowledge. One of the primary dynamics that can be mentioned is the manner in which knowledge utilization paradigms and roles have changed. Traditionally, there was a distinct separation between the producers of knowledge and the consumers of knowledge, owing to the costs associated with setting up and owning printing presses and media houses (the traditional primary medium of information dissemination). Only a few entities (i.e. large companies, and governmental organizations) would be in a position to publish and disseminate knowledge to the masses via the print, radio and television media. Corollary to that, the benefits (e.g. cultural, economic and political, refer to Section 2.4) associated with ownership of knowledge would only be accessible to those few entities. As a result, knowledge production was very centralized and tightly controlled.

The proliferation of ICT has structurally changed the organization of the knowledge economy by making content production and dissemination accessible to the masses. While the setup of the Internet has always been very open and accessible to people with connectivity, the knowledge exchange landscape really only changed dramatically with the introduction of Web 2.0 technologies which put the authoring of information within reach of the masses. This has been evidenced in the increasing proliferation of blogs and wikis.

The other effects of ICT on knowledge systems have been with regard to access to infor-

mation as far as the tools to access this information are concerned. More advancement and development in computing hardware have meant that there are more and more devices that one can use to access information and owing to their mobility and portability, these have provided a ubiquitous access to information.

ICT generally facilitates a better, more efficient and effective use of knowledge characterized by user-friendly and intuitive knowledge authoring tools, large and more reliable storage capacities for information, and real-time, efficient data communication infrastructures. At the same time, there have been challenges that have had to be addressed that were exacerbated by the growing use of ICT. These issues include ensuring security of information against compromise of its confidentiality, privacy, and integrity, and limiting the accessibility of information only to the people who should have access to that information. The ramifications of exclusion from participation in the knowledge economy have been heightened by ICT, in that those who are marginalized in this age of computing are more at a disadvantage than in previous eras, simply owing to the centrality of computing in everyday life.

Generally ICT has positively influenced the knowledge economy through the tools and technologies that have been developed. However, this dynamic is happening in the context of valid challenges that still need to be addressed.

2.8 Conclusion

The reality of underdevelopment and marginalization has motivated the exploration of solutions toward community development by governments, NGOs, private industry and academia. However as important as the consideration of ICT4D interventions towards community development, is the understanding of the paradigms from which underdevelopment can be formalized. The various paradigms address different aspects of community underdevelopment and propose different solutions. The role of ICT in development is therefore multi-faceted. Instead of being a panacea for all the communities' problems, it is a necessary enabler towards implementing solutions to those problems.

One aspect of ICT that must always be at the fore of ICT4D discourse, and at the core of the interventions, is that ICT is not culture-agnostic, both in terms of it being influenced by culture and in terms of the effect it has on a culture. This necessarily means that ICT solutions must inherently be ethnocentric as far being aligned with the culture where they are being implemented is concerned, and also that the solutions must be flexible enough for implementation of context-sensitive adaptations. A fundamental way in which ICT should be contextualized is by

encapsulating the local (indigenous) knowledge of the community in a manner that activates the community towards greater participation in the global knowledge economy, which is increasing dependent of ICT.

In the next chapter, we discuss the technology developments and architectures for knowledge based applications with a view to providing a technical context for the investigation of an appropriate architecture and knowledge platform.

Chapter 3

Technology Perspective on Knowledge Networking

"The Internet? We are not interested in it." (Bill Gates)

This chapter explores the ICT tools and the developments associated with the global knowledge economy. At the core of these are the Knowledge Representation (KR) technologies and languages that have been developed from the Artificial Intelligence (AI) domain, to address the problem of knowledge modeling and processing. These tools have paved a way for the realization of the Semantic Web (SW), which has been one of the recent major revolutions on the Internet, along with the introduction of Web 2.0. These tools have introduced new knowledge dynamics and challenged the traditional paradigms of information processing, as seen in the transition from a consumer oriented position (i.e. where the majority of the Internet users were just information consumers) to a consumer-producer oriented position (i.e. the users are equally able to consume and produce new information on the Internet).

An increasing number of web applications are transitioning towards Web 3.0, wherein the semantics are inherently integrated into the knowledge processed by these applications. These application are also increasingly being developed and deployed as web services based on Service Oriented Architectures (SOA). There is extensive research into integrating rich semantics into these web services architectures (e.g. OWL-S [27], and ONT-SOA [91]) with a view of facilitating flexibility, automation and heterogeneity on the Internet. At the same time there is also growing research into agent based architectures towards provisioning of distributed and flexible web applications.

3.1 The semantic web

> "I have a dream for the Web *(in which computers)* become capable of analyzing all the data on the Web – the content, links, and transactions between people and computers. A 'Semantic Web', which should make this possible, has yet to emerge, but when it does, the day-to-day mechanisms of trade, bureaucracy and our daily lives will be handled by machines talking to machines. The 'intelligent agents' people have touted for ages will finally materialize" [10].

Inherent in the Tim Berners-Lee vision of the Internet is the Semantic Web (SW), which is a vision of distributed and ubiquitous knowledge that is not encapsulated in the domain expert applications, but rather ingrained in the machine-processable meta-data [12]. This comes out of the realization that most of the information on the Internet cannot be processed by computers meaningfully. The information is rather predominantly designed with the human reader end user in mind. The vision is that computers will be able to semantically process the vast amount of structured and semi-structured information rather than just visual (i.e. layout and display) processing of the data. The SW is an extension of the web in which computing agents can process the data and its semantic definitions [92]. The SW leverages the universality attribute of the Internet, at the same time maintaining a possible compromise of inconsistency and ambiguity. The ambiguity is introduced as more chunks of knowledge are codified from different domains, resulting in the possibility that the lexical construction of the knowledge bases could have different interpretations for similar terms and constructs.

Current research into enabling semantic computation on the Internet is an inter-disciplinary undertaking that requires: philosophical considerations of the epistemological underpinnings of knowledge networks; linguists understanding of knowledge representation attributes of language; AI experts for the formalisms and algorithms adequate for KR and reasoning; computer scientists to design and implement the software applications and protocols that encapsulate and that can process knowledge bases; and sociologists with an understanding of the nature of knowledge networks in societies and the knowledge creation mechanisms inherent in user communities. These efforts are all guided by the vision of a SW where "a new form of content that is meaningful to computers" will unleash new web usage possibilities [12].

3.1.1 SW design overview

The SW provides a common framework for the realization of the web where there is data sharing and reuse among a community of computing agents and human agents [93]. The meaning of

data within SW is represented using ontologies, and the processing of the data and any reasoning on it is implemented through logic relationships and rules that have been encoded into the data. There are four main components that come into play in the realization of the SW: a global naming scheme, in the form of Uniform Resource Identifiers (URIs); a standard syntax for describing data, using Resource Description Framework (RDF); a standard means of describing the properties of that data, through RDF-Schema (RDF-S); and a standard means of describing the relationships between the data items, using Web Ontology Language (OWL) [93]. There has been a variety of opinions expressed around the design elements of the SW and some of these are addressed in Section 3.1.3. Suffice to say, these four components make up the necessary building blocks for an infrastructure that meets the objectives of a universal web of heterogeneous machine processable data.

Grau highlights the following aspects with regard to the design of the SW [92]:

1. A resource in the SW environment is everything that has a URI.
2. RDF and RDFS are based on SKIF [94] which is a non-standard logical representation formalism.
3. RDF(S) syntax is based on triples - a <subject, predicate, object> (<s,p,o>) KR construct.
4. Every piece of knowledge in a SW file should be written as a triple.
5. All triples share a common basic meaning, and are bundled together by a common namespace.
6. The meaning of the triples are independent from each other. The interpretation is only constrained by the commonality of URIs among triples.
7. All SW languages must use RDF syntax and the meaning of the triples should be aligned with RDFS Model Theory.
8. The semantics of ontology layer languages is based on expressive First Order Logic (FOL) formalisms.

3.1.2 SW architecture

Figure 3.1 depicts the traditional architecture for the SW as proposed by Tim Berners-Lee [95]. The stack that constitutes the SW is a hierarchical layering of languages and standards, with the higher layers utilizing and extending the functionality of the layers below.

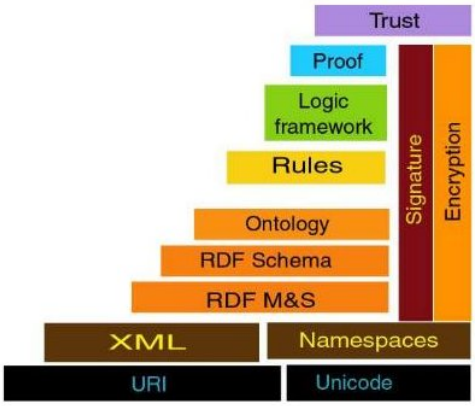


Figure 3.1: The semantic web architecture [96]

In this architecture, RDF, which is a derivative of XML, forms the basis of all the upper layer language constructs. The resources within a SW document are defined as elements that are uniquely identifiable through a URI. While the individual atomic units of knowledge in RDF are the independent triples, namespaces provide the domain of interpretation for the RDF resources and therefore the resolution of any semantic ambiguity. The ontology layer of the stack is implemented through a standardized OWL. The top layers or the semantic stack handle the associated reasoning and inference on the knowledge base through rules and first order predicate logic (FOL).

3.1.3 Issues and discussions

A number of pertinent criticisms have arisen with regards to the realization of the SW. The key issue in the implementation of the SW stack is the over-simplified assumption of the adequacy of RDF for future realization of the SW [96]. The current single stack view of the SW does not allow for an organic evolution of technologies and standards to accommodate the future requirements of the SW. This design decision ignores the frequency with which technology, languages and standards become obsolete on the web, especially considering the already significant concerns of the formal logical inadequacy of RDF(S), and the need for integration of rule-based processing on knowledge bases [96, 92].

As far as the layering of the SW is concerned, one area of contention is the constraining of the expressiveness of OWL by the underlying RDF(S). OWL is based on Description Logics (DL) and frame-based systems, which are very expressive KR formalisms with richer modeling primitives and constructs, and which do not integrate well with the RDF triple construct and

syntax [92]. A workaround solution implemented by the W3C Web Ontology Working Group was the definition of three variants of OWL [97]: OWL-Lite; OWL-DL; and OWL-Full which is a close, same syntax, semantic extension of RDF(S) that does not comply with the standard DL semantics. While there is a clear increase in expressiveness from OWL-Lite to OWL-DL, the division between OWL-DL and OWL-Full is unclear and complex. Grau argues that the only language that can be fully implemented and optimized for SW is OWL-Lite due to the fact that OWL-Full is too undecidable and has a complex, non-standard formalization, while OWL-DL on the other hand is structured on formalisms that still have unresolved problems (i.e. no practical solution to the satisfiability problem for the SHOIN(D) logic [98], on which OWL-DL is based) [92].

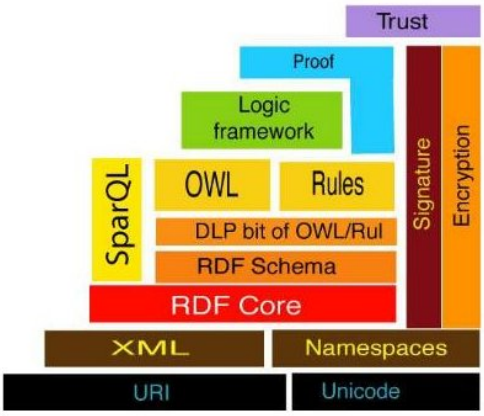


Figure 3.2: Single tower stack of the semantic web [96]

One other argument against the design of the SW is that the architecture should be implemented on multiple independent but interoperable stacks of languages [96]. This proposition argues for the inclusion of other rule-based languages which have matured in their own right and which can provide a big contribution to the SW architecture. These languages should be layered alongside OWL derivatives (Fig. 3.2) to support semantic capabilities that are otherwise not inherent in the OWL-based languages. Horrocks *et al* extended this idea by arguing that the Datalog/Logic Programming layer cannot be adequately layered on top of RDF(S) and suggested instead an architecture of ”two towers versus a stack” (Fig. 3.3) [99].

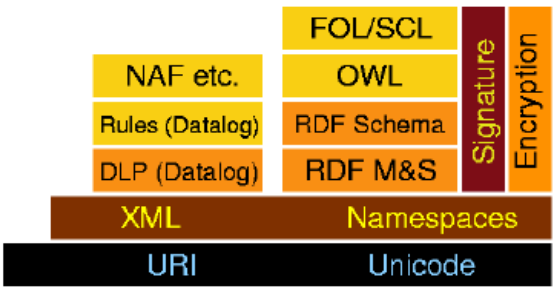


Figure 3.3: Two stacks model of the semantic web [99]

3.2 Knowledge representation

The amount of data and knowledge that is embodied in organization's intranets, databases and individual's data storage media is enormous. Part of making such information usable and available is to derive means of representing it in a manner that allows easy access, ease of use and reuse, and also to provide persistence to the knowledge. Knowledge Representation (KR) has received extensive research into the data types and language formalisms that can be used to represent knowledge. Traditional KR systems have focused on having a centralized and managed lexicography repository that dictates the rules and the grammar to be used within the knowledge systems [12]. This architecture imposes a strict structure on what was designed to be a decentralized and flexible semantic web, due to the lack of scalability and the difficulty of managing a single top-level lexicon in the context of heterogeneity of usage scenarios. Different languages and methods are used for KR and the choice is dependent on the usage scenario of such knowledge, which is usually influenced by the expressiveness of the language and the extensiveness of inference or reasoning on the underlying knowledge.

Five distinct roles have to be taken into consideration in regard to the nature of KR [100]: a surrogate, an ontological commitment, a theory of intelligent reasoning, a medium for computation, and a medium for expression. The first is that KR is a surrogate - a formal substitute for the entities in the real world. This substitution primarily allows for the reasoning about the entity by thinking rather than through action; one can therefore determine consequences of actions on entities in a hypothetical, representational world rather than through interaction with the real entities themselves. Secondly KR operates as an ontological commitment: all KR is only an approximation of reality, and therefore choosing to adopt one KR and not another is making a commitment about what is seen and how it is seen. Thirdly, it is a theory of intelligent reasoning. Fourthly, it operates as a medium for efficient computation and finally it is the medium for human

expression. Each of the different roles of KR necessitates a certain functionality on the part of the KR formalism and also dictates the underlying characteristics of those formalisms.

The choice of the KR formalism to use is influenced by a number of factors. At the very basic level it is a language preference matter, but at the same time it depends on the intended use of the knowledge repository. Each language formalism has an ontological commitment that it makes about what aspects of reality are important and necessary to focus on: therefore rule-based formalisms would tend to emphasize the world in term of objects and their attributes and the rules of inference between the objects; logic based formalisms on the other hand do not emphasize the view of the world as objects and their attributes; and frame-based systems involve thinking about reality in terms of prototypical objects [100].

McCarthy makes mention of the four levels of knowledge formalism, which indicate the different degrees and levels at which formalisms can be applied in representing reality [86].

1. Where a system uses no logical sentences to represent its knowledge but the state of the machine itself implicitly encapsulates its knowledge.

2. The level where a system or a program uses logical sentences in its persistent knowledge base but arrives at its decision making conclusions through *ad hoc* rules that are not based on logical inference.

3. Using clauses to represent sentences and using first order logic and logical deductions to make decisions.

4. The fourth and last level is still a goal to be achieved and this is where the general knowledge and common facts about reality are encapsulated and codified in a knowledge base, easily accessible to any program that requires the common sense/knowledge base to arrive at its decisions.

There are different systems that have been developed for KR. Here after we briefly highlight the key aspects of some of these systems: logic based, rule based, frame based systems and semantic nets. These systems form the foundation of the KR and as such have informed and shaped the current knowledge modeling constructs and languages.

3.2.1 Logic-based systems

Traditionally logic-based systems began as question-answering systems, and later developed as both procedural and declarative representation of knowledge. Logic-based knowledge representation systems typically employ logic programming in which the knowledge is encapsulated in

the coding of the system. However, logic, especial FOL, does not satisfy a number of the requirements for a KR formalism. One of these is the requirement of declarative semantics within the KR formalism to enable encoding of knowledge, independently of the processing performed on the knowledge base. Another requirement is for a mechanism, within the language, of retrieving implicit knowledge that is not explicitly encapsulated in the declarative knowledge base [101]. The other limitation in utilizing FOL for knowledge representation is that FOL has a high degree of undecidability.

There are languages derived from FOL that seek to provide more adequacy in utilizing logic-based systems for KR: these include Modal Logics, and Non-monotonic Logics for representing terminological knowledge, time-dependent or subjective knowledge and incomplete knowledge.

3.2.2 Rule-based systems

In rule-based systems, knowledge is stored as a set of rules in a knowledge base. The rules are of the form

> if *x* then *y,*
>
> *where* x *is some condition and* y *is an action.*

The following are four kinds of rules that can be encapsulated in a rule based knowledge base [102]:

1. Deduction rules (a.k.a. logical implication, material conditional or a Horn clause). For example: if one statement is true then another statement must be true.
2. Transformational rules - relating truth in one knowledge base to the truth in another knowledge base.
3. Integrity constraints of the form, *it must be true that x.* This can also be implemented with a deduction construct *if it is not true that x, then there is an error.*
4. Reaction or Event-Condition-Action (ECA) rules - wherein if a rule applies then there is an associated action.

A full rule-based system consists of the rule-base, an inference engine, and a workspace. The rule-base is the knowledge repository in the system. The inference engine is responsible for the formulation of new implicit knowledge by processing the underlying rules in the knowledge base and based on the data that is available in the workspace. Inference engines are primarily of two types [79]:

1. Forward chaining inference engines - this is also called data-directed inference, because the process of inference is triggered by the availability of new data in the working memory. When data arrives in the working memory, it triggers the rule that matches the new data, based on the *if condition then action* construct. The rule then performs the associated actions which may involve the addition of new data into the working memory, thus triggering more rules.

2. Backward chaining inference engines - this is sometimes called goal-directed or hypothesis driven inference because the process only happens when the system is made to answer a certain goal. In this case the system begins with the goal and searches for the data that correspond to the goal in working memory, or the rules whose conclusion matches the goal. It then tests the condition of those rules to check if they are true, after which, if the condition is true, the rule is used.

3.2.3 Frame-based systems

These are knowledge systems that use frames to represent data. The idea of a frame as a unit of KR was first proposed by Minsky in 1974, as ”a data-structure for representing a stereotyped situation” [103]. Each frame has different kinds of information attached to it: how to use the frame; what one can expect to happen next; and what to do if these expectations are not confirmed. Most of the information encapsulated in a frame is procedural in that it specifies the different actions associated with a frame. Features and characteristics that have become common to frame-based systems are:

1. Frames are structured in hierarchies

2. Frames are constructed out of *slots*, which are the attributes of the frame. For each of the slots, *fillers* have to be specified or computed.

3. Properties of a frame are inherited from super-frames to sub-frames on the basis of an inheritance strategy.

Frame-based systems do not offer a new level of expressiveness compared to the FOL based systems. Instead they provide a way of expressing knowledge in an object-oriented way, offering more efficient means of reasoning by using only a fragment of FOL.

3.2.3.1 Description logics

Description Logics (DL) is a set of knowledge representation formalisms which are descendant from frame-based formalisms and are based on FOL semantics to capture the declarative part of frames [92]. DL is also sometimes called terminological logics, concept languages, or attributive description languages [104]. In DL, a concept hierarchy can be built out of atomic concepts, which are specified as unary predicates, and out of attributes (roles), which are specified as binary predicates [105].

A DL interpretation is a mathematical structure [92]:

$$I = \left\{ \triangle^I, .^I \right\}$$

The domain of interpretation is $\triangle^I$, which is divided into the abstract domain and the concrete domain. Together these form a set of all the domain individuals and the data values respectively [92]. The DL structure also contains a function $.^I$ which maps: every concept name to a subset of $\triangle^I$; every role name to a subset of $\triangle^I * \triangle^I$; and every individual to an element of $\triangle^I$. DL allows for an extension of the interpretation function to formalize more complex class and property structures in a knowledge base.

3.2.4 Semantic nets

Gray *et al* define a semantic net as

> "a structure which is used to represent associations between objects. For each object of a given type, there is a corresponding collection of attributes which are applicable to it; some of these provide simple values, but others are found by following links in the net, which connect the object to other objects of various type" [106]

Semantic nets are also called mind-maps, based on the assumption that this is akin to the human cognitive modeling of knowledge. The main underlying idea in semantic nets is that the semantics of a concept is derived from its relationship to other concepts, and that the information in a net is stored by the interconnection of nodes with labeled arcs. An example of a semantic net representing knowledge about a bird is depicted in Figure 3.4.

The inference strategy in semantic nets is to follow the links between the nodes using an inheritance or intersection search strategy, in order to arrive at conclusions and to derive implicit knowledge from the system.

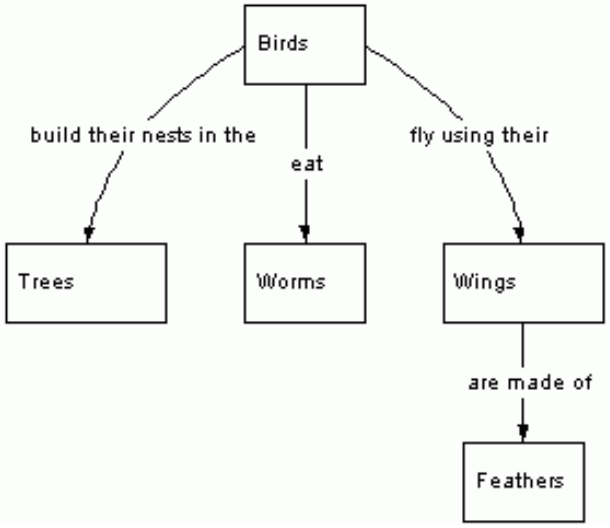


Figure 3.4: Semantic net example [107]

3.3 Common sense and intelligence

An integral objective of knowledge based systems and applications is to emulate the expertise of a human in solving problems and making decisions. One critical quality and characteristic of human beings is the ability to *solve problems* by *reasoning intelligently* about the world based on their accumulated *common sense*. These aspects (i.e. problem solving, reasoning, common sense and intelligence) have been a subject of extensive study and research in the knowledge modeling domain.

Common sense is the basis of intelligence and the basis of decision making in the common sense world [86]. Decision making in the real world involves dealing with ambiguities and uncertainties that require a wide range of pre-acquired knowledge within different domains and contexts.

Within the computing environment, the environmental ambiguities are constrained and minimized through a narrowly defined problem domain that an 'intelligent' system operates on. An intelligent system is therefore defined as one that perceives its environment and responds and performs actions to achieve maximum success [79].

Formalization of common sense knowledge has been a rather difficult task for philosophers, with some philosophers (e.g. Wittgenstein) claiming that common sense cannot be formalized and that mathematical logic is not the right tool for the task [86]. The monumental task of KR, especially with efforts toward codification of common sense knowledge, is with the end goal of facilitating intelligent behaviour in computing agents

3.4 Problem solving and reasoning

Various disciplines have contributed to AI to inform the study of reasoning, including mathematical logic, psychology, biology, statistics and economics. Each of these disciplines has contributed a distinct view and perspective on intelligent reasoning (Fig. 3.5) [100].

Mathematical Logic	Psychology	Biology	Statistics	Economics
Aristotle				
Descartes				
Boole	James		Laplace	Bentham Pareto
Frege Peano			Bernoullii	Friedman
	Hebb	Lashley	Bayes	
Goedel	Bruner	Rosenblatt		
Post	Miller	Ashby	Tversky,	Von Neumann
Church	Newell,	Lettvin	Kahneman	Simon
Turing	Simon	McCulloch, Pitts		Raiffa
Davis		Heubel, Weisel		
Putnam				
Robinson				
Logic PROLOG	SOAR **KBS, Frames**	**Connectionism**	**Causal Networks**	**Rational Agents**

Figure 3.5: Intelligent reasoning [100]

From mathematical logic, the perspective on reasoning is that it is a set of formal calculations involving steps of deduction. Psychology offers the insight that reasoning is a human behaviour and therefore extensive study is made into the process of human problem solving and into development of large knowledge bases. Biology concentrates more on the architecture of the entities that achieve intelligent reasoning, therefore focusing on the stimulus-response behaviour and the interconnection of numerous processors. Statistics provide the aspect of uncertainty to logical reasoning and the mechanisms of handling uncertainty. Finally economics derives the utility theory to introduce a notion of values and preferences that contribute to intelligent reasoning [100].

Different strategies and mechanisms for intelligent reasoning have therefore been realized and some of these are discussed in detail in the following sections. Besides the developments in reasoning formalization and strategies, there have been other developments that have facilitated and nurtured the intelligent reasoning and problem solving environment and these include: technical developments in machine learning and reasoning; the growth in CPU and memory ca-

pabilities; the evolution of the web and the resultant provisioning of connectivity, content and services; and the growing availability of data resources [108].

3.4.1 Causal reasoning

Causal reasoning is a type of inductive reasoning in which causal relationships are established among a set of events. It is based on the fact that some events, the causes, are systematically related to some other events, the effects. Therefore the environment can be altered through facilitating or preventing certain events from happening. A significant distinction has to be made between causal perception and causal reasoning. Causal perception is the ability to infer a causal relationship between events without conscious effort. Causal reasoning on the other hand is the systematic process of establishing a cause of an event that is usually out of the ordinary. At the core of causal reasoning is the establishment of adequate evidence for causation, based on the logical material implication construct [109]. Part of articulating causality involves the consideration of causes that are both sufficient and adequate for the effects to occur. Apart from relying on uniformity and regularity of observation of events, Stuart Mill devised five methods that can be used to determine the nature of causal relationships between observations [110]: the method of agreement - similar effects are likely to be a result of similar causes; method of difference - with all things being equal, different effects are likely to arise from different causes; the joint method of agreement and difference - a combination of the first two methods to say that genuine causes are necessary and sufficient conditions for the resultant effects; the method of concomitant variation - there is a correlation to the degree and the extent of the cause, and to the degree and the extent of the effect; and lastly the method of residues - if elements of a complex effect are shown to have been a result of elements of a complex cause, then whatever remains of the effect must have been caused by whatever remains of the cause.

3.4.2 Model-based reasoning

Model-based reasoning refers to inference technique that uses formal models for the task of process modeling, verification, monitoring and diagnosis. The models describe a particular domain in terms of the relationships and behaviours of the smaller units that make up the domain. Some of the models used include automata, process algebra, and Petri nets. In general three different types of models are used in model-based reasoning systems [111]:

1. Quantitative models - are used to describe continuous behaviour, and therefore have an infinite domain of variables. This description of behaviour is achievable through the use

of equations, inequalities and logical sentences, which allows for formulation of rules that can be solved with a constraint solver.

2. Qualitative models - are used when variables are drawn from a finite set, and support the capture of aspects of a behaviour that help in understanding the model and the results that are produced by the model.

3. Hybrid models - combine both the quantitative and the qualitative aspects of model systems. This could be applied for example to systems that have internal states and at the same time continuous behaviour.

3.4.3 Case-based reasoning

The history of case-based reasoning (CBR) is in the work that was done on dynamic memory and the role that earlier situations, cases, episodes and situation patterns have in problem solving and learning [112]. In CBR, past experience and cases already experienced are used in the reasoning strategy of the system for the purposes of solving the new cases. The strategy of using past experiences in the determination of solutions to new problems is a mechanism that is regularly applied by humans to solve problems. This is supported by observations from cognitive psychological research.

Aamodt defines CBR as

> ”a cyclic and integrated process of solving a problem, learning from this experience, solving a new problem, ...” [112].

CBR is also an approach to long term, incremental and sustained learning as new experiences are retained each time a problem has been solved. A typical CBR cycle (Fig. 3.6) is as follows: retrieve similar cases from the knowledge base; reuse the knowledge in that case; revise the proposed solution; and finally retain the experience that is likely to be useful in the future.

CBR is similar to analogical reasoning in which analogies are used to make connections between different concepts, and to transfer knowledge from a well understood domain to a new and unfamiliar domain. Different types of CBR are articulated by Aamodt: exemplar-based reasoning; instance-based reasoning; memory-based reasoning; case-based reasoning (differentiated from the generic term); and analogy-based reasoning. All of these types differ based on the methods they employ to organize, retrieve, use and index knowledge [112].

As a problem solving paradigm, CBR does not rely on the general knowledge of the domain or on making associations along problem descriptors and conclusions [112]. As such the

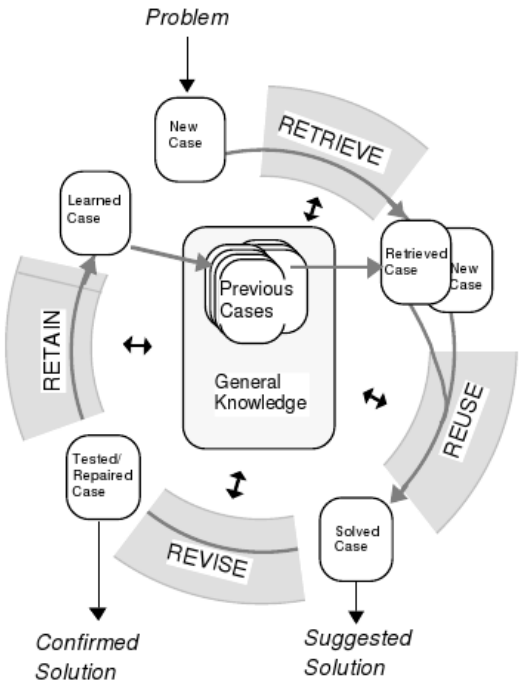


Figure 3.6: Case based reasoning cycle [112]

representation, indexing and organization of cases is very important towards achieving higher efficiency in their retrieval [113].

CBR is also a machine learning paradigm that enables continuous retaining of experience through the updating of the case base once a problem has been solved.

3.5 Ontologies

Ontologies are a formalism that is being predominantly used for KR. An ontology is simply defined as a body of knowledge about a specific domain [114]. Ontologies encapsulate not only the domain vocabulary, but also the relationships between the different domain concepts. This level of formalization of knowledge is necessary to enable wide scale re-use and sharing of knowledge between different domains. It is also essential in facilitating more effective use of knowledge through machine-understandable representation formats. Ontologies promise a ”shared and common understanding of a domain that can be communicated between people and application systems” [115].

A basic life-cycle in the use of ontologies begins with the knowledge acquisition stage. This

involves the basic gathering of the knowledge to be contained in the ontology. Most of the knowledge is currently encapsulated in webs and heaps of unstructured and semi-structured information. In order to make this information usable, ontology extraction tools have been developed that use natural language heuristics to derive semantics from the information. Instead of extracting knowledge from the existing unstructured/semi-structured Knowledge Bases (KB), ontologies can be designed and populated from scratch. The second stage of the process is the KR stage, where a choice of the formalization of knowledge is made. This is influenced to a large extent by the intended usage of the ontology. This phenomenon is commonly known as the task-dependence of ontologies [114] and is evidenced in the differences in the sub-classing of the *thing* root class in the various super-ontologies. For example, the CYC ontology subclasses *thing* into *individual object*, *represented* and *intangible* [116]. The Wordnet ontology on the other hand subclasses *thing* into *living* and *non-living* [117]. Therefore depending on the intended use and the domain of consideration, the underlying formalism is selected. The third stage in the process is the knowledge maintenance stage which involves ensuring the validity, accuracy, and the internal consistency of the ontology. The last stage is the actual usage of the ontology in support of the daily domain specific processes. Hwang *et al.* documents a more detailed and granular ontology construction process that includes determining the scope of the ontology, and considering reusing existing ontologies [118].

3.5.1 Role of ontologies

Ontologies play a crucial role in the achievement of the vision of the SW. The four factors inherent in the SW that are met by the usage of the ontologies are :

1. Human understanding - encapsulated in the ontologies is the domain expert knowledge that is usable and understandable within that specific domain.

2. Machine processability - one of the limitations with the data repositories on the Internet is the fact that web applications and system agents cannot decode the semantics inherent in the data. Ontologies enable machine processability by using meta-data annotations and KR languages that can be processed by software agents.

3. Knowledge sharing - once knowledge is codified in an ontology, it can then be shared and communicated to the domain actors. Numerous tools exist for extracting information from ontologies. These include query languages such as the RDF Data Query Language (RDQL) and visual Ontology presentation tools.

4. Knowledge re-use - once an ontology has been developed, it is accessible to be used in other systems without the need for replicating the same efforts. This is made possible on a wider scale by the use of generic frameworks and interfaces such as the OKBC.

3.5.2 Ontology languages

Ontology languages are used by ontology editors as a formalism and representation of the specific domain knowledge. Numerous languages have been proposed for KR, some of which include the Ontology Markup Language (OML) and Conceptual Knowledge Markup Language (CKML). In this section we explore in more detail four of these ontology languages.

3.5.2.1 SHOE

Simple HTML Ontology Extensions (SHOE) was developed to mitigate the problem of semantic in-operability due to various organization's Document Type Definitions (DTDs) [14]. This problem is not solved by XML and is only partly addressed by the RDF specification.

3.5.2.2 RDF

Resource Description Framework (RDF) is W3C XML based standard for the representation of meta-data. RDF uses graph theory concepts to represent knowledge as connected nodes, and to define the relationships between nodes. The minimum unit of knowledge in RDF is a triple consisting of an object, a predicate and a subject. A triple represents a single fact in the knowledge base and is the only syntactic construct available in RDF. Higher layers in the SW hierarchy also use this construct, as the entire SW architecture is predicated on RDF and RDFS. In graph theory language, a predicate is the edge between the nodes. The structure of the RDF file is defined by RDF(S) language which specifies the constructs that can be utilized in the development of an RDF ontology. The RDF(S) constructs that are used to declare classes, resources, and properties of resources are *rdfs:class*, *rdfs:resource* and *rdfs:property* respectively. RDFS also defines meta-properties that can be used in order to differentiate between domain concepts by a means of XML namespaces. This allows specific object classes and values to be tied to a publicly available definition of the concept, which is referenced by a Universal Resource Identifier (URI) [12]. In RDF, a resource is therefore defined as everything that has an identity (i.e. everything that can referred to by a URI) [92]. The individual URIs and the ontology namespaces form the domain of interpretation of the ontology, and are used for disambiguation and resolution of semantic complexity within an ontology.

RDF allows for more ”syntactic freedom” in the definition of an ontology, in that a resource can occupy any position within a triple, making any of the following triples valid [92]:

```
<rdf:type, rdf:type, rdf:property>
<rdfs:range, rdfs:domain, rdf:property>
<rdfs:class, rdf:type, rdfs:class>
```

Other variants of RDF(S) include RDF-Schema Fixed Layer Meta-modeling Architecture (RDFS (FA)) and RDF-Schema Description Logics (RDFS(DL)). RDFS(FA) is a sub-language of RDF that has been designed to handle some of the problems in the layering of the SW, in particular the loss of expressiveness experienced via RDF(S) [119]. RDFS(FA) has been developed to provide standard FOL semantics and meta-modeling architecture, therefore RDFS(FA) implements standard DL semantics.

RDFS(DL) on the other hand is a sub-language of RDFS(FA) with a suppressed meta-modeling architecture [92]. This design feature is prompted by the fact that meta-modeling in RDFS(FA) was motivated by the need to align the language with RDF(S) and its suppression does not lead to loss in expressiveness or inference capabilities [92].

3.5.2.3 DAML + OIL

DARPA Agent Mark-up Language with Ontology Inference Layer (DAML + OIL) is an ontology language that has been developed through the merging of DAML and OIL. DAML and OIL are comparable in terms of their basic knowledge modeling constructs. OIL however achieves a greater compatibility with RDFS than DAML and it also handles reasoning more efficiently than DAML. OIL is designed to provide the basic knowledge modeling primitives similar to those of frame-based languages, and to allow simple and clear enough semantics based on DL, with support for inference and automated reasoning [120]. The sub-languages of OIL are:

1. Core OIL - handles basic knowledge representation and reasoning with added reification functionality.

2. Standard OIL - data modeling based on frame logic primitives (e.g. classes, slots, and roles).

3. Instance OIL - this provides database integration functionality.

4. Heavy OIL - adds more inference and data representation capabilities.

3.5.2.4 OWL

Web Ontology Language is a formal language for representing ontologies on the web. It was developed as an extension to the DAML + OIL language. Beside the basic object description capabilities of most ontology languages, OWL also provides features to define *range* and *domain* restrictions on properties [98]. For example, instead of just being able to define the following:

```
Noname is a Boy
South Africa is a Country
Gracie is a South African
```

OWL also provides the ability to define more characteristics about the classes and the restrictions on the properties, for example:

```
Noname and Gracie are disjoint classes {restrictions on classes}
Person can be a citizen of one or more Countries
```

The main influences on OWL have been from DL, the frames knowledge representation paradigm, and the RDF/XML specification: the semantics of the language is based on DL; the syntax and the structure is influenced by frame-based systems; and it has been designed to be compatible with RDF(S) [92]. OWL has different sub-languages to cater for the different usage profiles: OWL-Lite, OWL-DL (encompasses OWL Lite), OWL-Full (encompasses OWL DL), OWL-Eu and OWL-E.

3.5.3 Ontology development

Different actors are involved in the knowledge networking paradigm, and these actors play various roles in the system. These include: the domain experts, who are the source of the expertise within the concerned domain; knowledge workers, who are vested in the tools and the formalisms of KR; and also the knowledge users, who make use of the ontologies that have been developed. In the ontology development process, tools are used to extract knowledge from semi-structured and unstructured information on the web, and also to help knowledge workers to construct and develop new ontologies. A detailed assessment of the various ontology tools has been made by Corcho *et al.* which highlights the different features and capabilities of these tools [121]. The following are some of the ontology tools that are used by knowledge workers:

1. Protege [122] - is an ontology editor and a knowledge acquisition system. More than that, Protege is a knowledge based framework based on JAVA, that provides an extensible and flexible plug-and-play environment for application development. Protege provides a set of knowledge modeling structures and processes that support the creation, visualization, and manipulation of ontologies in numerous representation formats. The two predominant ways of modeling ontologies are through Protege-Frames (in line with the OKBC protocol) and Protege-OWL editors.

2. OilEd [123] - has not been developed as a full scale ontology editor but rather as a minimal system for the purposes of learning about ontology technologies and the reasoners. OilEd is integrated with a Fast Classification of Terminologies (FaCT) reasoner for the purposes of checking and validating the consistency of ontologies. It allows for the exporting of ontologies into different formats (including OIL-RDF and DAML-RDF).

3. pOWL [124] - is an open source PHP based ontology editing web tool. It handles the editing of ontologies based on the OWL standard. The advantages of pOWL over some of the other ontology editing tools include the possibility for distributed collaborative development of ontologies. pOWL is implemented in a four layer architecture: pOWL store layer, which is the SQL database back-end. This back-end database is accessible to the user through an SQL inspired RDF Data Query Language (RDQL); RDFAPI, RDFSAPI and OWLAPI layer which are APIs for handling the corresponding ontology formats; pOWL API layer which allows for the implementation of applications on top of the ontology representation APIs; and User interface layer, which handles the user interface components for interacting with pOWL [124].

3.5.4 Ontology integrity

The nature of domain expertise, and organizational intelligence, is such that it is distributed among different individuals in a community. One of the ways of codifying such knowledge is through developing an ontology for that domain. In order to capture detailed and distributed knowledge, tools that enable distributed and multi-user ontology authoring can be deployed. The strengths of multi-user authoring systems have been displayed in the success of Wikipedia and other community wikis [125]. But coupled with the strengths of the multi-author and distributed architectures, is the increased possibility of inconsistencies, instance inaccuracies, and low quality of the resultant ontology [126]. The integrity and trustworthiness of an ontology directly influences its usefulness and applicability. In developing end to end knowledge man-

agement systems, measures should be implemented to maintain the ontology integrity. One of such intervention is the "Trustworthy Ontology Approach", in which an intermediate ontology is populated by the domain experts, and then validated by a domain administrator using rules and checks before the triples can be included in the main ontology [126].

3.5.5 Ontology interoperability

Semantic interoperability is a crucial feature of knowledge engineering because without it, it is impossible for knowledge to be shared across different, distributed systems [119]. One of the strengths of the SW, machine processability of semantically annotated data, is stifled by lack of interoperability between ontologies. The lowest level of providing compatibility within knowledge engineering is at the KR language level. If two languages are compatible, the interpretation of the data encapsulated becomes trivial and seamless. Compatibility of the ontology languages also means that plugging in a single inference mechanism (e.g. FaCT, Tracer) on the data is possible due to the bidirectional connection between the languages. RDF(S) has a mechanism of providing a basis for the implementation of more expressive languages. However, this mechanism has been found problematic for extending RDF(S) semantics to specify OWL constructs [127]. Pan and Horrocks propose RDFS(FA) as an extension of RDFS to enable more expressive compatibility with OWL, allowing RDF to function as the foundation for the SW [119].

3.5.6 Ontologies instances

The following are a few ontologies that reflect the scale and variety of usage goals and domains that ontologies can be developed for: OpenCYC; Wordnet, and SUMO are upper ontologies which are designed to capture the top level, meta, abstract and generic terms and concepts. Lexical mapping sometimes exist between these upper ontologies.

1. OpenCYC - this a open source derivative of the CYC ontology which was created and maintained by the CYC Foundation. OpenCYC is a formalized common knowledge (consensus reality) coupled with a commonsense reasoning engine [116]. OpenCYC is meant to be used as a knowledge layer for intelligent applications that utilize commonsense knowledge as their domain. OpenCYC is bundled as a knowledge server, an inference engine, knowledge base browser and an API for accessing the OpenCYC knowledge base. Interaction with the server is through making assertions and querying the knowledge it encapsulates.

2. Wordnet - is a lexical database for the English language which was developed at Princeton University. This ontology originated with the words from Brown Corpus, that was then merged with other English lexical works including Laurence Urdang's *Basic book on synonyms and antonyms*, and Robert Chapman's edition of *Roget's international thesaurus* [128, 129]. Wordnet groups nouns, verbs, adjectives and adverbs into sets of cognitive synonyms called synsets, with each synset expressing a distinct concept [117]. These synsets are then linked to each other based on the semantic and lexical relations between them. The Wordnet ontology currently contains 147,249 unique synsets and there are undertakings to implement wordnets in other languages.

3. Suggested Upper Merged Ontology (SUMO) - is created as part of the IEEE Standard Upper Ontology Working Group. The four design goals of SUMO are: promoting data interoperability, enhancing information search and retrieval, enabling automated inferencing and allowing natural language processing [130]. SUMO is released in various representation formats including OWL, KIF and Protege data format.

4. Dublin Core Metadata Initiative (DCMI) - is an open organization involved in the development of interoperable online meta-data standards that support a broad range of purposes and business models [131].

5. DAML-S - is an upper ontology that is used within the web-services framework, to facilitate the automatic discovery of web services and to add semantics to service descriptions. Some of the predicates that are defined within this ontology, that are used in the service description include [132]: *presents*, which has as its range a *serviceProfile* class; *describedBy*, where *serviceModel* class provides the description for the service (the range of the predicate); and *supports*, which has the *serviceGrounding* class as the range, to specify the supported services.

6. Gene Ontology - is an ontology, maintained by the Gene Ontology consortium, that encapsulates gene related concepts for use by biologists. It started as a merging of three organisms databases: flybase database, Saccharomyces Genome Database and Mouse Genome Database. The ontology describes the gene products in terms of biological processes, cellular components and molecular functions [133].

Ontologies are growing to be the *de facto* method of capturing domain specific knowledge, and as a result the number of different ontologies is constantly growing. Currently Swoogle [134], a SW search engine, is indexing more than 10 000 ontologies. The strength of the SW is the

heterogeneity of the ontologies. There is however still a lack of tools and techniques that facilitate the integration of different, but similar ontologies [135]. For example, ontologies that encapsulate the knowledge around agriculture could have differences due to the KR language used, ontology engineering methodology, or a different point of view of the authors. Four concepts that are pertinent in the context of multiple ontologies usage are:

1. Mapping - this involves using the equivalence construct to relate concepts or predicates from different ontologies [136].
2. Aligning - the process of removing inconsistencies between two ontologies [137].
3. Merging - combining ontologies about the same domain or subject into one ontology [138].
4. Integrating - building a new ontology from existing ontologies [139].

Kong *et al* suggest an ontology merging technique that relies on using the Wordnet ontology [135]. In their proposed merging process, the first step is mapping the domain concepts into the Wordnet ontology, in order to get a richer set of values to perform a similarity comparison. Once the initial mapping has been done, a restricted set of candidate concept pairs is selected, and a similarity computation is performed on that set using both the Jaccard coefficient and Most Specific Parent (MSP) techniques [135].

3.6 Taxonomies

A taxonomy is a classification of entities and arrangement in a hierarchical structure, and therefore taxonomies are the basis for the classification and indexing processes. Taxonomies are another tool that has been explored for usage in the context of facilitating knowledge representation. The key distinction between ontologies and taxonomies is that while ontologies typically define the knowledge within a whole domain, taxonomies only represent the subtype-supertype relationships between entities [132]. Taxonomies serve the purpose of defining the vocabulary for use within a knowledge system and a framework for structuring a knowledge base [140].

3.7 Folksonomies

Folksonomy traditionally referred to shared vocabularies, but has evolved to become a collaborative and open end-user driven classification technique, that accepts the notion that users can create

useful results when given freedom from the formalized rules of indexing, or from an established vocabulary [141]. This view is pegged in the notion of the Web as a system that operates "as close as possible to no rules at all" as articulated by the inventor of the Internet, Tim Berners Lee [10]. In this complex system, as described in the complexity theory, there is no central controlling entity, but rather a spontaneous self-organizing pattern that is achieved as every unit follows basic and simple rules [18]. Folksonomy also goes by the following references: collaborative tagging; social classification; social indexing; social tagging; and free-tagging classification. All of these names reflect an aspect of the nature of folksonomy.

The proliferation of folksonomies has been aided by developments in Web 2.0 technologies that enabled natural language tagging of resources. This process of natural language tagging and labeling inherently introduces contextual variability and ambiguities, which require an extensive analysis of the context domain to facilitate lexical disambiguation of the labels [141]. The motivations for using folksonomies have to do with the fact that some of the data that has to be used and stored is spontaneous, unstructured, volatile and cannot be encapsulated in normal corporate knowledge repositories [142]. This type of information typically exists in a fairly closed system, and in tightly coupled processes between the supplier and the recipient of the information. Therefore due to that nature of the information, no sufficient systems were available to retrieve that kind of knowledge, which typically resided with the people and would get lost when they ceased to exist relative to the organization or a community.

In a folksonomy based knowledge system, tags are dynamic and local within a community of users. They are not part of a controlled vocabulary and therefore play a role of providing *ad hoc* categorization for the information in the community [143]. This feature of folksonomy is intrinsically at odds with the top down taxonomy design. Some of the biggest criticisms against folksonomy are leveled at this aspect, which sometimes leads to categorization that does not reveal immediate structure, and that is difficult to process and reason upon, due to the lack of an internal hierarchical structure. The mechanism that has sometimes been recommended for retaining the best of both worlds is using a collabulary [18]. A collabulary, also called a supervised folksonomy, is a compromise between folksonomies and controlled vocabularies, where there is a collaboration between classification experts and content consumers to create a rich, and systematic tagging system. Another solution that captures the advantages of both folksonomies and structured vocabularies, is a taxonomy-directed-folksonomy. In this mechanism, tags that are already in a formal taxonomy are suggested to the user via the user interface components. However users can still use their own tags if there are no adequate tags in the taxonomy.

3.7.1 Folksonomies advantages

The following are the advantages that are realized through the use of folksonomies [18]:

1. Low entry cost, since there is no formal vocabulary or taxonomy for the users to learn.
2. Open ended and therefore easily and quickly adaptable.
3. They are simple, and can be created, shared and re-used.
4. Democratic meta-data generation, both by consumers and producers.
5. Folksonomies cater for the ”long tail” - a category of web users who search the web using less frequent keywords.
6. Leads to a richer vocabulary that is relevant and accessible to the community of users.
7. They facilitate navigation by providing dynamic hyperlinks between documents
8. The tags are shared and comprehensible by the user base.
9. Ability to adapt quickly to language changes
10. Categorization on folksonomies follows a phenomenon observable on complex systems, that might actually offer the only chance to create indexing systems that would be appropriate for large dynamic information systems, in ways that a single controlling entity cannot [18].

3.7.2 Folksonomies challenges

One of the biggest challenges of folksonomies is that information retrieval on folksonomies is non-trivial [142], due to some of the following disadvantages [144]:

1. A high level of tag variation - anyone can choose any tag, and it is almost impossible for the platform to make the connections between similar tags.
2. Polysemy - the situation wherein a tag has a number of different meanings.
3. Synonymy - where different tags have same or similar meanings.
4. Folksonomies are flat - the overall structure of tags is flat, making tag searching more difficult.
5. Folksonomies can lead to the problem of mob-indexing and meta-noise.

3.8 Social networks

One of the growing phenomena on the Internet, along with proliferation of Web 2.0 tools and integration of semantic capabilities on the web, is the increasing number of online social networks and communities. This phenomenon has largely contributed to the ”small world” aspect of the Internet (discussed in Chapter 1), which provides online spaces where individuals interact, and exist in a community regardless of the traditional barriers to communication (i.e. geographical distances, and social status barriers). The effect of this is that the world is evolving into a globally connected set of communities. The notion of a social network alludes to a structure made of nodes that are tied by a common interdependency (Fig. 3.7): interests, values, relationships, and hobbies. The term ”social network” has traditionally been used to refer to a set of relationships between individuals, at small interpersonal scale and large international scale. Social networks have been studied extensively in disciplines such as sociology, anthropology, communication studies, organizational studies and sociolinguistics.

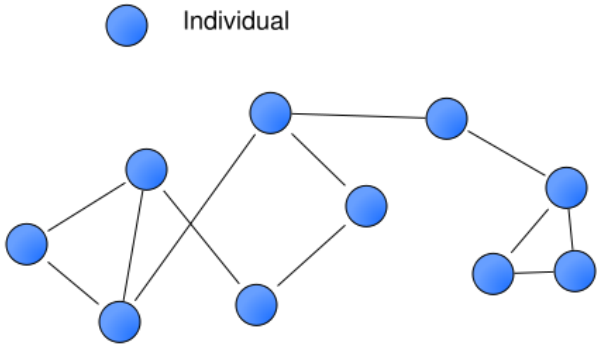


Figure 3.7: Social networks [79]

It is important to consider the role of social networks in knowledge systems because knowledge always exists in the context of a community, as explicitly alluded to in the SECI model of knowledge processing (Section 2.6.2), and in the discourse on IK (Section 2.5). From the perspective of social constructivism, knowledge networks are social networks - this taps into the philosophy underlying understanding of knowledge as a social construct. The more positivist perspective, however would tend to refrain from equating knowledge networks to social networks.

3.8.1 Virtual, online communities

In recent years, with the proliferation of ICT and the associated information tools, there has been a growing interest in social networks from Information Sciences perspective. ICT provides social networks with the communication and interaction infrastructure that subsequently feeds into the information and knowledge life cycles. ICT has contributed to the proliferation of what has come to be known as virtual communities, online communities, or communities 2.0. The nature of virtual communities and the dynamics associated with them have been the subject of many discussions: what is the essence of relationships in a virtual community? Is ”community” an appropriate and relevant term to use for these entities? What are the elements of virtual socialization? And what are the motivations and criteria for membership on an online community?

Online communities tend to emphasize the following social attributes [79]: identity, reputation, presence, relationships, groups, conversations, and sharing. Some of these attributes are derived from experience in physical communities and form structural pillars on which communities are founded.

One of the technological developments that have facilitated the proliferation of online communities is the growing integration of SW technologies into online social networking. This has been realized through the implementation of ontologies that are defined for the social networking domain. Friend Of A Friend (FOAF) is an RDF-based ontology that describes persons, and their relations with other entities. Tim Berners-Lee described FOAF as the initial piece of technology that will facilitate the realization of the Giant Global Graph (GGG): ”I express my network in a FOAF file, and that is a start of the revolution” [145]. GGG is a redefinition of the SW to describe a system where relationships transcend networks and documents.

3.8.2 Virtual community dynamics

Within any community of entities, there are emergent dynamics that occur as a result of the interactions between the different individuals. There is continuing research into understanding these dynamics and effort to develop frameworks and models to encapsulate and articulate the nature of these systems.

Kollocks has developed a framework for detailing the motivation for participation in online communities. He identifies three factors that contribute to participation [146]: anticipated reciprocity - the expectation on the part of the participant that they will receive from the community to the measure that they put into it, a basic give-and-take attitude to community membership; increased recognition - as individuals contribute more and more to the group, there is a simulta-

neous increase in their recognition, dependent on the quality and not necessarily quantity of contribution; and sense of efficacy - the feeling of significance from having contributed something meaningful to the group. This is one of the strongest motivation for community membership.

Jo Kim *et al* have developed the following online community membership cycle [147]: Peripheral (lurker), Inbound (novice), Insider (regular), Boundary (leader), and Outbound (elder).

3.9 Multimedia systems

While there has been noticeable developments as far as computing technologies, infrastructures and system are concerned, there have also been developments in content processing. These developments happen simultaneously and influence each other: the development of tools for processing multimedia data necessitates development of mass storage equipment; the developments around mobile hardware influences and motivates the development of mobility protocols and mechanisms, and at the same time presents interesting challenges for content rendering on typically small devices. Apart from the fact that there is a growing heterogeneity of data on the information networks (i.e. images, video, and text) there are also new challenges associated with the characteristics of these new formats: efficient audio visual searching algorithms, performance requirements for real-time media, and user interaction with multimedia content. The future of the applications and services is one where there is a heterogeneity of media formats, necessitating the capability within systems to handle the different media and to transcode between the different formats.

3.10 Multimodal systems

Human interaction with computers is developing to integrate natural modalities of communication to enable wider accessibility to computing systems. Some of these modalities include voice, hand and pen-based gestures, body movements and eye-tracking [148]. Multimodal interfaces allow for a richer user interaction with applications across different tasks and environments than a uni-modal interface by supporting a selection of modalities suited to different usage contexts [149]. A key factor that motivates the implementation of multimodal interfaces is the diversity and heterogeneity of usage profiles, taking into consideration interaction preferences, abilities, cognitive styles, skills level, sensory impairments and task environments [149]. Multimodal systems handle accurate modeling and adaptation of multimodal interfaces to user's multimodal integration patterns [150]. Some of the primary differences between multimodal interfaces and

uni-modal interfaces include uncertainty of human input and simultaneous processing of input streams in multi-modal systems [151].

Some of the initial work into the development of multimodal interfaces was Bolt's *Put that there* demonstration of voice and gestures input to provide a concerted, natural modality [70]. Since then multi-modal applications have evolved to process varying multimodal integration patterns and more complex input modalities (e.g. speech and pen, speech and lip movements, and speech and 3-D gestures). These systems have also grown to handle multi-user contexts and to provide more robust and tolerant multiple modality human input processing. Different modalities are handled either as alternative inputs or as combined input. Two architectures for processing input of combined modalities are early fusion and late fusion [152]. In early fusion architectures, input processing in one modality influences the processing of other modalities. As such early fusion architectures are applicable for more coupled and synchronized modalities. Late fusion architectures on the other hand utilize individual modality processors and sequential integration processors to handle modalities that are heterogeneous and that have different temporal characteristics. The advancements in multimodal architectures have been made possible by the developments in computing hardware, natural language processing technologies and cognitive theories.

3.11 [Web] Application architectures

Application development platforms and architecture are constantly evolving to accommodate the needs within the software development domain. This is primarily due to an increase in heterogeneity, where systems must operate in a context of multiplicity of protocols, languages, technologies and programming paradigms. Application development architectures are also harnessing developments in distributed networking, resulting in the proliferation of service composition architectures implemented on the web, and in the implementation of services that incorporate the developments from the SW domain.

3.11.1 Web services

The provision of an architecture for distributed systems in heterogeneous usage contexts is one that has been pursued extensively in literature [153, 154]. The idea of distribution of functional processing in applications can range from implementing a web service component that is accessed through diverse protocols and formats over geographically spread out locations, to an

implementation within one thread on one computer. The overall web service architecture model is depicted in Figure 3.8.

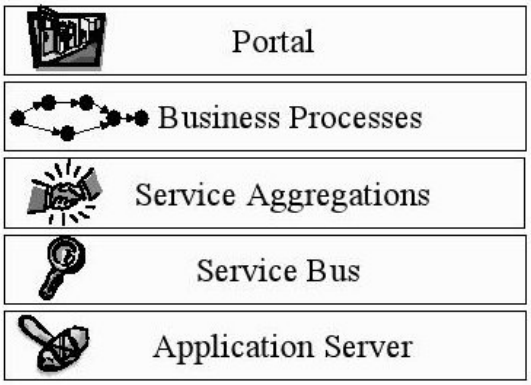


Figure 3.8: Web services architecture [154]

Key features in web services and distributed application architectures are interoperability and the implementation of standards to facilitate the discovery, invocation and communication between the different web service components. To this end, a number of standards and protocols have been defined [154]: WSDL, SOAP, and UDDI. The components in a web service environment are: the requestor, which is the component requesting the services; the web service itself; and the collection of the web services in the form of a directory. Key interactions between these components include: service *publishing*, in which the web service registers with a directory specifying its constraints and the functions it provides; the requestor then *finds* a web services that can provide a needed solution from the service directory; and finally a process of *binding* occurs in which the requestor associates with the web service and initiates an exchange to implement the needed service [154]. These interactions describe the processes in Service Oriented Architectures (SOA).

The automation of service composition through aggregation is an ideal within SOAs. Frameworks, tools, and languages are developed towards this goal, and one such language is Business Process Execution Language (BPEL). BPEL facilitates the composition of services from other small services based on the business process models. This introduces a two layer programming paradigm to web services, on one layer is the implementation of small, basic, elementary web service components, tightly coupled with the implementation environment (i.e. programming language, application and server environment) and on the other layer is the high level definition of a composite web service that is defined in a high level business process specification language [154]. This type of automated aggregation of web services is facilitated by a host of different

aggregation models: global models, service domains and coordination models, which all differ based on the key specifications within the model. Some models specify in detail the interfaces and the interaction between the different web services, and other models focus on specifying the resultant functional outcome of the composite model [154].

3.11.2 Semantic web services

SW services are an extension to the web service model, to incorporate semantic capabilities in the web service life cycle [26]. This extension is to facilitate seamless evolution of web services in conjunction with the evolution of the web to SW. The utilization of SW technologies (Section 3.1) is throughout the different processes in the stack (Fig. 3.9). DAML-S (Section 3.5.6) upper ontology forms the underlying knowledge layer within the architecture.

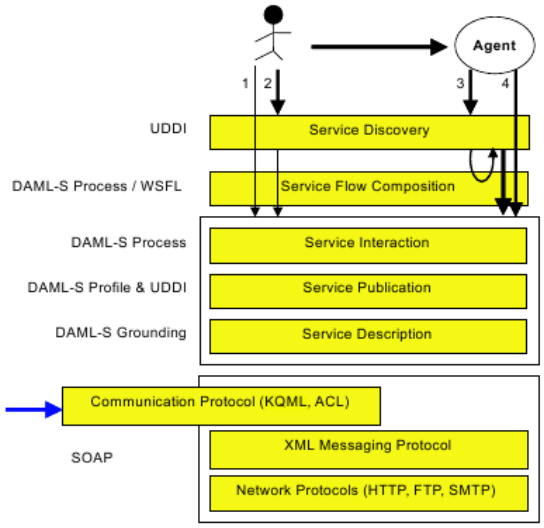


Figure 3.9: DAML-S stack [26]

3.11.3 Onto-SOA

Onto-SOA is an architecture that adds semantic machine-processability of ontologies to SOA in a manner that allows for automatic execution of service related tasks: service discovery, invocation and composition [91]. Inversely, the advantage of adding SOA to ontologies is that the new architecture inherits the loosely coupled and networked nature of SOA. The architectural style of SOA comprises processing agents, connectors and data. The processing agents encapsulate the domain expertise to execute a solution to a problem. The connectors act as messaging

channels among the agents and the services consumers. The connectors are characteristically simple, generic and application independent and only handle inter-agent communication. In order for communication to occur, the services have a well defined service description schema and vocabulary. Integrating ontologies into the SOA is achieved by encoding this schema into an ontology. One of most prolific implementations of SOA is through web services, which achieve a synergy of loosely coupled, distributed and networked service agents through the utilization of standard web transport protocols. WSDL/SOAP initially supported communication through RPC which hindered and limited the integration of services [91]. Later versions of SOAP however supported an XML document based description of web services, which still lacked semantic expressiveness. XML based web services architectures (e.g. WSDL/SOAP, REST [155]) allow for an easy integration of ontologies through encoding the XML document elements into ontology languages. Merging ontologies with SOA is a move that is being embraced by the SW community though the implementation of the OWL-S and Web Services Modeling Framework (WSMF) [156, 157]. Implementing a single framework for different tasks within different domains, and providing sufficient meta-data for service discovery necessitates a complex reasoning support, which makes the SWS cumbersome and significantly difficult to deploy [91]. This complexity is handled in Onto=>SOA by introducing a constraint on the semantic interoperability of messages, and by assuming that a service and its consumer share a common knowledge of the underlying ontology.

3.11.4 Agent architectures

Web services provide one perspective on the problem of implementing solutions that require interaction between different computation entities. The other perspective to problem solving in heterogeneous environments is through agents. Agents are different from web services in that they are [158]: problem solvers, pro-active, goal-oriented, context-aware, and autonomous. Multi-Agent Systems (MAS) tools and frameworks are becoming more popular, with a subsequent proliferation of production deployments of MAS. There is also a simultaneous growth in the standardization of protocols, and the formalization of software development methodologies for MAS environments. One such development methodology (Fig. 3.10) suggested by Nikraz *et al.* builds on the fundamental processes of planning, analysis, design and implementation to address the specific constraints of MASs, and to utilize the features that are present in MASs but not in other programming paradigms [159].

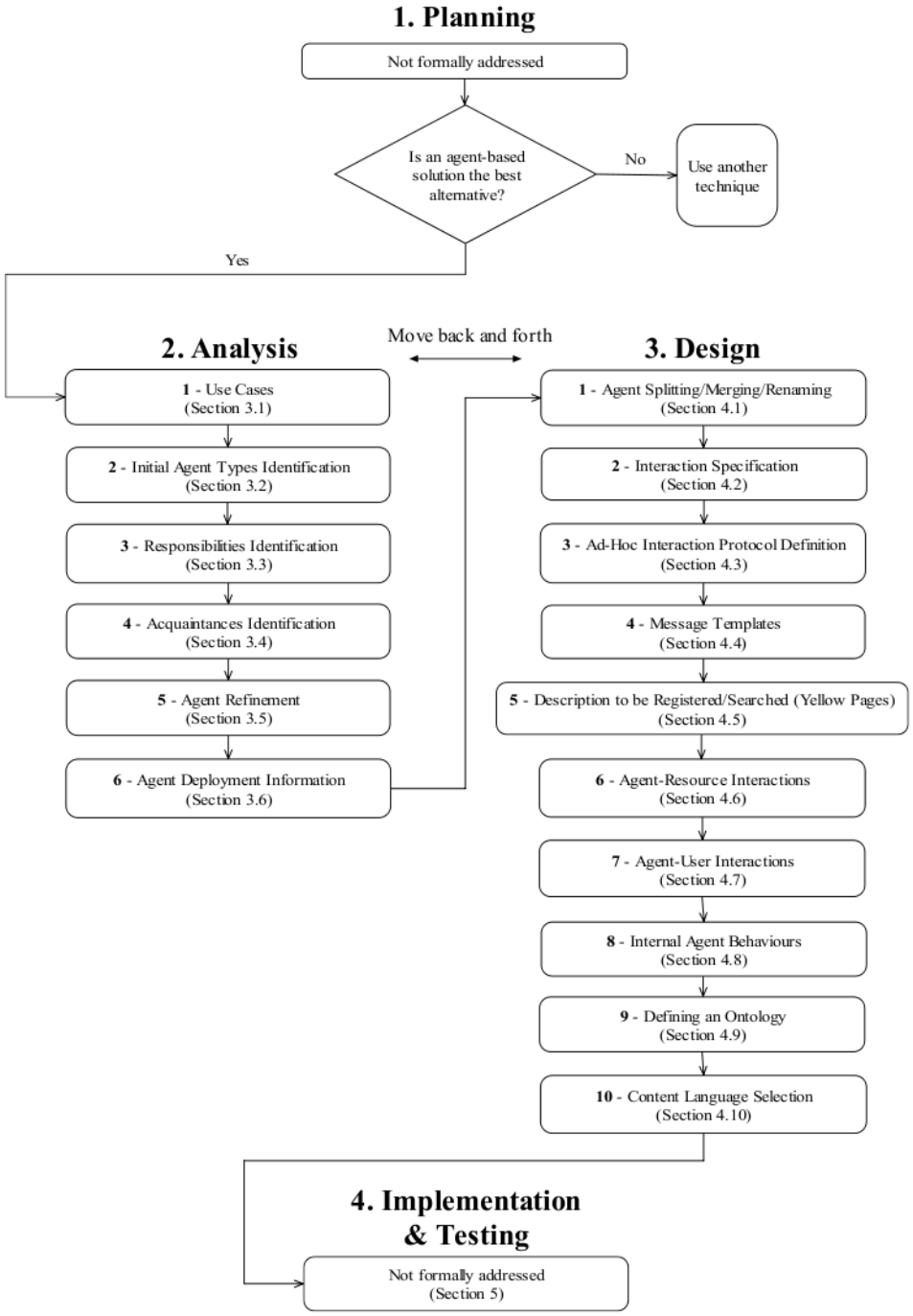


Figure 3.10: Agent software development methodology [159]

3.11.4.1 Agent standards and protocols

Common agent standards include the IEEE Foundation for Intelligent Physical Agents (FIPA) standard, and the Object Management Group (OMG) Multi Agent Facility (MAF) standard [160]. The overall goal of FIPA is the promotion of agent-based technology and of interoperability with other agents standards. This is encapsulated in twenty three specifications for different aspects of agent architectures: agent communication, agent management, agent application, and agent message transportation. While FIPA has become the primary agent architecture standard, there are other technologies and protocols in the domain of distributed services architectures:

- JSR 87 [161] - a Java Specification Request that defines the deployment and the operation of distributed agents. This specification is based on FIPA's Abstract Architecture and defines two types of entities: objects that correspond to Agent Communication Language (ACL) and Content Language (CL) elements; and interfaces that represent services for agent registration, discovery and communication.

- JINI [162] - originally developed by Sun as a network architecture of distributed systems, JINI shifts the emphasis of computing from the traditional local platform orientation to the network as the platform of operation. As such, the resources and service available on the network are utilized as if they were local. JINI provides similar functionality to RMI but offers advanced services processing features (e.g. service discovery and searching). The three main components to JINI are: the client, the server, and the lookup service. All these components have different roles in the provisioning of a distributed service architecture.

- JXTA [163] - an open protocol that specifies peer-to-peer communication between any devices on a network. This protocol consists of a number of XML messages that can be implemented through languages specific API, targeting a range of different network devices (e.g. PCs, cellphones, PDAs). Some of the protocols available in the JXTA specification are: Peer Resolver Protocol, Peer Information Protocol, Rendezvous Protocol, Peer Membership Protocol, Pipe Binding Protocol, and Endpoint Routing Protocol.

- SOAP [164] - a lightweight XML based protocol for the exchange of information in a distributed environment. The three components of the SOAP protocol are: an envelope that defines a framework for describing the contents of the message and how they are to be processed; encoding rules for application-specific data-types; and a description of how to represent remove procedure calls and responses.

A number of agent environments and platforms have been developed. The platforms differ based on factors such as: compliance to standards (in particular, the FIPA specification), the scalability of the agent platform, the application and service domain of the platform, and the implementation environment on the platform. The examples of some of these agent environments are Open Agent Architecture, JAVA Agent Development Framework, Cougaar, Grasshoper, and Aglets.

3.11.4.2 Open agent architecture

Open Agent Architecture (OAA) introduces a programming paradigm where agents provide services to an agent community that they are part of [165]. This architecture can be traced from the Object Oriented Programming (OOP) paradigm, where objects encapsulate attributes and methods to provide a service to Distributed Object architectures like Common Object Request Broker Architecture (CORBA) and Distributed Component Object Model (DCOM). These architectures allowed objects to reside on different machines and to communicate using the same interface definition languages. In the Distributed Object architectures, objects are tightly coupled and positioned within the implementation of a specific application. The OAA supports objects/agents that are less coupled and for building a community of agents that do not have the pre-knowledge of implemented sub-routines and parameters in other agents [165]. Communication within a community of agents is managed by the Facilitator agent, which handles the decision making as far as which agents provide the necessary services for completing a request. Agents in OAA are wrappers around programs that are formed in a context of an organic community of processing components. Agents communicate with each other using a high-level logic based language, and agents also have a built-in ability to monitor the events in their world and to respond accordingly.

3.11.4.3 Java Agent DEvelopment framework

Java Agent DEvelopment (JADE) framework and is a middleware and a software framework for the development of agent based applications [166]. JADE is developed by Telecom Italia Lab (TILAB) and is written entirely in Java. JADE implements the FIPA agent standard and provides a platform that can be distributed across different platforms and hosts. Communication within JADE is based on JAVA RMI for intra-platform interactions (i.e. interactions where the agents reside on the same platform) and on FIPA specified protocols for inter-platform communication (i.e. interactions where the agents are on different platforms) [166].

A derivative project from JADE is the Lightweight Extensible Agent Platform (LEAP). LEAP is intended to provide a FIPA compliant agent platform for any JAVA enabled, connected device

[167]. This is achieved through the implementation of a new kernel for the JADE platform, which handles the deployment of legacy agents on the small mobile devices.

3.11.4.4 Grasshoper

Grasshoper is a FIPA and OMG Mobile Agent System Interoperability Facility (MASIF) specifications compliant mobile agent platform. Grasshoper is both a development environment and also a run-time platform for agents in a manner that achieves a synergy out of the traditional client/server architectures and mobile agent technology [168]. The following are the components of a Grasshoper agent system [168]: a *Distributed Agent Environment* (DAE); a *communication service* which facilitates interactions between different platform components; *registration service* that allows the agents to know about each other; *management service* for monitoring and controlling the agents; a *security service*; and a *persistence service* which allows agents to be placed on a storage medium for recovery after a system failure.

3.11.4.5 Aglets

Aglet is an agent platform for deployment of mobile agents. Aglet provides libraries that can be used to integrate agents into applications, and a platform server called *Tahiti*. Within the Aglet platform, the key components are [169]: the *aglet*, which is essentially a JAVA object that is both autonomous and reactive, and that gets executed on aglet enabled computers; a *proxy*, which is a representation and an abstraction of the aglet; a *context*, which is the execution space of an aglet; a *message* that is exchanged between the different communicating aglets; a *future reply* which defines a response handler for messages communicated to the aglet; and an *identifier* which is a unique identification bound to each agent.

3.12 Related technologies and tools

Tools and technologies developed for knowledge networking are constantly evolving and being adapted to integrate new insights from the theoretical discourse. For each phase in knowledge processing, there are tools that have been developed to facilitate the undertaking of associated activities. For designing taxonomies and ontologies, there are tools that abstract from the specific KR language, provide visual representation of the underlying knowledge and facilitate navigation around the ontology (e.g. Protege [170], and CMapTools [171]). There are also Application Programming Interfaces (APIs) that can be utilized to provide semantic capabilities in applica-

tions (e.g. JENA [172], Redland [173], Sesame, RDF API for PHP (RAP) [174], and JRDF). The various APIs differ based on: KR languages that are handled by the API; the underlying model of persistence that is provided in the API (e.g. external database, in-memory); the inference capabilities that the API provides; the level of scalability with regards to the number of triples and users of the knowledge base; the diversity of language bindings that the API accepts; the level of portability and interoperability with other APIs; and finally the extent that the API provides support for robustness and reliability.

Once the ontologies have been designed, tools are available that facilitate the population of the ontologies with domain knowledge. This functionality is typically bundled with most of the ontology editors available, but there is a growing number of automatic knowledge acquisition tools that use Natural Language Processing (NLP) or data mining on the web to populate the ontologies. Once the ontologies have been populated, they need to be stored, and for that there are ontology storage systems (3store - a MySQL based triple store, RDFStore, and SDB). The knowledge storage tools typically integrates with or provide facilities for knowledge querying using SPARQL (D2R Server, SPARQLer, and Talis Platform) and for performing inference on the underlying ontologies using any of the available reasoners and inference engines (FaCT++, KAON2, RacerPro). At a global Internet level, there are services that index and allow for searching on the plethora of ontologies that are available online (Swoogle, Falcon, and Sindice). Accessing and viewing information in an ontology is made easy through tools that provide a graphical representation of the underlying ontologies (BrowseRDF, mSpace, OWLsite, and Ripple).

The increasing proliferation of semantic services on the web and in the corporate world means more tools are being developed to acquire and codify data, and to make knowledge machine-readable and processable. The existence of these various KR tools and languages means that large amounts of information is stored in data formats that are not immediately interoperable and exchangeable. This results in the problem of KR heterogeneity which leads to inefficient management of knowledge due to duplication and redundancy. One protocol that has been developed to mitigate against the heterogeneity of KR formalisms and languages is Open Knowledge Base Connectivity (OKBC), which is a successor of the Generic Frame Protocol (GFP) [175]. It achieves this by exposing a web service interface, which is accessed over the HTTP transport layer, to the various Knowledge Representation Systems (KRS) and to use different KR tools in a manner that is analogous to Object DataBase Connectivity (ODBC).

3.13 Conclusion

Developments as far as ICT is concerned have brought substantial changes in the use and management of knowledge in societies. These developments include increased processing power of computers, increased data storage capacities and the wide-spread proliferation of computer networks. These advancements have been accompanied by corresponding developments in standards, protocols, application architectures and web service specifications. The overall result is a dynamic global network of knowledge that is easily accessible to an increasing number of people.

Chapter 4

Knowledge Networking Situated

> ”My people perish for lack of knowledge” (Hosea 6:4)

The current state of the art in knowledge networking technologies presents numerous possibilities and benefits for human societies. As highlighted in Section 2.1.3 these benefits depend on numerous ethnographic, technological, sociological and philosophical factors. In exploring the implementation of knowledge networking technologies for ICT4D, it is also important to ground the research in practice, in order to understand the possible effects of these factors. This chapter situates the work undertaken within a specific geographic and cultural context, by describing the community that hosts the field site for this research.

Building up from the discussions on the nature of IK, we explore the problem area of knowledge modeling for a specific community, highlighting the requirements and constraints that feed into the development of a community knowledge platform. The culture-sensitivity aspect of the development of a knowledge platform is presented through a profiling of the local knowledge system dynamics which would need to be modeled on the knowledge platform.

4.1 Dwesa

The general context in which this research is undertaken is that of ICT4D. The specific research site for the project is a deep rural and marginalized community of Dwesa in South Africa. This community is characteristic of many third world rural realities in which ICT4D projects are undertaken. Situating the research in a specific area allows for an extensive and close study of the community and a situated determination of the direct needs and requirements of the community. The objective is still that the solutions developed and implemented in this specific context will be implementable in other similarly marginalized and rural communities.

4.1.1 An overview of the community

Dwesa is located on the Wild Coast in the Eastern Cape province of South Africa. It is situated in the former homeland of Transkei [60]. The community has approximately 15,000 inhabitants who are distributed into 2000 households. Dwesa is traditionally a subsistence farming community, and to a large extent the community depends on their land for their livelihood [176].

This community is the site of the second successful land reclamation case in South Africa, which saw the land and in particular the nature reserve in the area being given back to the community. A Trust was established to oversee the activities of the reserve and to represent the community in the management and the running of the reserve.

As a result of the presence of the nature reserve, the area has a rich potential for eco-tourism. Due to the rich cultural heritage in the community, it also has potential for cultural tourism. Finally there are arts and crafts entrepreneurs in Dwesa who serve the local markets with their artifacts. These entrepreneurs are mostly organized as groups.

4.1.2 Infrastructural constraints

There are extensive infrastructural constraints in Dwesa. The first is an underdeveloped road network. Dwesa is approximately an hour of travelling by car on a gravel road away from the nearest town and transportation between the village and town is very sporadic.

The availability of electricity is limited to a few places in the community: the office at the nature reserve, some of the schools, the clinic, a few shops, and a few households. This has direct influence on the availability and utilization of technology in the community. There is a high prevalence of cellphones in the community and the results from a baseline study that was undertaken in 2008 indicate that the majority of people have access to a cellphone within the immediate circle of friends or relatives. The schools and shops where there is electricity have adopted a working business model and an arrangement wherein people bring their cellphones to be charged at a small fee. There have been previous attempts by government and NGOs to install solar panels in the schools for electricity but most of these are not utilized anymore due to theft and vandalism.

The telecommunication infrastructure is also very limited. There are very few fixed telephone lines in the community, and the few public pay phones that had been installed in the community have been vandalized and damaged. There is an availability of GSM network in the community provided through the country's mobile operators Vodacom and MTN. The GPRS and EDGE connectivity is sometimes erratic and not always available.

4.1.3 ICT4D activities in Dwesa

The formal establishment of Dwesa as a research site for the ICT4D intervention came as a result of a link with previous research activities in the region by researchers from the Department of Anthropology at Rhodes University. The project undertaken in Dwesa was initiated in 2005, with the preliminary discussions being held with the stakeholders *in situ* to assess its feasibility. Research activities began towards the end of 2005. The project in Dwesa is a joint venture between Rhodes University, situated in Grahamstown and the University of Fort Hare situated in Alice. These two universities represent a strategic alliance in the execution of an ICT4D project. On the one hand, the University of Fort Hare is a previously black university (under the apartheid system) and as such is closely situated to the context in Dwesa, as most of the students at the university come from similar geographical and socio-economic contexts. On the other hand Rhodes University is a previously white university and has more resources as far as the technical and the organizational aspects of the project are concerned. Due to the geographical remoteness of Dwesa and its distance from both the University of Fort Hare and Rhodes University, project activities are normally arranged into one-week long monthly trips. This was the initial arrangement until key individuals were identified within the community who would be directly responsible for specific project activities.

The initial objectives were to develop a prototype of an eCommerce platform for the arts and crafts entrepreneurs in the community, and also for the possible exploration of micro-tourism potential in the area. The introduction of the eCommerce aspect to the economic activities in Dwesa was aimed at activating the community towards greater involvement in economic activities in the region, but also at opening up the market base to incorporate wider international customers. An eCommerce portal was developed in direct interaction with the local arts and crafts entrepreneurs to integrate their specific needs and requirements into the platform. The portal was developed around a metaphor of a mall (i.e. an eMall) in which the different sellers have a store that they manage and that they are responsible for. The three different levels of users on the platform therefore are: the eMall administrator, the shop owners (i.e. store administrators) and the customers who visit the portal to purchase arts and crafts artifacts. Two distinct interfaces have been developed for the eCommerce portal: the customers interface which has been developed to the expected standard for the international buyers; and the shop owner's interface which has been developed to be accessible to the local shop administrators as far as localization into the local language is concerned, and through the integration of relevant cultural markers[1] on the portal.

Besides the deployment of the eCommerce portal in Dwesa, a number of other projects have

[1]Cultural markers are UI localization features (e.g. colours, shapes) that are specific to a particular culture [177]

been undertaken within the context of facilitating the implementation of the ICT4D intervention:

1. A key component of a successful ICT infrastructure in Dwesa is the network. A project was undertaken to setup a WiMAX based, local loop to connect the different points of presence in Dwesa. At the moment, the schools are the key points of Internet access as they are some of the few places within the community that have electricity. Another project related to the networking in Dwesa was exploring the back-haul connectivity options to the Internet and their associated costs and benefits. The resultant back-haul connectivity is provided via a VSAT satellite link. There is a current project addressing the issue of ensuring robustness and resilience on the network, through exploring options for redundancy, remote network monitoring and management, and fault-tolerance mechanisms within the network.

2. The introduction of ICT in Dwesa was done at a time when there was minimal computing literacy in the area. A project was undertaken specifically for running training sessions in the community, exploring the best practices as far as learning methodology, eLearning and blended learning approaches are concerned. The result of this is a computer training strategy aimed at the whole community, wherein the University based researchers train a few members of the community who then train the rest of the community, in an iterative manner.

3. The sustainability of the project is a key objective that has been explored from a financial, technical, social and cultural point of view. As a result research is under way to develop cost-sharing systems and models for the community wherein the different members can contribute in various ways to the upkeep of the deployed infrastructure. From the technical sustainability point of view, researchers are involved with developing a help-desk system that will be utilized to capture and disseminate the problem solving knowledge that has been accumulated in Dwesa. Other research is associated with studying and implementing the culture specific requirements as far as the HCI components of the deployed systems are concerned [33].

4. The key benefit of deploying the computing network in Dwesa is the ability to provide different value-added, relevant services for the people in Dwesa. Currently there are projects under way exploring the development and implementation of eHealth services, eGovernment services, and eJudiciary service for the community. An internal Voice Over Internet Protocol (VOIP) communication network has been set up in the Dwesa.

The initial deployment of the services in Dwesa was centralized and predominantly web-based. Some of the above mentioned service portals are accessed primarily through a web interface to a server deployed in one of the schools in Dwesa. This initial deployment paved a way for a subsequent evolution in the project discussed in Section 4.4.

The manner in which the project has been structured adheres closely to participatory and multi-disciplinary action research. The key realization made at the conception of the project is that the problems that are being addressed in these types of projects (i.e. ICT4D projects) are multi-faceted in nature, and the associated solutions are intrinsically multi-disciplinary. The research group on this project therefore consists of researchers from a number of disciplines: computer science, information systems, communication and marketing, journalism, sociology, education and anthropology. The solutions proposed in this project therefore tend to encompass factors that address problems from the different perspectives represented in the research group.

The relationship of the community members and the project is one in which all the stakeholders are key participants in the project, in a manner that is based on ethnography as a research methodology. This structuring of the project activities has also motivated the exploration of the project from the perspective of a *Living Lab* [178], in which the users, the researchers and the different stakeholders are part of an organic, innovating, problem solving system. The need for innovation and for responding to the prevalent environmental conditions has meant that the overall undertaking of the project is done in an iterative manner in which the current problems are analyzed, a solution developed and implemented, a reflection and observation of the effects of the implemented solution is undertaken, and where corrective measures are undertaken to respond to the identified problems, thus beginning an iteration through the cycle again.

4.2 Supporting ICT infrastructure

The deployment of the different services in Dwesa is supported by the ICT infrastructure that has been deployed in the region. Figure 4.1 is a graphical depiction of the network that has been deployed in Dwesa. The four schools that are connected to the network are: Mpume Junior Secondary School (JSS), Ngwane JSS, Mtokwane JSS and Nondobo JSS and the last school where a computer lab has been set up, but which is yet to be connected to the network is Nqabara High School (not shown on the network diagram in Fig. 4.1).

The deployed networking infrastructure consists of the following:

1. A Very Small Aperture Terminal (VSAT) satellite provides the back-haul connectivity to the Internet. This is located at Mpume JSS. The connection to the Internet is shared be-

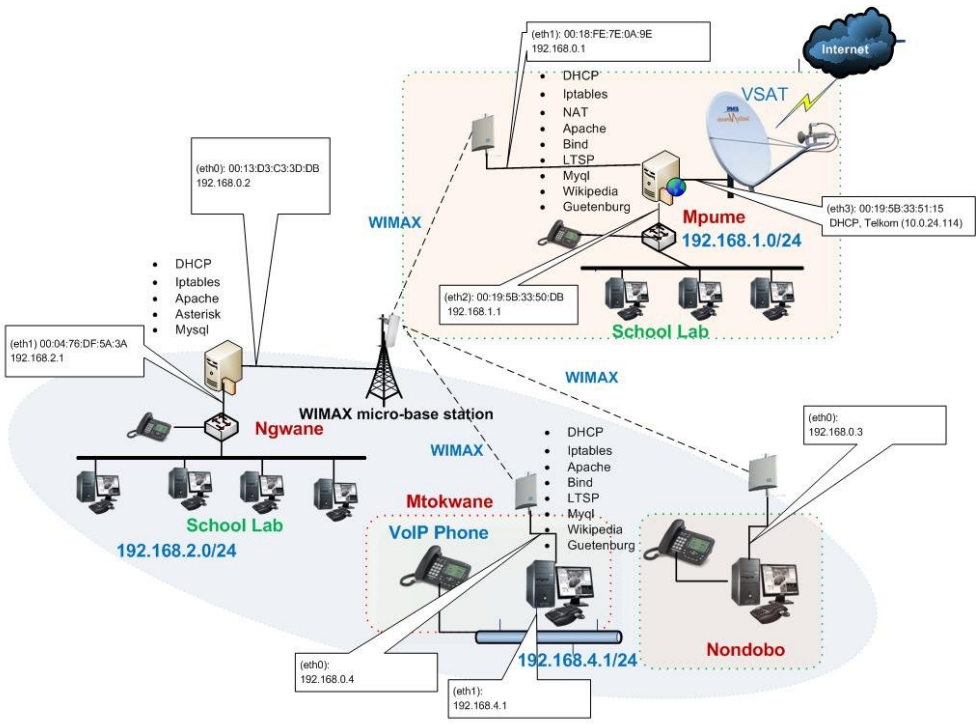


Figure 4.1: Dwesa network [179]

tween the different points of access (i.e. currently the schools) via a local-loop wireless network.

2. WiMAX has been deployed as the wireless technology for the local loop within the Dwesa network. The WiMAX micro-base station is situated at Ngwane JSS and communicates with the Customer Premises Equipment (CPEs) that have been installed at the other three schools. WiMAX provides an adequate connection technology due to the fact that it can operate in Non Line Of Sight (NLOS) environments which are prevalent in Dwesa between the different schools.

3. The communication between the different schools is through a (Point-to-Point Protocol over Ethernet) PPPoE tunnel that provides encryption of the traffic on the wireless channel. Each school is therefore connected to a router that establishes the tunnel between the different wireless points and the Access Concentrator that is located at Mpume JSS.

4. A VOIP server based on the Asterisk PBX is located at Ngwane JSS and provides the communication service between the different schools which have a VOIP phone each. Access to the VOIP server is also available to the network users through soft-phones that

are available on the lab machines.

5. A web server based on Apache has been deployed at the different schools to support requests for the offline content repositories and web based services. An offline version of Wikipedia is provided on the servers as well as the books in the public domain through project Gutenberg. A service application that is made available on servers is SchoolTool which provides services for the management and organization of school activities.

6. The labs deployed range in size from three computers at Nondobo JSS to 15 computers at Ngwane JSS. The computers labs are configured in a thin-client architecture in which a few powerful servers have been deployed and the clients boot from them. The labs are running completely Free and Open Source Software (FOSS), with Edubuntu as the specific Linux distribution used. FOSS allows for the affordability that is suitable for the poor rural communities and also the flexibility that allows for source level customization of the software.

4.3 Supporting societal structures

While ICT provides the crucial infrastructure for the realization of the intervention in Dwesa, it is necessary that relevant societal structures are in place in support of the intervention. One of the advantages of this arrangement is that they allow the project to have buy-in from the community, and to be ingrained within its life system, thus ensuring a long-term sustainability of the intervention. The following are the structures whose role we found important: the local community governance through the headman, the schools and the School Governing Bodies (SGB), and lastly the community Trust that has been set up to represent the community with regard to the activities associated with the nature reserve.

1. The headman - the role of the headman in the life of a community such as Dwesa is paramount. He approves and permits the undertaking of externally initiated activities within the community. The buy-in from the headman has meant that the project gets approval within Dwesa and the necessary permission to undertake the research activities within the area.

2. The schools and the SGBs - the support of the schools and the SGBs paved the way to establishing computing labs in the campuses. Within the community, the schools are key institutions and centres of learning and therefore associating the project with the schools

ensured an alliance wherein the schools are able to undertake some of the responsibilities as far as training the community is concerned [32].

3. The Trust - the Trust represents the community in the management of the nature reserve but also in planning the overall developmental activities around Dwesa. The involvement and support of the Trust has ensured that the project is undertaken within a broader development plan that they have drawn up.

4.4 Multi-functional, distributed communication platform

The successful deployment of the initial phase of the project paved the way to an alternative conceptualization and revision of the intervention in Dwesa. The initial phase of the project was centered around the eCommerce platform and the establishment of the associated infrastructure to support the effective utilization of the portal. The subsequent realization of the platform is as a multi-functional, multi-service, distributed communication platform for the local community. This integrates into the platform the flexibility to deploy a plethora of community based services in a manner that is distributed across the different points of access in the community. One of the key features of this new architecture and platform is that it is an inherently multi-service platform. The provision of eCommerce, eGovernment, eHealth and eJudiciary services would be built in an integrated manner on the platform as opposed to as independent service portals.

4.5 Knowledge dynamics in Dwesa

Knowledge system dynamics differ from one community to another based on numerous factors. Some of the factors that characterize different communities and that directly influence the usage and exchange of knowledge include: the levels of social stratification and the relationship between the different strata, the general power relation dynamics, and the extent of communal orientation within the community. These factors have a direct bearing on the realization of knowledge systems in communities and form part of the critical points of departure in the implementation of knowledge systems for different communities. For example, a knowledge system developed for a fairly egalitarian community where there is a cultural sense of openness and sharing would implement far fewer features around confidentiality and privacy than for a society with opposite cultural orientation.

This section therefore highlights the different factors and the associated dynamics that are

specific to the Dwesa community, that have direct bearing on the realization of a knowledge platform for that community. These factors are characteristic of other similarly rural, marginalized, African communities.

4.5.1 Types of knowledge in Dwesa

Section 2.6.2 discussed the different conceptualizations of knowledge from philosophical and sociological perspectives. Within that discussion, Nonaka's SECI framework of knowledge was highlighted in which he makes a distinction between two types of knowledge: tacit knowledge and explicit knowledge. Within the SECI framework, a process of externalization provides access to the internal tacit knowledge through explicit expression by the individual. In understanding the specific knowledge dynamics in Dwesa, the focus is on explicit knowledge as this is the knowledge that is accessible and codifiable.

The categorization of knowledge in this sections is based on a superficial aggregation of similar types of knowledge that exhibit similar characteristics, and not on any ontological or epistemological considerations. The differences in the categories of knowledge highlight specific requirements for the knowledge platform. The different categories of knowledge have been elucidated through discussions with the Dwesa community members, individuals from the Xhosa culture and other South African cultures. We present these different types of knowledge in a graph we have developed, called the OSCA knowledge matrix, which is based on the mapping of different knowledge types along the dimensions of Ownership, Social advantage, Confidentiality and Accessibility (Fig. 4.2):

1. Common knowledge - This is the knowledge that is common to every human being and not particular to any locality or a group of people. This knowledge once codified, is accessible in the public domain and every individual is entitled to access, use and benefit from it.

2. Shared-cultural knowledge - This is the knowledge that is specific to a cultural group. This knowledge is identifiable with the culture and can be assumed to be owned by that cultural group. Examples of this knowledge include folktales, stories, proverbs and riddles. This knowledge also includes arts and crafts patterns and artifacts. The issue of ownership of this kind of knowledge is a contentious one, and at the core of it is the concern for the commercial exploitation that sometimes occurs. While this knowledge can be assumed to be owned by a specific ethnic and cultural group, it is however accessible to the public.

3. Specific group knowledge - Within the Xhosa culture (i.e. the culture of the people in

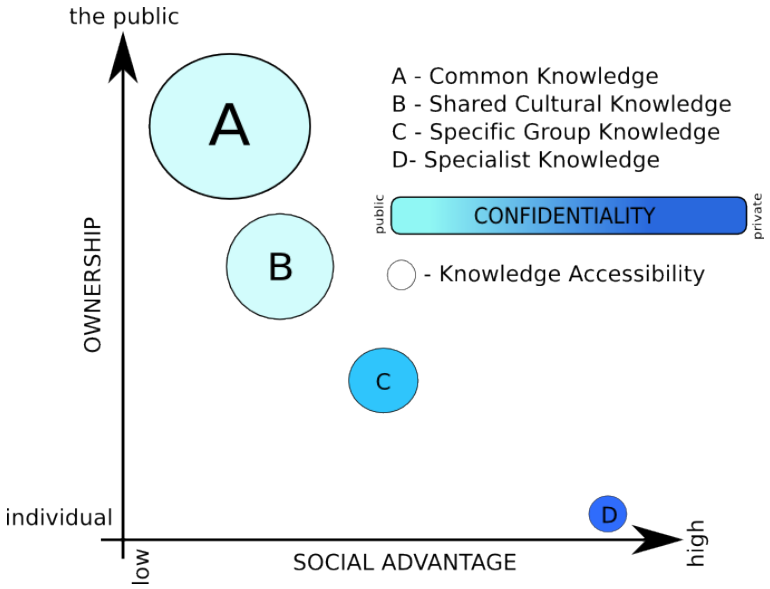


Figure 4.2: The OSCA knowledge matrix

Dwesa), and in fact within other South African cultures, there is knowledge that is associated with different groups. These groups form around age, gender, social status, or ethnicity. The kind of knowledge in this category is not only owned by the specific group, but it is also private and confidential (at varying levels) to that group. Examples of this kind of knowledge include, the secret knowledge of the *amadoda* (the men, vs the boys) in the Xhosa culture, or *banna* in the Sesotho culture. Access to this kind of knowledge is normally associated with an initiation process into the group, which in this particular case is through the initiation schools. This kind of knowledge is exchanged and communicated within the confines of the group. This knowledge is associated with key social and power dynamics and in a sense access to this knowledge (or membership of the group through the initiation process) gives an individual certain social advantages. An example from the Xhosa culture is that the males who have not been to *the bush* (the initiation school) have lesser roles to play in family ceremonies, are held in lesser regard as *amankwenkwe* (the boys) within the community.

4. Specialist knowledge - An example of this type of knowledge is the medical knowledge of the *amagqirha* (the traditional healers) in the Xhosa culture. This knowledge is very confidential. It is owned by a specific individual or a close knit group of individuals. This knowledge gives the owners an advantage within their community or society. This social advantage can be in the form of the prestige that the person gets in the community, or the

direct competitive advantage from the point of view of the commercial benefits of being the healer in the community. Access to this kind of knowledge is very strict and only a few people (e.g. a protege, an heir) have access to this knowledge.

A key factor in the development of a knowledge platform for a community is the initial ability to have access to the different knowledge for encapsulation on the platform. While it is unrealistic to expect certain types of explicit knowledge to be exchanged on the knowledge platform (e.g. the specialist knowledge), the different characteristics of the knowledge categories enunciated above will inform the design and the development of the knowledge platform for Dwesa (Chapter 6). The mechanisms for catering for the specific needs of each category will be integrated into the platform. The responsibility to tag appropriately and to use the different categories of knowledge on the platform rests with the users.

4.5.2 Role of knowledge

The role of knowledge in society is more than making facts and information available. Within the SECI framework of knowledge creation, the processes involved have a crucial social component to them, in which the knowledge processing occurs in a context of relationships. For example, the socialization process involves the exchange of tacit knowledge out of close interaction between individuals, and both externalization and combination are under-girded by a relational dynamic.

Within the discourse on folklore and culture, the role of knowledge (e.g. folktales, proverbs) is associated with entertaining, informing, correcting, encouraging, and warning. These are all processes that occur within the context of the relationship between the knowledge exchanging individuals.

This intrinsically relational and social aspect of knowledge exchange processes is a crucial consideration that must inform the development of the knowledge platform. It is an aspect that differentiates one deployment from another due to the differences in the underlying social dynamics within communities.

4.6 Knowledge platform and eServices development

The knowledge platform envisaged for deployment in Dwesa plays a role not only in providing knowledge services for the community, but also in terms of supporting other service development projects that are undertaken in Dwesa. In particular the knowledge platform provides a

deployment layer for the eServices (Section 4.1.3) that are developed for Dwesa on the revised multi-function distributed communication platform (Section 4.4).

The approach taken in the realization of the knowledge platform embodies the knowledge-centric paradigm for ICT4D interventions. In this paradigm the core of the developed solutions is processing the underlying local knowledge in the community, and subsequently developing specific eServices for the various community activities (e.g. learning, health, and farming).

4.7 Conclusion

This research is situated within a context of an ICT4D intervention undertaken in Dwesa. The Dwesa community is representative of many rural, marginalized communities in Africa (in terms of infrastructural constraints, IK system dynamics, and socio-technical profiles). Within this community, the factors associated with the local knowledge systems that have to be taken into consideration in developing a community knowledge platform include the different types of knowledge in the community and the associated issues of ownership, confidentiality, accessibility and the resultant social advantage; and the role of information in these communities.

This research builds on the activities that have already been undertaken in Dwesa with regard to the networking infrastructure deployed, the rapport that has been established with the community, the literacy training that has been undertaken in the community and the eServices that have been deployed.

This chapter has positioned the undertaken research within a specific community context. This provides an initial building block for the investigation and the conceptualization of an architectural framework (Chapter 5) and subsequently the realized knowledge platform (Chapter 6).

Chapter 5

Architecting The Knowledge Platform

> ”Each pattern describes a problem over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice” (Christopher Alexander)

The previous chapters have been towards the development of an architecture for the provisioning of knowledge based services for marginalized communities. This research explores a different way of envisioning ICT4D interventions, and also a new way of structuring and layering knowledge based service platforms to achieve the following goals:

1. Provision of an end-user device agnostic interface to the underlying knowledge, making allocation for handling heterogeneous device requests.
2. Allowance for varied interaction modalities with the users.
3. A context-sensitive and ethnocentric knowledge platform - through encapsulation of local knowledge and the emulation of local knowledge system dynamics. This is towards achieving a seamless mobility between knowledge exchange in the real world and knowledge exchange in the virtual space (i.e. on the knowledge platform).
4. Handling of multimedia knowledge.
5. A future-proof platform that embraces the developments in knowledge engineering technologies and SOAs.

The realization of this architecture is a multi-faceted process that addresses the requirements that have been discussed in the previous chapters. The architecture is named PIASK, which is an

acronym for the different tiers that have been identified as necessary in the implementation of the knowledge based services platform (presentation, interaction, access, social networking and knowledge base). This chapter discusses PIASK as an architecture and a new way of conceptualizing eServices delivery platforms that have context-sensitivity and ethnocentricity as their design goals. We, however, first discuss a key facet in the architecture, which is the underlying knowledge that is made available in the PIASK based platform, called KnowNet. The ontologies discussed in this chapter (Section 5.1) are particularly for the realization of a knowledge platform in Dwesa, and as such encapsulate the knowledge for the implementation of specific eServices (e.g. eHealth, eCommerce, eLearning).

5.1 The underlying knowledge

The encapsulation of knowledge is performed at two levels. At one level, the coding of the logic and the implementation of procedural functions represents the internal knowledge about the functioning of the platform. At the core of the platform is another level at which knowledge is explicitly encapsulated in ontologies and folksonomies. This occurs at the knowledge base layer (see Section 5.3 for a detailed discussion of the PIASK architecture). Ontologies provide the formal and explicit specification of conceptualization of knowledge (Section 3.5). Ontologies are domain-specific and narrowly defined for the particular area of knowledge. On the other hand, folksonomies allow for a freely structured, widely distributed, bottom-up categorization of knowledge entities. Folksonomies represent and reveal the widespread conceptualization and the emergent structure of knowledge.

A number of ontologies have been developed, specifically contextualized to Dwesa. These ontologies represent the following key areas of targeted eServices deployment as part of the larger project undertaken with this community (Section 4.1.3):

1. eHealth services (health ontology - Section 5.1.1) - The area of health service delivery in Dwesa is very under-resourced; at the same time, and more significantly, the IK within the community with regards to health practices is not explicitly codified. Through the provisioning of eHealth services, a greater effectiveness in health service delivery might be achieved together with a level of preservation of the long held traditional knowledge around health.

2. eCommerce services (commerce ontology - Section 5.1.2) - One of the perspectives from which under-development can be formalized is the economic paradigm, and so within

this view, interventions that activate and mobilize economic and entrepreneurial activities should be undertaken. The development of eCommerce services in Dwesa is positioned within a space of great economic potential in the region, through eco-tourism and micro-tourism initiatives. These services can also provide the needed efficiency and access to wider markets for the already existing arts and crafts initiatives.

3. One of the underlying objectives of the project undertaken in Dwesa is the activation of a community towards a knowledge society, and to that end the goal is for the provisioning of eServices to extend to all areas of community life. The culture (Xhosa) ontology (Section 5.1.3) allows for the encapsulation of the culture specific artifacts and the cultural expressions of the community. Since this community is traditionally a subsistence farming community, a farming (agriculture) ontology (Section 5.1.4) also becomes adequately positioned to provide the necessary technological support for farming.

The design of these ontologies and the discussion of the other supporting ontologies that have been implemented and utilized follows. The graphical and RDF/OWL specification of all the ontologies are in Appendix A.

5.1.1 Health ontology

In our experience, knowledge about the traditional medicines and health considerations is predominantly encapsulated within societies as tacit knowledge. This ontology codifies the indigenous health practices and knowledge in a manner that allows for an easy use of the knowledge, as well as an easy integration with other external health ontologies. The core concepts encapsulated in this ontology are medical conditions and medicines. For each of these, the knowledge contained in the ontology represents both what would be termed traditional medicine and Western medicine.

Also in our experience, there is a definite distinction made between traditional medicine and Western medicine in rural communities. The attitudes and the beliefs around medical conditions in these communities are fed by the overall worldviews that are prevalent. Akin to the experiences of doctors in other parts of rural Africa (as well as of those working among Aborigines in Australia, for example [180]), the role and the influence that animism plays becomes immediately evident. Therefore, while the different medical conditions would be identifiable both within Western medical practice and traditional medical practice, the diagnosis and the resultant treatment become very different. This is one of the considerations for flexibility that is taken into account in the structuring of this ontology.

In order to facilitate the interoperability and consistency with other medical knowledge repositories, the Anatomical Therapeutic Chemical (ATC) Classification System[1] has been used to inform the structuring of the ontology (Fig. 5.1). The ATC is maintained by the WHO to facilitate the classification of drugs.

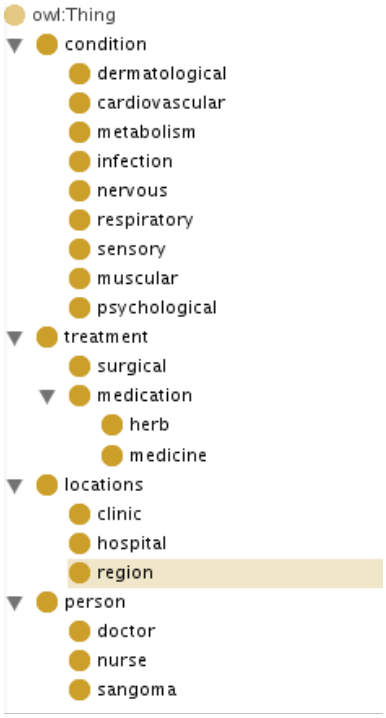


Figure 5.1: The health ontology

5.1.2 Commerce ontology

This ontology defines the body of knowledge surrounding commerce as it pertains to the local economic activities. It defines the classes and the associated properties of products, the different roles of users (e.g. seller, owner, buyer) of the eCommerce platform, and the relations between the different entities. The top level entities defined as sub-classes (i.e. *rdf:subClassOf*) of the *owl:Thing* class are *dwesa:Item*, *dwesa:Location*, and *dwesa:Person*. The *dwesa:Item* class is further sub-classed into:

1. *dwesa:event*, events which customers can attend. For example, music and dance shows.
2. *dwesa:product*, items and products for sale. These include arts and crafts, and cultural artifacts.

[1] http://www.whocc.no/atcddd/

3. *dwesa:service*, services that are provided for the customers. Accommodation services and leisure services are examples of instances of this class.

This ontology (Fig. 5.2) is developed to facilitate the provisioning of a basic eCommerce service in Dwesa for the entrepreneurs.

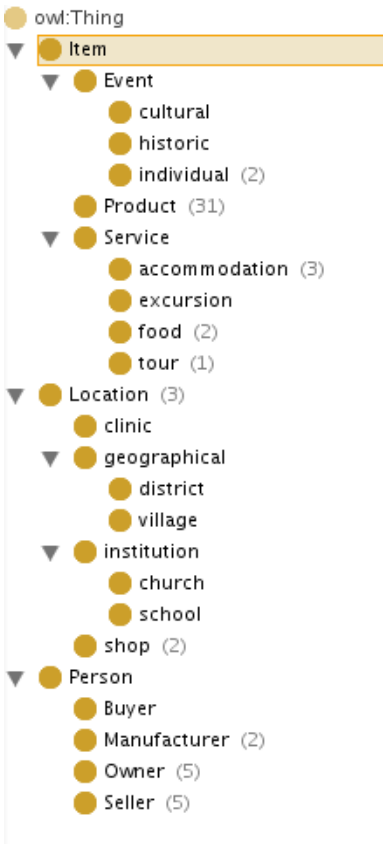


Figure 5.2: The commerce ontology

5.1.3 Xhosa ontology

Most of the culture of a community is implicitly encapsulated in its practices, stories, and traditions (see Section 2.5 on IK). Codifying this type of knowledge becomes all the more difficult owing to the diversity of the expressions of a culture. At the base of this ontology is the encapsulation of the stories, the riddles, the poems and the different practices of the culture (Fig. 5.3).

Figure 5.3: The culture ontology

5.1.4 Agriculture ontology

Rural and marginalized communities possess practical knowledge in the area of agriculture because the majority of these communities rely on subsistence farming for a living. The agriculture ontology (Fig. 5.4) embodies the knowledge in the agricultural sector of community life

5.1.5 Supporting ontologies

Additional knowledge is encapsulated in the following supplementary ontologies that are available for use in the platform.

1. At the social networking layer, the Friend Of A Friend (FOAF) specification provides the ability to model the profile of the users of the platform and also to specify the relationships (i.e *foaf:knows*) between the users. This specification is used in an ontology for the users of the platform.

2. The PIASK system ontology (Fig. 5.5) used within the local knowledge agent (Section 6.6.3).

3. Project Gutenberg ontology which contains the information about the books that are in the public domain [181]. Making these books available on the knowledge platform is of strategic importance as far as the schools and education centres in the rural marginalized areas are concerned: owing to the infrastructural constraints, access to written materials is very limited in the schools and most of the schools in the area do not have access to any

Figure 5.4: The agriculture ontology

library service.

The project Gutenberg books are currently available as text files or audio books that are accessible via a web browser on the Internet or on a local disk. These books are accompanied by an index file which identifies the author, the title of the book and the name of the folder and subsequently the file that is associated with the book. The project Gutenberg ontology encapsulates this information as predicates associated with objects of type *Book* (i.e. *rdf:type dwesa:Book*) which is a domain of the predicates (Table 5.1). Note that the ontology is represented in a predicate form.

Predicate	Description	rdfs:domain	rdfs:range
dwesa:hasId	the unique identifier of the book	*dwesa:Book*	*xsd:string*
dwesa:hasTitle	the title of the book		
dwesa:hasAuthor	the author of the book		
dwesa:hasType	file type of the book (e.g. audio, text)		
dwesa:hasLocation	the file system location of the book		

Table 5.1: Gutenberg ontology

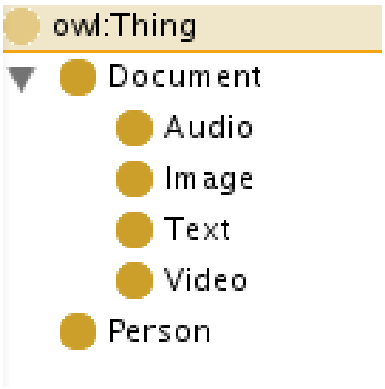


Figure 5.5: PIASK ontology

5.1.6 Support for folksonomy

Folksonomies take on the added role of providing a mechanism to facilitate the validation of the IK that is added to the knowledge platform. In the discussion on IK life-cycle (Section 2.5) the process of ensuring the relevance, accuracy, and significance of the collected knowledge was highlighted as crucial for the efficacy of the IK systems. This discussion on the use of folksonomies for IK validation is continued later in Section 8.1.2.

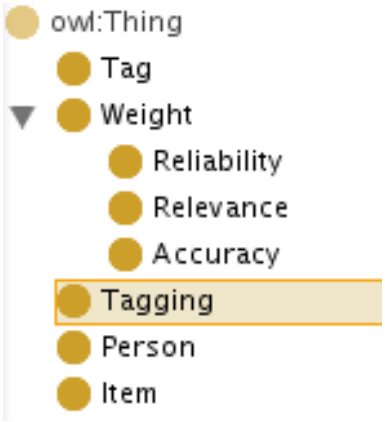


Figure 5.6: PIASK tagging and validation ontology

The tagging system is implemented through an ontology that defines two primary classes, *dwesa:Tag* class and *dwesa:Weight* (Fig. 5.6). *dwesa:Tag* represents a basic tag within an associated *dwesa:Tagging*, which defines the elements of a tagging process (e.g. tagger, time, tagged item). The *dwesa:Weight* class is sub-classed into *dwesa:Reliability*, *dwesa:Relevance* and *dwesa:Accuracy*, and it provides properties for the validation of the knowledge added into the knowledge platform by the end-users (as opposed to expert users, or system administrators).

5.2 The design specifications for the knowledge platform

The design requirements for the knowledge platform, KnowNet, derive from the deployment context as introduced in Chapter 4 and also from the objectives that have been highlighted in the discussion on the knowledge networking in rural communities. Of particular significance are the challenges that pertain to rurality, processing of IK, keeping abreast of technological developments around Web 3.0, and also ensuring ethnocentricity and context-sensitivity of the developed solution.

The discussion and specification of the platform requirements undertaken in this section allows for the initial conceptualization of the overarching PIASK architecture. The realization of the knowledge platform and the formalization of the architecture have been intertwined processes that inform each other, hence the initial discussion of the platform requirements and subsequently the concrete formalization of the architecture.

The IEEE 830 standard has been used to provide the high level template for the specification of the platform requirements [182].

5.2.1 Functional requirements and features

The goal of this research is to implement a knowledge platform for rural, marginalized communities. The primary requirements of the platform are to complement the traditional knowledge systems already present within the community, and to provide a seamless and natural user mobility between the knowledge platform and the traditional knowledge systems. The KnowNet platform has to support the interaction with IK throughout the four stages of the life-cycle of the knowledge (Section 2.5):

1. Knowledge capture - The knowledge within a community is available in different formats and at different repositories within a community. KnowNet provides the functionality for the knowledge to be recorded and made available within the platform for subsequent use.

2. Knowledge validation - Within the dynamics of processing IK, there is a need for validating the knowledge captured. This is necessitated by the inevitable inaccuracies, ambiguities, and errors that occur as a result of mass authoring of content. The process of knowledge validation is one that is undertaken within the space of the mass users of the system. Within KnowNet, the validation of knowledge is provided through a basic process of users assigning relevant weights to different aspects of the knowledge. For example, weights can be assigned for the presumed accuracy or reliability of the knowledge. These weights

are then used within the system to inform the significance of knowledge retrieval and are made available to the users to inform their knowledge use decisions.

3. Knowledge storage - The knowledge that has been captured is stored in KnowNet for subsequent processing and utilization. This meets the availability expectation from the users who need to have access to the knowledge as and when they require.

4. Knowledge dissemination - The general consensus is that knowledge is produced within a dynamic space of exchange, transfer and interaction with other knowledge sources. KnowNet implements mechanisms of interaction with the knowledge that is encapsulated in the repositories, making this knowledge accessible and so facilitating the creation of new knowledge.

The functionality required on the knowledge platform is broken down into the following specific requirements.

5.2.1.1 Heterogeneity

The implementation environment of KnowNet is characterized by variety and heterogeneity at different levels. This is by far one of the most significant constraints on the platform, that requires great levels of flexibility in KnowNet. At the very core of the problem is the reality of heterogeneous human/users expressions at the levels of personality, preferences, culture, worldviews and life philosophies (see Chapter 2). The variety that emanates from this reality should be handled in such a way that provision is made for, among other things: flexibility to modify and customize HCI components; allowing for a variety of interaction modalities (e.g. speech, text, click, gestures); ability to process and handle content in different media formats; allocation for integration of different protocols at the different layers of the application stack; and handling of interaction from different end-user devices with varied features and capabilities.

5.2.1.2 Multi-modality

Numerous advancements have been made in the field of UI design since Bolt's publication on the *put that there* multi-modal system, which discussed a combination modality of speech and gesture [70]. Research in the field of multi-modal UI interfaces addresses the specific issues and challenges associated with handling and processing multiple input commands from different sources. In this research, the requirement for multi-modality is as far as making allowance for interaction requests of different modalities. The traditional and dominant interaction on web

applications is through a mouse (i.e. clicks on buttons and hyperlinks) and through a keyboard (i.e. text commands). In KnowNet, support for various other interaction modalities is required to provide an accessible system for the different users: audio based modalities for the illiterate and the blind, and text based modalities to suit the different users' preferences.

5.2.1.3 Multimedia

The knowledge that is being processed within the platform is of multiple formats, and these represent the different aspects of the knowledge available in a community. The different media formats are also better suited for different types of knowledge, and the objective within KnowNet is to make allocation for the processing of as much community knowledge as possible. For example, the stories and the folklore in the community could be better encoded as audio content, and the cultural art could be better processed as graphics. Video would be used for traditional dances or documentation of traditional ceremonies and places of cultural significance. There is also a need to allow for transcoding between the different media formats to handle requests from different end-user devices.

5.2.1.4 Modularity

The operational environment of the knowledge platform is dynamic, with continual changes to the technology and the specific culture that is being targeted. The platform therefore needs to be modular and flexible. This equates to the ability within the platform to add, remove, and modify different components in a manner inherently supported by the architecture.

5.2.1.5 Context-sensitivity

One of the documented failures of ICT4D interventions is the lack of sensitivity to the local context. This manifests itself in interventions that do not add any immediate value to the lives of the people in the community, UI components that do not match the needs, capabilities and even less the aesthetics of the community, and solutions that have an unnecessarily steep learning curve as far as requiring the user to learn and adopt a foreign way of interfacing with the solutions implemented. The need for sensitivity to ethnographic considerations is one of the tenets of ethno-computing [3]. In KnowNet, context-sensitivity is in terms of:

1. The underlying knowledge being the local knowledge (i.e. IK) in that community.

2. The UI components that are flexible and customizable to the users preferences.

3. A variety of interaction modalities to cater for the user's preferred manner of interacting with the system.

4. Support for the different end-user devices.

5. Media transcoding depending on the capabilities of requesting devices.

6. A platform that models the social knowledge dynamics within a community (Section 4.5).

5.2.2 External interfaces

The primary users of the knowledge platform are the community members. However, the platform also needs to handle the requests for web services implemented on the platform (Section 4.6). KnowNet therefore also provides a platform upon which service applications can be implemented and deployed, exploiting the inherent features in the platform. The primary interaction with the users is through extensible access layer components that handle interaction with the end-user devices: web browsers, IM clients, telephones, and VOIP user agents.

5.2.3 Performance and attributes

The availability of benchmarks for online applications in rural marginalized communities is very limited. There are, however, industry benchmarks that can assist in the comparison of the performance metrics. In our experience the most important requirement in the rural marginalized communities is with availability of services. This is a critical factor for consideration, as it directly affects the perception that people have about technology and also influences the buy-in from the community. The expectation is for the systems to "just work" and be constantly available. Therefore the following are the performance based considerations for the implementation of the knowledge platform.

1. Robustness, resilience and recovery.

2. Correctness.

3. Maintainability.

4. Security.

5.2.4 Anthropocentric and ethnocentric design considerations

The implementation environment for KnowNet introduces factors that are specific to the community, and that need to be taken into consideration in the design of the knowledge platform. Besides the UI contextualization and IK integration into the system already discussed (Section 5.2.1.5), the platform has to make provision for the following socio-technical issues:

1. Ownership of knowledge and Intellectual Property (IP) - there are different theoretical positions with regards to IK and IP, the ownership of public domain cultural knowledge, the beneficiaries of gains made on IK, and around custodianship of the IK. It is imperative, therefore that the knowledge platform makes allocation for the implementation of different ownership structures of the underlying knowledge as enunciated in the OSCA knowledge matrix (Section 4.5).

2. Permissions and access rights - in the real world, there are specific mechanisms for controlling and determining who has access to what knowledge. These mechanisms are mostly based on the relational dynamics between people and groups. For example, there are stories that would be told within a certain family, customs and rituals that are specific to a certain clan, or craftsmanship skills that are passed within a community group. The knowledge platform, under the requirement for context-sensitivity, must allow the flexibility to establish and maintain access permissions that are based on the underlying relationship between the knowledge author and the knowledge consumer.

5.3 The PIASK architectural framework

This section presents PIASK as a re-usable architectural framework for implementation of knowledge based community oriented systems that exist in a heterogeneous environment. These environments are characterized by a multiplicity of access protocols, end-user devices, interaction modalities, usability preferences and a variety of underlying content formats. PIASK is a modular architecture that isolates and organizes the system functionality into five distinct components: presentation, interaction, access, social networking and knowledge base components. Figure 5.7 provides a high level overview of the technologies and aspects that are handled and provided for in the architecture. It presents a functional categorization of the different components in the architecture, rather than a functional stack layering. The functional dependencies in the architecture are discussed in Section 5.3.2.

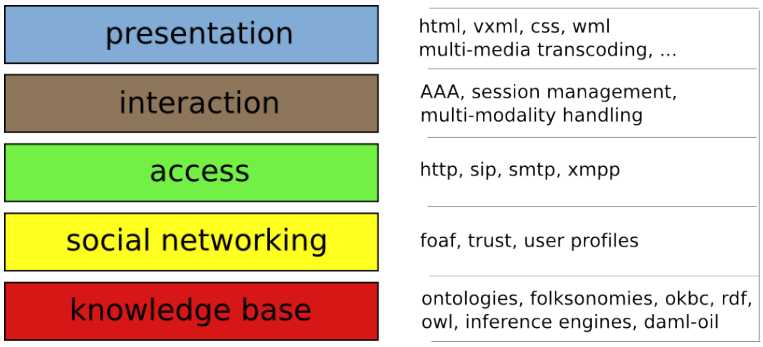


Figure 5.7: PIASK Layered architecture

The five tiers within the architecture have been derived from the system requirements specified in Section 5.2 (context-sensitivity, flexibility, multi-modality) and have been informed by the accumulated body of knowledge in software engineering and systems architecture modeling. The following are the ideas that have informed the architecture:

1. The OOD principles of behavioral completeness and modularization of software components.

2. The traditional separation of presentation, domain logic and data models as articulated through a number of architectural patterns (e.g. MVC, MVP).

3. The community oriented-ness and situated-ness of social applications (which are proliferating at an increasingly high rate)

4. Component reuse that has been prevalent in OOD and now in SOAs.

Figure 5.7 is a presentation of the architecture from a functional perspective, highlighting the components in the architecture from the point of view of the functionality that they provide. This perspective on the architecture maps to the *development view* of software architectures as one of the four views elaborated by Kruchten [183].

From the *logical view* perspective, the interaction within this architecture is primarily from the users through UI components in the *presentation* tier. The *interaction* tier provides the logic components within the architecture and therefore the primary logic handler for the user requests. The handling of user requests is facilitated by the *access* components which abstract the application layer (on the OSI model) protocols associated with the specific device. The *social networking* components facilitate the establishment of interaction parameters between the user

and the required knowledge in the *knowledge base* tier. Communication within the architecture is through message passing from components in one tier to the components in another.

The interaction specified is from the point of view of the user of the system, in terms of the services and the functions that are provided by the system. This is presented in more detail in Section 5.3.1. Further perspectives on the components interaction are discussed in Section 5.3.2.

5.3.1 PIASK functional components

The PIASK architecture abstracts the implementation for systems with the following distinct requirements: inherently knowledge based; community-oriented (or ethnocentric) as far as awareness of the social relational dynamics and interactions among the users is concerned; distributed and multi-user; operating in a varied environment with regard to transport protocols, end-user devices, and media formats; and sensitive to the context of deployment. Each tier plays a distinct functional role towards addressing these overall system requirements.

5.3.1.1 Presentation

The presentation layer handles the formatting of content for the specific devices that are requesting the information, taking into consideration the different device capabilities. For example, a request from VOIP phone will have the information transcoded into an audio stream that is sent to the phone, while a request from a Personal Digital Assistant (PDA) will have the information in a mark-up language specific for the PDA (e.g. HTML). This layer also handles the reformatting of content based on other usage scenario factors, which are provided by the interaction layer. For example, depending on the network usage patterns (e.g. network congestion), an audio clip, a graphic or textual description can be sent as alternative content for a response in order to achieve effective bandwidth use. This layer handles the provision of some of the following standards and technologies: voiceXML, Hyper Text Mark-up Language (HTML), eXtensible HTML (XHTML), and Cascade Style Sheets (CSS).

5.3.1.2 Interaction

The interaction layer provides the primary implementation logic within the architecture and facilitates the interaction between the user devices and the underlying knowledge base. This interaction is influenced by the features and capabilities of the requesting devices and by the different usage scenarios. The different application use-case scenarios would therefore be implemented

within the interaction layer components. This layer also handles the different modes of interaction and the translation of different types of requests from the users (e.g. textual requests, audio requests, voice commands, and Dual Tone Multi Frequency (DTMF) input).

5.3.1.3 Access

Access to content is generally limited by the lack of devices that handle the provided communication protocols. This layer handles requests for information from multiple devices. The access layer provides an extensible listener interface that handles requests based on different application level protocols. Hyper Text Type Protocol (HTTP) is used extensively on the Internet and is an example of the protocols that are handled by the access layer. This layer also provides the access interface to the end user devices (i.e. through interaction channels), for requests that are initiated from the system.

5.3.1.4 Social networking

In view of the fact that knowledge creation occurs within a social context, this layer models the social behaviors that are prevalent in the everyday interactions among the users of the system. The social networking layer also handles the trust related considerations, which are articulated within the relationships between a community of users. This layer is implemented to integrate different standards and protocols that are already established for social networking (e.g. the Friend Of A Friend (FOAF) standard). It situates the developed systems within the social dynamics of the users, and therefore provides the community orientation of the system.

5.3.1.5 Knowledge base

The knowledge base layer hosts the knowledge that is accessible to users. The basis of this layer is knowledge encapsulated in ontologies (or other knowledge bases) that have been locally populated and are locally maintained (Section 5.1). The population of these ontologies is made possible via tools that are implemented to capture and codify the local knowledge. The knowledge base layer provides interoperability between different knowledge repositories through the Open Knowledge Base Connectivity (OKBC) interface. This also allows for the integration of externally maintained knowledge repositories, taking into consideration the heterogeneity of KR formalisms used (e.g. RDF, DAML + OIL, OWL).

5.3.2 Component interaction

The primary component level interaction within the architecture is depicted in Figure 5.8 and shows the functional dependencies between the components. In this and subsequent diagrams, the letters: P, I, A, S and K, represent the five layers of the architecture: presentation, interaction, access, social networking and knowledge base respectively. The circles highlight the outward dependencies and the rectangles show the inward dependencies between the components. As an example, in Figure 5.8 the social networking layer functionally depends on the knowledge base layer which stores the user profiles in a FOAF based ontology.

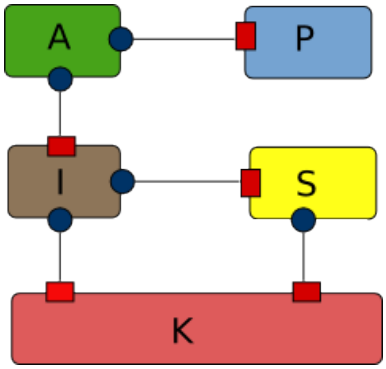


Figure 5.8: PIASK system level interactions

As illustrated in this figure (5.8):

1. The access level components primarily handle the interaction with the end-user devices. In handling this interaction, the access components depend on the presentation tier for the rendering of the content specifically for the requesting device. There is therefore the outward functional dependency from the access tier to the presentation tier.

2. The access tier also depends on the interaction components for the logical processing or requests.

3. All the domain logic is encapsulated within the interaction tier, which makes the interaction components the most tightly coupled tier in the architecture. In handling user requests, the interaction components primarily interact with the knowledge base and social networking components.

4. The social networking tier is also functionally dependent on the knowledge encapsulated in the knowledge base tier.

The interaction between the system components gives support to the following useful dependencies which together provide a more complete representation of the architecture:

1. Object Oriented Architecture aspects - This is due to the behavioral completeness of the components which encapsulate the operations and the associated data on which they operate. The components are autonomous black boxes as far as the interaction with other system components are concerned, thus ensuring the modular architecture that supports reuse, and for replacing system components in a flexible, extensible manner (Fig. 5.9).

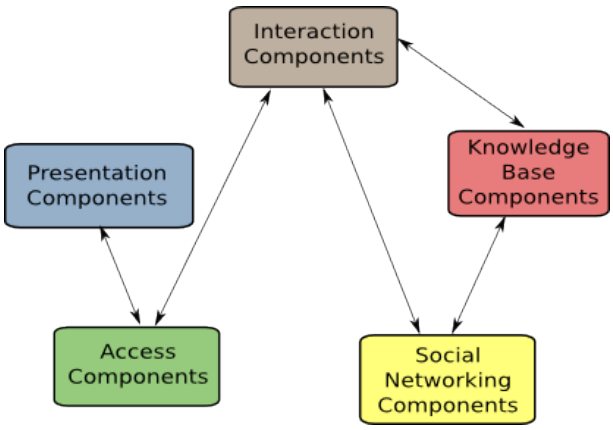


Figure 5.9: PIASK Object Oriented Architecture perspective

2. Service Oriented Architecture aspects (Fig. 5.10) - The different components within a system provide the functionality that handles a specific business process. This functionality is available as independent services to the requesting service consumers. The interaction between the different service components is achieved through message passing based on the specified communication protocol.

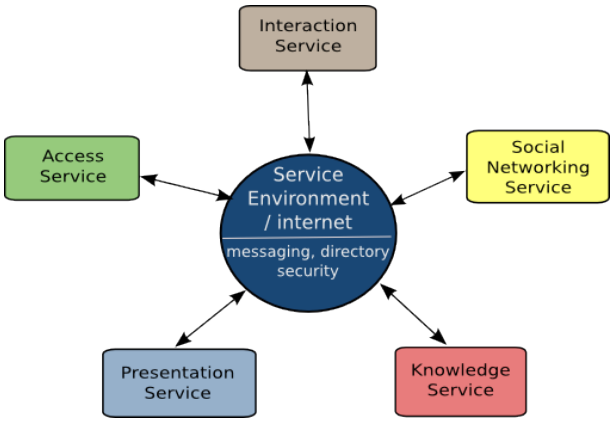


Figure 5.10: PIASK Service Oriented Architecture perspective

3. Layered Architecture Aspects - Figure 5.11 is a Dependency Structure Matrix (DSM) of the PIASK architecture based on the component interaction discussed in Section 5.8. From the initial DSM, the architecture does not show any immediate structure with regards to the interaction (i.e. functional dependency) between the components. However undertaking a partitioning of the DSM and rearranging the tiers reveals a more definite pattern of a layered architecture, as discussed by Sangal *et al.* [184] (Fig. 5.12). The PIASK architecture is not a strictly layered architecture (i.e. represented by a single line of dependency below the diagonal line) but the lower triangular pattern in the DSM indicates the layering within the architecture, where the top layers are dependent on the lower layers (and not the other way around). This emergent layering in the architecture reveals the logical dependencies in the architecture associated with handling user requests. The partitioned DSM (Fig. 5.12) further formalizes the PIASK architecture from the point of view of specifying the logical dependencies that are allowed and disallowed in the architecture.

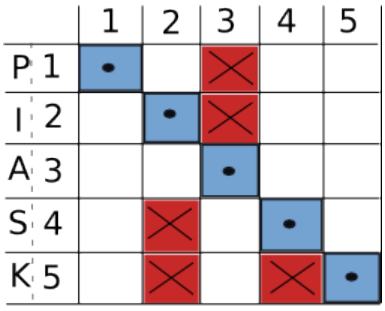


Figure 5.11: PIASK initial DSM

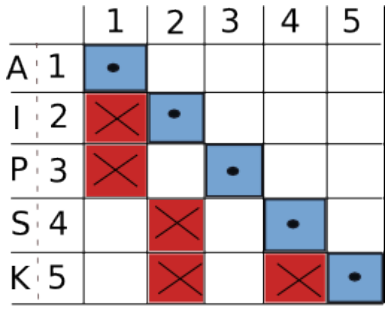


Figure 5.12: PIASK partitioned DSM

5.4 PIASK and other architectures

Architectural patterns[2] are developed for structuring system components for different domains. We have developed PIASK as an architectural pattern for distributed knowledge-based systems that exist in a heterogeneous environment and that are intrinsically context-sensitive. This section introduces other architectural patterns, and then juxtaposes them with the PIASK architecture to highlight the key differences and similarities. This highlights the areas to which the PIASK architecture makes a contribution, within the domain of software architectural and design patterns.

5.4.1 The MVC and associated architectures

The Model-View-Controller (MVC) pattern was developed in the 1970s and early 1980s by Trygve Reenskaug while visiting Xerox Parc. It was developed initially for applications running on single graphical workstations, but was later adapted for distributed applications [185]. The key aim of the MVC pattern was to ”bridge the gap that exists between the user’s mental model and the digital model that exists in the computer” [185]. This objective highlights the key feature of the MVC architecture, which is the separation of the presentation elements from the domain objects. There have been a number of derivatives of the MVC pattern, which include the Model-View-Presenter (MVP) pattern, and the similarly structured Presentation-Abstraction-Control (PAC) pattern [186, 187]. The separation of the presentation components from the domain logic and data models allows for different and multiple presentation elements to be linked to the independent underlying data and logic objects. The presentation elements in the architecture are encapsulated in the *view*, the interaction and the control logic is handled by the *controller*, and the underlying data is in the *model* component of the architecture.

[2]In this thesis, the terms *architectural pattern* and *architectural framework* are used interchangeably

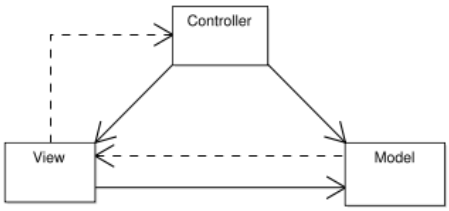


Figure 5.13: MVC architecture [79]

The MVC is a triangular architecture where the *view* elements communicate with the *controller*, and *controller* to the *model* components which then directly communicates with the *view* element (Fig. 5.13). The user interaction is with the *view* components, which trigger an event which is processed by an event handler in the *controller*, the *controller* forwards the corresponding changes to the model components, which then provide the information which the *view* uses to update the UI elements.

5.4.1.1 The three-tier architecture

The three-tier architecture is a derivative of a client-server architecture that emerged in the 1990s, within the era of distributed systems [79]. Conceptually the three-tier architecture is similar to the MVC in that distinct lines of separation are drawn between the presentation tier, the logic tier and the data tier (Fig. 5.14). The key difference is in the manner in which processing is handled in the architecture. Unlike the MVC, processing within the three-tier architecture is layered and so communication is from the presentation layer, through the logic layer and then the data layer. The reverse process occurs through the logic tier as well, without directly passing to the presentation (as in MVC where communication goes directly from the model to the view).

5.4.1.2 The naked objects architecture

The naked objects architecture is derived from the OOP principles of domain objects encapsulating all the business logic, and presentation elements being the direct representation of the domain objects. The separation of systems components within the naked objects architecture is informed by the MVC pattern. The key point of difference between the naked objects architecture and the MVC is that the View and the Controller roles are completely generic to allow for the automatic rendering of the presentation elements based on the underlying domain objects [188].

The users in this architecture directly interact with and view the underlying domain objects, hence the name “naked objects”. The interaction on the system based on the naked objects pattern is through invoking behaviours that have been defined in the respective domain objects.

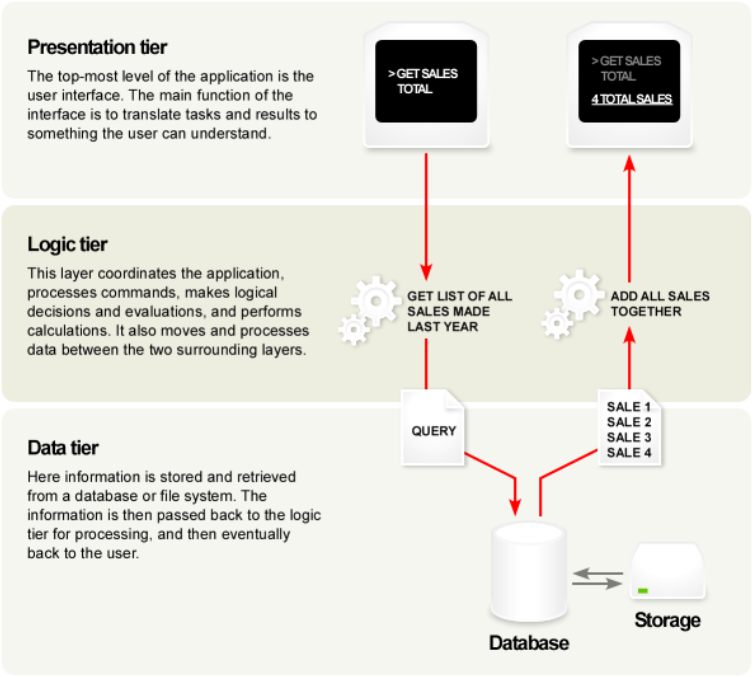


Figure 5.14: 3-tier architecture [79]

This architecture therefore encourages the development of domain objects that are behaviorally complete.

5.4.2 Pipes and filters architecture

"The pipes and filters pattern provides a structure for systems that process a stream of data" [189]. Within this architecture there are two main types of components: the filters and the pipes. Processing goes through sequential steps of transformations that are undertaken by the filters on the data that is communicated through the pipes. Each filter therefore defines an input interface and an output interface through which data is communicated. There are specialization of this pattern which place constraints on different aspects [190]: *pipelines* puts a constraint of handling execution in a linear order, *typed pipes* allows for communication of well defined data types on the pipes, and *bounded pipes* specifies the amount of data that is allowed on the pipes.

5.4.3 Data abstraction and object oriented architecture

In this architecture, the main components are the objects which encapsulate the data and the native operations on the data. Object oriented architectures allow for a decomposition of a problem

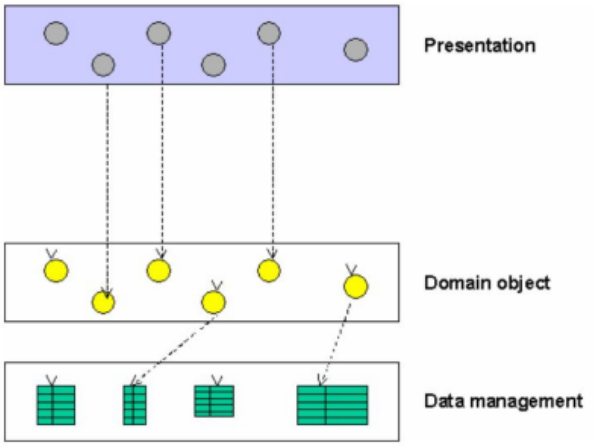


Figure 5.15: Naked Objects [188]

into modular, reusable objects. This architecture has proliferated extensively within the software engineering domain and in programming languages. One of the key concepts in this architecture is data abstraction, which allows for an object to be manipulated only through the well defined external methods [191]. Processing within this architecture is undertaken by objects invoking functions and procedures defined in other objects, and as such the objects have a reference to the other objects on which they are functionally dependent.

5.4.4 Event based, implicit invocation architecture

In this architecture, the flow of execution is directed by events. The components (event-consumers) register an interest in an event and associate a procedural handler to execute when the event occurs. The events in this architecture can be messages or conditions which are generated by the event-producers. This architecture has a strong support for reuse as the components can be replaced without affecting the connectors between them [190].

5.4.5 Repositories or blackboard architecture

In this architecture, there are two types of components: the blackboard, which is the central data structure; and the independent knowledge source components that operate on the data [190]. This architecture allows for handling of complex problems in which the constraints are not clearly defined. Processing within this architecture involves the blackboard being updated by the knowledge source components until a solutions is reached. The update to the blackboard is undertaken when the state of the blackboard matches a defined constraint in one of the components [79].

5.4.6 The PIASK perspective

The understanding in the development of architectural patterns has been informed by the observations that were made as far back as 1960s when OOP paradigm was coming to the fore. Within the OOP paradigm, the separation between data and software procedures was abandoned in favour of objects, which are behaviourally complete software entities that represent the mental-models of the associated real-world entities in terms of their properties and behaviours. The PIASK pattern is built on the OOP principles of behavioural completeness to make a distinction between the different functional components in the system. At the same time, PIASK adopts a user centric approach (that is observed in the MVC architecture and the other architectures that separate the UI presentation components from the logic and data model components) in which the interaction with the user is facilitated by the presentation components. In PIASK, however, a deeper level of interaction with the user has been implemented, beyond the UI components (through the presentation layer) to include an integration of the relational, social dynamics associated with the usage of the underlying knowledge (through the social networking layer). As seen in the MVC architecture, PIASK maintains the structure of separating the data components from business logic components, through the knowledge base and the interaction components respectively. PIASK introduces another tier which handles the interaction within a context of end-user device multiplicity and heterogeneity: the access layers allows for handling of requests in a manner that removes the application level (on the OSI model) protocol logic from the business logic, and allows greater flexibility to abstract the interaction with the end user devices.

The PIASK architecture is designed specifically for knowledge based applications, and therefore the knowledge base components provide the basis on which computation is undertaken (i.e. the basis is the processing of the underlying knowledge). While these components are central in the architecture, they do not play the same role as the central *blackboard* in the blackboard architecture. Also, the order of processing within PIASK is more determined in that the functional dependencies between the components are restricted in the architecture (as illustrated in the DSM in Figure 5.12). Computation within the architecture is primarily through method invocation between the components.

PIASK encompasses a Component-Based Development perspective, in which the different layers of the architecture are populated by software components [192, 193]. The architecture does not make a specification with regards to the nature of the components at the different levels, which allows for the realization of the architecture through service oriented components, object oriented components, or agent based components (as illustrated in Chapter 6).

The closest similarity is observed between PIASK and the MVC architecture (and its deriva-

tives) in their user centric design and their definition of functional dependencies between components. PIASK, however, introduces a separation of functional components at social networking and access levels, to provide a more user centric (integrating the social dynamics) and flexible (abstracting the device interaction logic) architecture, respectively. The nature of the components within the architecture has been influenced by the OOP paradigm in which the components are behaviourally complete and the interaction between the components is well defined, ordered and functionally deterministic (unlike the blackboard or event-driven architectures).

5.5 Conclusion

PIASK is an architectural framework for knowledge based applications that are ethnocentric and with flexibility towards context-sensitive adaptation. The five basic component tiers that are defined in the architecture are presentation, interaction, access, social networking and knowledge base. These components are behaviourally complete, and functionally loosely coupled for the implementation of systems that are flexible, modular and scalable. The overall architecture has been informed by the experiences within OOD and other software architecture with regards to the separation of user interface components, business logic, and data models. The unique features of the PIASK architecture are the explicit abstraction of the device specific access functionality into the access layer, and the encapsulation of the context social dynamics through the social networking layer. PIASK provides a high level architectural template that can be realized in numerous ways. A specific realization of the PIASK architecture, for the Dwesa community, is described in the next chapter.

Chapter 6

A Realization of PIASK

> "In theory there is no difference between theory and practice, but in practice there is" (Murphy's Technology Law)

The PIASK architecture described in Section 5.3 has been utilized as a pattern in the implementation of a knowledge platform for marginalized rural communities. The architecture provides the high-level structuring and organization of the components in the platform, according to the functional/logical tier on which they reside. The derived and developed knowledge platform, called KnowNet, is a practical specialization and a proof-of-concept example of this architecture. It is also a validation of the adequacy of the architecture to provide a template for systems in similar domains.

6.1 A MAS knowledge platform

KnowNet is implemented as a Multi-Agent System (MAS). The agent architecture fits in with the PIASK architecture to provide decoupled, modular units (i.e. agents) of computation that exist within a community. The mapping of the agents onto the PIASK tiers is on the basis of the functionality they provide, and not through predefined associations between the agents. Agents appear suitable for the attainment of the design requirements of the knowledge platform (Section 5.2): robustness, flexibility and resilience provided by the ability to add, replace, or terminate agents in a manner that is natural to the agent community, and also the ability to migrate agents from one host to another to respond to the different environmental scenarios. The agent architecture also provides a scalable platform in which more agents can be added to the community to handle increased system loads, thus ensuring greater levels of availability of

the system services. Redundancy can also be easily built into the agent systems with the goal of ensuring even greater levels of system performance. The motivations for the utilization of a MAS to realize the knowledge platform are retrospectively further discussed in Chapter 7.

Amidst a plethora of agent systems, the Java Agent Development Environment (JADE) agent platform has been selected for the implementation of KnowNet. JADE appears adequate as the implementation platform due to the ease of use, availability and application in various similar domains. The agents that have been developed and their various logical interactions are shown in Figure 6.1, with a colour-coded mapping of the corresponding PIASK architectural layer.

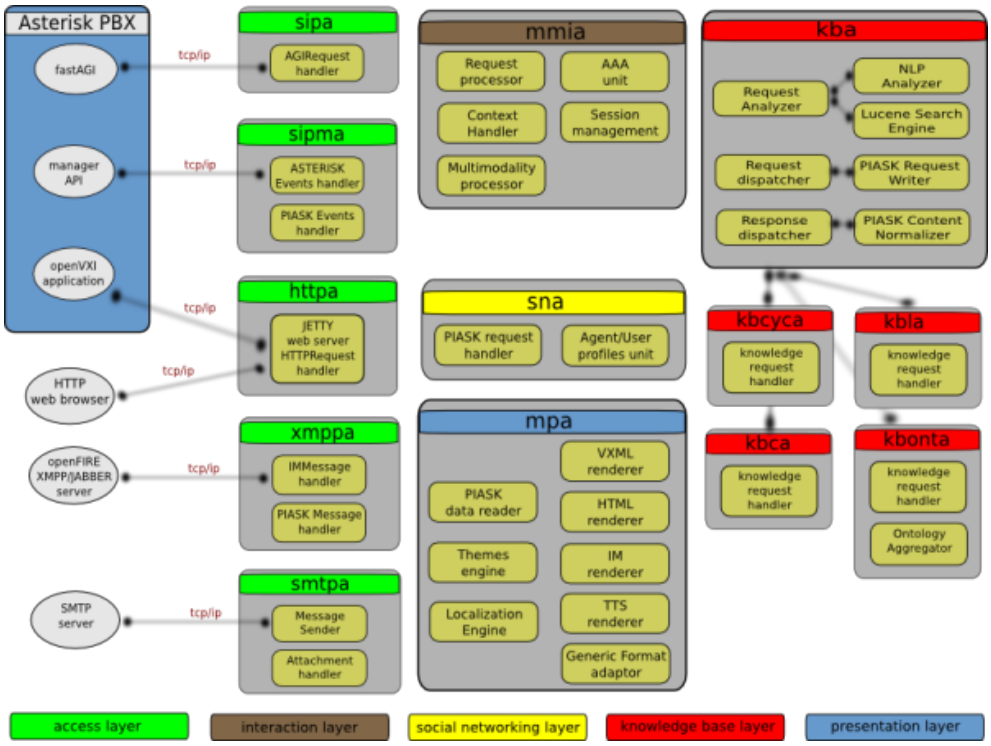


Figure 6.1: KnowNet community of agents

The agents with KnowNet are autonomous and provide specific services to the community that they are part of. Within the JADE MAS platform, agents exist within containers, which can be located on different hosts [166]. To be part of a community, agents have to exist in containers that join a single Main Container to form a distributed system that appears as a single platform. The Main Container is different from other containers in that it holds two platform specific agents: the Agent Management System (AMS) and the Directory Facilitator (DF). Once the agents have joined a container, they advertise their behaviours to the DF, which provides directory services for the community. These behaviours are then available to be queried and

requested by the other agents on the platform. Agent behaviours represent services that are provided by an agent. The agents within a platform communicate by exchanging messages that adhere to a specific ontology or a defined platform *lingua franca*. The communication protocol within KnowNet is defined in Section 6.7.

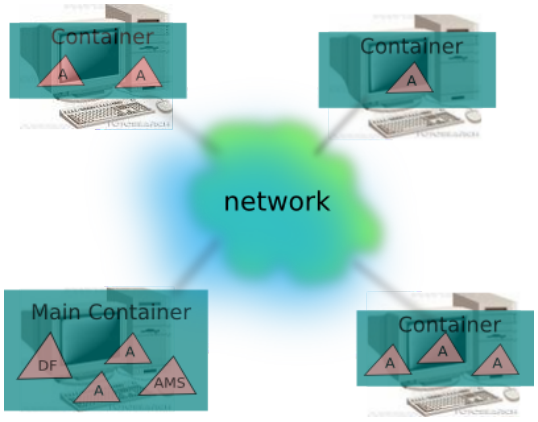


Figure 6.2: KnowNet MAS platform

The KnowNet MAS is implemented as a single agent platform with containers distributable across different hosts (Fig. 6.2). The agents can then be mobilized and positioned to factor in the environmental scenarios. For example, an agent that provides access to knowledge bases located on the Internet could be positioned in a container on a gateway machine or a host with Internet access. Agents can be also moved around for load-balancing purposes on the available hosts. Figure 6.3 is a view of the agent platform through a Remote Management Agent (RMA) showing the Main Container on one host (*thepentagon.ict.ru.ac.za*) which contains most of the agents (including the AMS and the DF), and a container located on another host (*thebox.ict.ru.ac.za*) which contains the Multi-Modal Interaction Agent (MMIA).

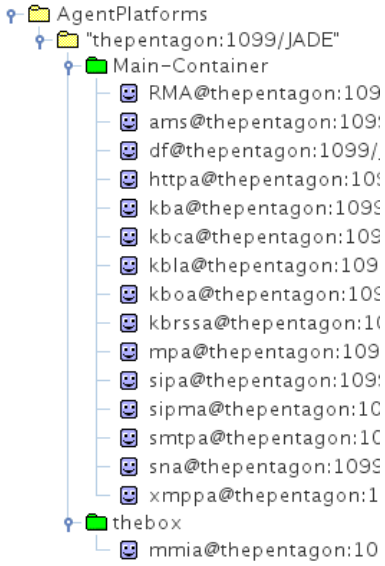


Figure 6.3: Multi-host KnowNet container

The following sections discuss the agents that have been implemented at each of the five tiers of the PIASK architecture.

6.2 Presentation agents

The presentation agents handle the rendering of the content received from the knowledge base layer agents, and the content generated from within the KnowNet platform, for the different devices requesting the information. These agents primarily handle the heterogeneity of the content that is communicated and the diversity of end user preferences. The role of the presentation agents is therefore the rendering of content at two levels: the device level and the user level. At the device level, the content is formatted according to the requesting device capabilities (e.g. mark-up language, screen-sizes) and at the user level, the content is rendered according to the user preferences (e.g. UI language, aesthetics, layout, theme). The primary agent that handles this behaviour is the Media Presentation Agent (MPA).

6.2.1 MPA

The MPA agent[1](Fig. 6.4) implements content renderers in order the achieve the transcoding of data between different media formats and mark-up languages. The MPA agent also handles interface localization in terms of allowing for customization of interface themes, and the translation of platform specific interface strings into different languages.

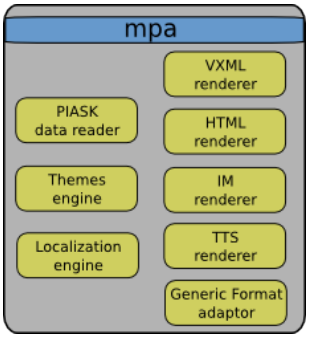


Figure 6.4: MPA agent

The *mpa*[2] receives the content formatted in either plain XML format or the internal platform format (Section 6.7). This information is read by the *PIASK data reader* and forwarded to the relevant renderer. For example, when an unauthenticated request is handled on KnowNet, the interaction agent returns content as shown in Figure 6.5. The *mpa* then transcodes this content into a corresponding HTML content (Fig. 6.6) or VXML content (Fig. 6.7), using the HTML renderer or the VXML renderer respectively. Allocation is also made for new content formats which are defined and integrated into the platform through the *Generic Format adaptor*.

```
<piask:div>[piask message]
     You have not been authenticated
<piask:divoff>
<piask:div>[Login]
     <piask:form>[login.piask]
       <piask:text>[username]<piask:br>
       <piask:password>[password]<piask:br>
     <piask:formoff>
<piask:divoff>
```

Figure 6.5: Content in platform data format

[1]Note that even though the last "A" in the acronym MPA stands for *Agent*, we still explicitly use the word *agent* after the acronym because MPA also becomes the nickname (within the implementation MAS environment) of the particular agent.

[2]The italicized, small letter *mpa* is a JADE specific nickname for the MPA agent. The two forms (i.e. the italicized small letters and the capital letter acronyms) are used interchangeably to refer to the different agents.

```
<html><head><style type="text/css">...</style></head>
<body>
<div id="results">
  <div id="property">
    <div id="piask message">
      <div id="propTitle">piask message</div>
      <div id="propVal"> You have not been authenticated </div>
    </div>
  </div>
  <div id="property">
 <div id="Login">
   <div id="propTitle">Login</div>
   <div id="propVal">
   <form method="GET"
         action="http://thepentagon.ict.ru.ac.za:8080/login.piask">
      username <input type="text" name="username"/><br/>
      password <input type="password" name="password"/><br/>
      <input type="submit" value="Submit to piask"><br/>
   </form>
   </div>
   </div>
   </div>
</div>
</body>
</html>
```

Figure 6.6: Content in HTML format

```
<?xml version="1.0"?>
<vxml version = "2.0" xmlns="http://www.w3.org/2001/vxml">
<link dtmf="0" next="#exit"/>
<form>
   <block>piask message :  You have not been authenticated </block>
   <field name="username" type="number">
       <prompt> Please enter username, followed by hash </prompt>
       <filled>
         <prompt> You entered <value expr="username" /> </prompt>
       </filled>
    </field>
    <field name="password" type="number">
       <prompt> Please enter password, followed by hash </prompt>
       <filled>
         <prompt> You entered <value expr="password" /> </prompt>
       </filled>
    </field>
    <block>
       <prompt>Processing information </prompt>
       <submit next="http://thepentagon.ict.ru.ac.za:8080/login.piask"
               method="get"
               namelist="username password "/>
    </block>
</form>
<form id="exit">
    <block>Goodbye!<disconnect/></block>
</form>
</vxml>
```

Figure 6.7: Content in VXML format

The *mpa* agent is typically at the end of the request handling cycle, in which knowledge has already been queried from the knowledge base layer agents, and is about to be dispatched back to the requesting client devices by the access agents. Figure 6.8 shows one such interaction, which occurs as follows:

1. The *access agent (httpa)* receives a request for information - a HTTP request from a web browser in this case.
2. The *access agent* forwards the request to the *interaction agent (mmia*).
3. After validating the request, the *interaction agent* forwards the request to the *knowledge base agent (kba).*
4. The *knowledge base layer agent* analyzes the request and forwards it to the relevant *knowledge agent (kbca).*
5. The response from the *knowledge agents* is sent back to the *interaction agent.*
6. The *Interaction agent* appends the necessary interaction components (e.g. menus, updates, tags) and sends the content back to the *access agent.*
7. The *access agent* sends the content to the *presentation agent (mpa)* for formatting, based on the users' profile.
8. The *presentation agent (mpa)* formats the content and returns it to the *access agent* and ultimately to the requesting device.

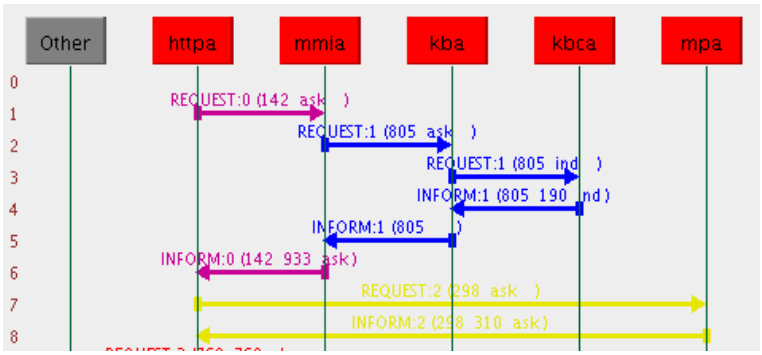


Figure 6.8: MPA agent interactions

The *mpa* agent's cycle through the presentation behaviour, which it advertises with the DF-Service, involves receiving the requests from agents and responding with the formatted data.

The *mpa* agent is therefore functionally independent from any other agents within the community other than the DF agent. This is in contrast to, say, the access agents which have to be aware of the interaction and presentation agents on the platform.

6.2.1.1 Themes engine

The themes engine is utilized specifically for web browser based requests. The engine utilizes Cascade Style Sheets (CSS) as the theme definition language for the web pages that are rendered by KnowNet. The different styles are read from the platform's theme directory and loaded in-memory into a *HashMap*. The requests that are handled for rendering by the MPA agent encapsulate the preferred themes as specified in the user's preferences.

Applying a CSS style to a page is achieved by specifying the style parameters for the different objects within the DOM structure of a page. The fine-grained control over the styling of different aspects of pages is achieved through a highly granular div tag (*i.e.* <*div*>) structure that is associated with each HTML file as formatted by the knowledge base layer agents (Section 6.6) (Fig. 6.9).

6.2.1.2 Localization engine

The localization engine is specifically responsible for the translation of UI content into different languages. The engine utilizes the standard GNU Gettext Portable Object *(.po)* files which are loaded in memory (into a *HashMap*) when the MPA agent is initialized. The in-memory storage of the translations allows for faster processing of requests, by eliminating the file read and string comparison operations on the *.po* file. The utilization of the *.po* files also provides the added benefit of being easily editable by humans, allowing for easy localization of the platform.

On initialization of the platform, the MPA agent queries the *languages* directory and loads all the available *.po* files. The associated languages are then made available to the users to specify as their preferred interaction language in their user profiles.

The linguistic localization is the last process that occurs in the MPA agent request handling cycle. The translatable strings are specified using a platform specific tag, <*piask:string*>*[string]* (see Section 6.7 for a discussion on the platform communication protocol and the message tags). The localization engine falls back to the default language strings for cases where no matching translation exists.

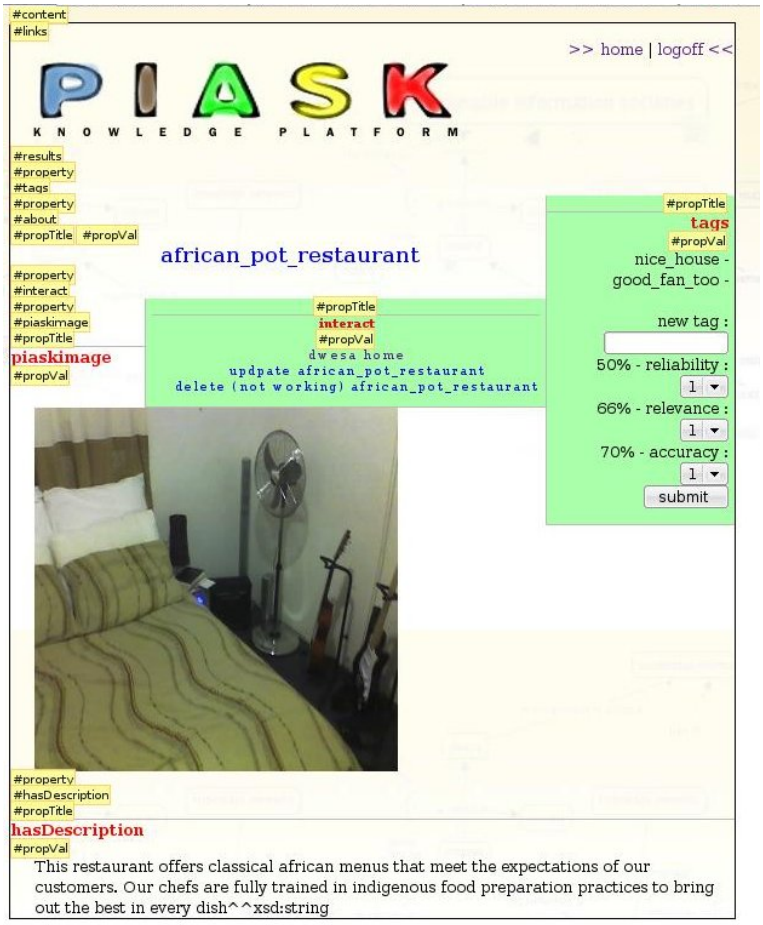


Figure 6.9: Standard page div tag structure

6.3 Interaction agents

The interaction agents play a central role in the implementation of application service logic within the knowledge platform. They provide the logical interface between the platform users and the underlying knowledge, and between the different access layer agents and the knowledge base layer agents. The Multi-Modal Interaction Agent (MMIA) has been implemented as the only interaction agent in KnowNet to handle the provision of the layer specific functionality.

6.3.1 MMIA

The MMIA agent primarily handles user requests that it receives from the access layer agents, and also internally triggered system requests. Two behaviours have been implemented in the MMIA agent and both of them are triggered by the reception of a properly signed message.

One behaviour polls the message queue for incoming requests, and the other polls the queue for incoming responses. A typical interaction with the MMIA agent is shown in Figure 6.8, wherein it provides request handling on behalf of access layer agents and forwards the request to the relevant agents for further processing. Other interaction scenarios involving the MMIA agent are discussed in subsequent sections on different agents. The modules within the MMIA agent (Fig. 6.10) collaborate to provide an end-to-end logical handling of the user requests.

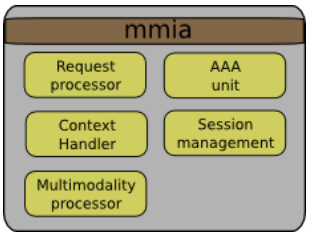


Figure 6.10: MMIA agent

6.3.1.1 Request handler

The request handler module is the first to receive requests from the access agents. This module first unpacks the message payload to strip off internal inter-agent platform messages (further discussed in Section 6.7). These messages encapsulate system information such as User Session IDs, request context, and variables required for the processing of the request. The request handler then engages the rest of the agent modules in resolving the request. The next module in a typical request scenario is the Authentication, Authorization and Accounting (AAA) module.

6.3.1.2 AAA module

The AAA module is primarily responsible for the tasks of: *authenticating* the users on the platform, by interfacing with the Social Networking Agent (Section 6.5) which is aware of the different platform users' profiles and authentication credentials; determining and enforcing *authorization* for various requests on the underlying knowledge in the system; and finally for maintaining basic usage logs and *accounting* for the requests and the processes that KnowNet handles.

A key feature that has been implemented is in the manner in which authorization is handled within the platform. As established in the requirements specification for the platform (Section 5.2), the environmental context necessitates implementation of flexible and granular access permissions on the underlying IK. KnowNet therefore provides an authorization mechanism based on a more detailed and granular relational association between the author (i.e. owner) of the

knowledge and the consumer (i.e. requestor) of the knowledge. A first step in implementing this feature was to define various other relevant relationship types within the users ontology, which are all derived (*rdf:subClassOf*) from the knows (*dwesa:knows* which is equivalent to *foaf:knows*) relationship, and which relate one person to another (*rdfs:domain* and *rdfs:range* is *dwesa:Person*) (Tab. 6.1).

Relationship	Definition
dwesa:isFamily	associates two persons who are in the same family
dwesa:isExtendedFamily	associated two persons who are in each other's extended family
dwesa:isClan	associates two persons who in same clan
dwesa:isFriend	associates two persons who are friends

Table 6.1: KnowNet relationship types

Having established the underlying ontology for specifying the different relationship types, each content object within the platform was associated with a permission mapping based on the relationships that the owner has defined within their profile. Each content object on KnowNet is defined within the dwesa ontology as a sub-class (*rdf:subClassOf*) of the *dwesa:Item* class. The *dwesa:hasPermission* property has the *dwesa:Item* as its domain and *dwesa:Permission* as its range (Fig. 5.5).

The resolution of access permissions for every user request involves determining the relationship status between the content author and the requesting consumer, checking the permissions as specified in each content item's permission property, and enforcing the authorization control through either forwarding the request to the relevant knowledge base layer agents or responding to the access agents with an appropriate *access denied* message.

6.3.1.3 Session management

The MMIA agent is also responsible for managing the user sessions on the platform. This is maintained through an in-memory list of the current sessions. For every session on the platform, the MMIA agent queries the Social Networking Agents (SNA) to determine the relevant users details, and stores this in the session list. The details that are queried from the SNA agent include, the IM address of the user and the online status, the telephone extension of the user, and their email address. This information is utilized by the MMIA agent to determine the best channel of interaction with the users. For example, if a user is logged onto KnowNet through a VOIP agent,

and requests a download of a certain non-audio file, the MMIA agent will determine that the best channel to send the file through is via an email.

Other example of where these details come handy is a scenario where a user is browsing the content available on the platform, and from the content page they access the profile page of the author of the content, from whose page they can follow a link that establishes a direct call with the author - this is the kind of value-added functionality that would improve an eCommerce service user experience.

6.3.1.4 Context handler

As mentioned already, the MMIA agent is the primary agent that implements procedural logic on KnowNet. It does this by defining various contexts of operations to which each request is forwarded. The following table (Tab. 6.2) gives a brief overview of the different contexts that have been defined.

MMIA agent context	Description
login.piask	this is the context for handling login requests
logoff.piask	this handles the request for logging off
know.piask	handles all the requests for knowledge on the KnowNet platform
update.piask	handles the updating of knowledge on KnowNet and adding new knowledge
do.piask	handles requests on KnowNet to perform platform related operations - send an email, call an extension, notify a user of a message, etc.
upload.piask	handles uploading of content onto KnowNet
register.piask	provides the logic for new user registrations
delete.piask	handles delete content requests on KnowNet

Table 6.2: MMIA agent contexts

While these contexts are not an exhaustive specification of all the functionality implementable in KnowNet, they form the core of the procedural logic for operations on the underlying knowledge base repositories. The rest of the MMIA agent contexts are detailed in Appendix C.

6.3.1.5 Multi-modality

The MMIA agent also handles and provides multi-modality interaction on the platform. Multi-modality in this case is used to refer to the handling of interaction with the user through multiple channels that the platform associates with the user. These different channels are directly handled and provided by the corresponding access layer agents (e.g. the HTTPA agent provides the channel to handle textual interaction from a HTML web browser, and also to handle audio interaction with the VXML browser through a VOIP PBX implemented using Asterisk [194]). The majority of the interaction undertaken on KnowNet is uni-modal and through a single channel. The basic multi-modality mechanism implemented on the platform can be described as temporally closely coupled, synchronous, and based on the early fusion technique:

1. Temporal coupling - the handling of user interaction from alternative channels is initiated within a temporal proximity of the user's previous events on the platform.

2. Synchronous - the processing of input from different channels is coordinated based on the trigger from the primary and the initial user channel on the platform. The secondary interaction channel is therefore synchronized to the events in the primary channel.

3. Early fusion - in early fusion techniques, the processing of one interaction mode triggers the handling of another mode. For example, within the platform, the event handling of a DTMF input could be activated as a result of a user request via the web browser channel.

The following scenarios highlight the different aspects of multi-modal interaction on the platform:

1. A user accessing the knowledge via a web-browser can change the interaction channel to the VOIP device, and get the content rendered as audio through that channel.

2. The authentication of a user of the platform can involve requesting the user to enter their username through the web interface, and to key in their password through the DTMF channel.

3. Output from the platform can be routed through any of the relevant channels that are associated with the user.

Multi-modality implementation The level of multi-modality that is implemented is based on the association of the online users, with the different interaction channels that are implemented in the access layer agents. On authenticating the user on KnowNet, the MMIA agent queries the relevant agents to determine the rest of the channels that are available for interaction with the user. For example, if a user accesses the platform through the web interface, the following available channels resolution process ensues:

1. The MMIA agent creates a new online session object for the user (Fig. 6.11).
2. MMIA agent queries the SNA agent to determine profile details of the user (e.g. telephone number, IM address, VOIP extension, and email address).
3. MMIA agent queries the relevant, presence-aware agents to determine the availability status of the user. For example, the XMPPA agent is queried to determine if the user is available through their IM chat client.
4. The MMIA agent stores the information about the available interaction channels for the specific user, and associates that with the user session object (Fig. 6.11).

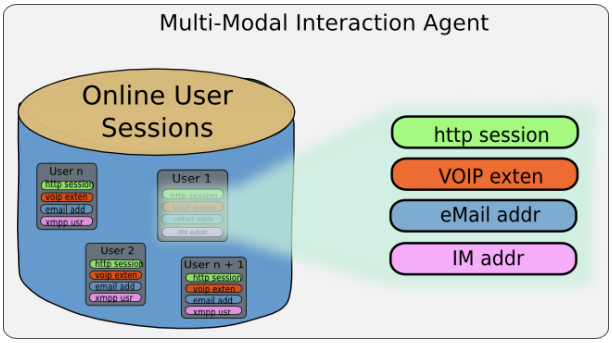


Figure 6.11: MMIA agent current session channels

Once the MMIA agent has aggregated all the channels that are available for interaction with the user, it then handles the different conditions to determine the appropriate interaction modality with the user. These conditions include: user initiated events, where users can explicitly request for the interaction to be routed to an alternative channel; interaction specific conditions - for example a user who is interacting with the platform through the VOIP interface, and requesting a download of a file that cannot be transcoded into audio (e.g. an image file - currently at least), would result in the file being routed through as an email attachment for the user, or a file transfer

through the IM channel; and environment specific conditions, where the platform determines the best interaction channels as a result of platform specific factors (e.g. load balancing, avoiding congestion on certain channels, and overall performance optimization).

A multi-modal interaction Figure 6.12 is a depiction of the message passing between the agents in handling a multi-modal interaction with the user.

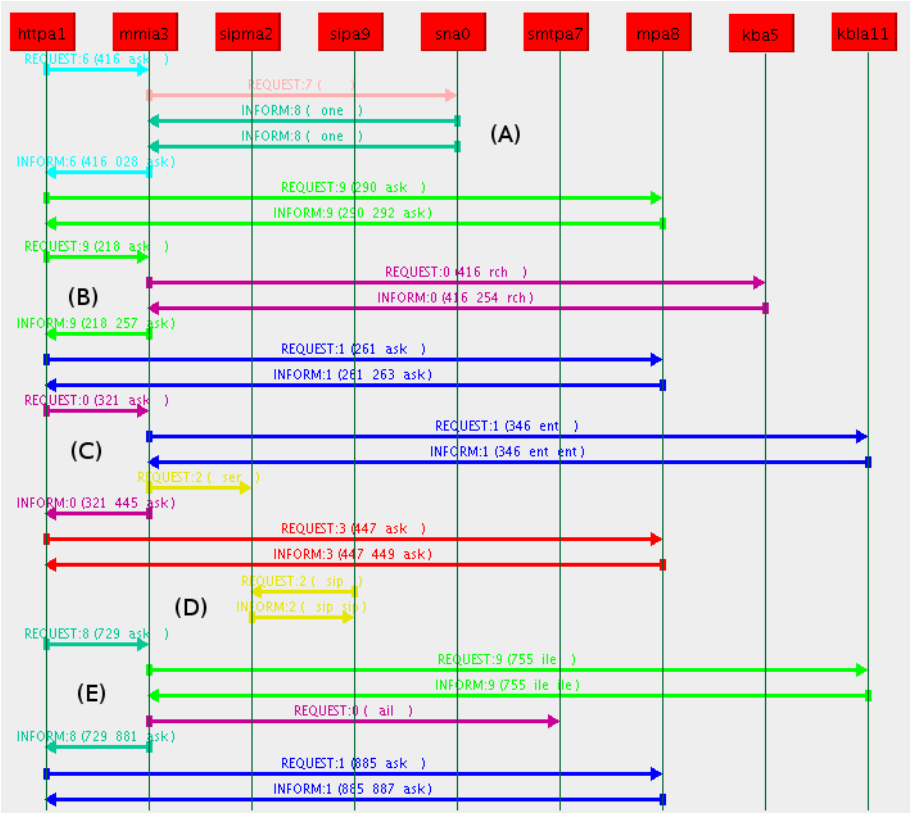


Figure 6.12: Platform multi-modal interactions

The different stages of the interaction are handled as follows:

- Figure 6.12 (A) - This interaction is undertaken with the web interface (i.e. via the HTTPA agent) as the primary mode of interaction. This particular stage of the interaction is after the user has logged into the platform: and, in communication with the Social Networking Agent (SNA), the MMIA agent authenticates the user of the platform. Of the two messages sent from the SNA agent to the MMIA agent, the first one is validating the authentication credentials supplied by the users, and the second one updates the MMIA agent with the information about the user's alternative channel details (e.g. email address, phone number)

from the user's profile. The response to the HTTPA agent is handled in a standard manner, with the MPA agent rendering the content accordingly.

- Figure 6.12 (B) - This leg of the interaction shows a standard request for knowledge from the platform, handled by the Knowledge Base Agent (KBA), and rendered by the MPA agent. Therefore this interaction is totally uni-modal through the web interface.
- Figure 6.12 (C) - In this stage of the interaction, the user triggers an interaction through a VOIP channel (i.e. by requesting, through the web interface, that the content be rendered as audio). The MMIA agent activates the VOIP channel by sending a message to the SIP Manager Agent (SIPMA).
- Figure 6.12 (D) - The SIPMA agent in interaction with the SIP Agent (SIPA) handles the processing of the output through the VOIP channel.
- Figure 6.12 (E) - The last leg of the interaction shows the utilization of the SMTP agent (SMTPA) to present the result of the request as an email attachment. The first part of this request is handled by the KBLA agent, which provides access to local files on the platform.

This interaction highlights the manner in which multi-modality is handled and provided for in the platform. The implementation of KnowNet as a MAS, is tightly aligned with the current trend of implementing multi-modal User Interface (UI) systems as MAS's [195].

6.3.1.6 Interaction

The utilization of KnowNet decomposes down to a series of interactions between the users and the platform. These interactions can be visualized as a graph, where at any point in the interaction, there are a number of paths that the user can take. For example, after the users have authenticated (an action node on the graph) on KnowNet, they can follow an interaction path to browse the available knowledge bases, upload new content, or logout of the knowledge platform (Fig. 6.13). The MMIA agent is responsible for determining the interaction paths available for the user based on a number of factors: their current location, their authentication status, their authorization permissions, and other internal conditions on the platform.

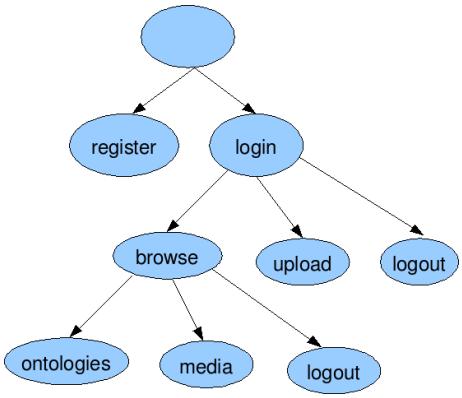


Figure 6.13: Interaction paths

Consistently with the modularity design goal of the platform, the underlying interaction graphs for different requests are the same and all get processed by the MMIA agent. They, however, get handled differently per end-user devices. The interaction with HTTP based devices has matured in its own right through the use of hyperlinks and HTML form elements. It is also a relatively user-friendly interaction mechanism. On the other hand interaction with VOIP devices can be implemented through simple DTMF grammars or through the use of Automatic Speech Recognition (ASR) engines integrated into the VOIP PBX platform (in this case Asterisk). At the time of undertaking this research, the maturity of the available Free and Open Source Software (FOSS) ASR tools is not at a stage of deployment in as heterogeneous a context as that in focus in this research. Most of these tools (e.g. SPHINX) require extensive prior training by the users. DTMF therefore remains an applicable interaction method for implementation on the VOIP channel, albeit the downside of complexity which reduces usability.

We have developed and implemented an interaction mechanism for the VOIP devices, motivated by GUI mouse gestures, wherein the users can utilize the layout of the phone keys to undertake standard operations (interactions) within KnowNet. The motivation for this is to increase the ease of use of the DTMF interfaces through standardized, easy to remember interaction patterns. The following are the interactions that are available to the user at any point in their utilization of the platform (Fig. 6.14):

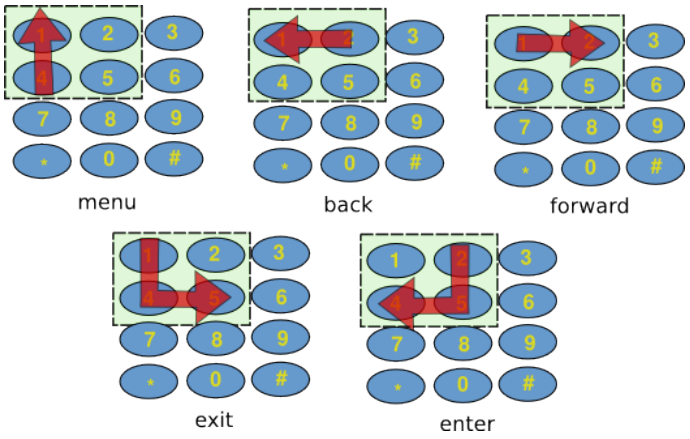


Figure 6.14: Phone gestures

The phone gestures have been implemented to be active four the four keys (1,2,4 and 5) shown in the diagram. As defined in the DTMF grammar, a gesture is activated by pressing *star* (*) followed by the gesture keys. For example, the *menu* option is accessed by pressing *star* (*), *key four* (4) then *key one* (1).

1. Menu: this takes the use to the main menu options for KnowNet
2. Back: this allows the user to navigate to the previous item, in their session history or a previous item in a list of subject item. For example if a user is navigating the list of accommodation options (i.e. items with *rdfs:subClassOf* property of *dwesa:Accommodation*), the back option takes the user to the previous accommodation option in the list.
3. Forward: this takes the user to the next item in the list being currently navigated.
4. Exit: this logs the user off the system and exits from the platform.
5. Enter: during an enumeration of menu option prompts, this allows the user to select a specific menu choice.

This menu structure is implemented for VOIP devices through a VXML DTMF grammar which defines the options, and also the corresponding voice commands for a seamless integration with an ASR engine in the case of one being available within the PBX platform. Further (and more complex) gestures can be implemented by extending the DTMF grammar.

Listing 6.1: KnowNet interaction VXML grammar

```
<grammar type="text/x-grammar-choice-dtmf">
```

```
    ...
*145# | *145 {exit}
*254# | *254 {enter}
*41# | *41 {menu}
*21# | *21 {back}
*12# | *12 {forward}
    ...
</grammar>
```

A similar level of interaction, as far as interaction path options are concerned, is provided for the users who access KnowNet through the web browser interface (Fig. 6.15). Hyperlinks are used to render the different options to the users, and the MMIA agent handles the requested interaction path.

Figure 6.15: KnowNet browser interaction paths

The functionality and the role of the MMIA agent makes it a critical agent on the platform, in a sense that it implements handlers for the requests from the access agents and provides a logical interface to the rest of the platform. The functionality currently implemented in the MMIA agent can easily be extracted and implemented in a number of separate agents to cater for scalability and performance requirements on the platform. The MMIA agent is cognizant of most of the agents in the platform, with which it achieves the handling of the end-user requests. The MMIA agent is aware of the knowledge base layer agents to forward the handling of the requests to, the access layer agents for establishing a communication channel with the users, and the social networking agent for the functionality associated with AAA within the platform.

6.4 Access agents

Access to the the platform is facilitated by the access layer agents whose role is to implement the specific transport level protocols for communication with end-user devices and other software entities (i.e. as in the case of web service clients). The access layer agents provide the ability to handle the multiplicity of the end-user devices in an extensible manner and in a way that is

separated from any other operational structure of the platform. An access layer agent must be aware of interaction and presentation agents, with which it communicates to handle end-user's requests and render the response appropriately for the requesting device. The following are the access layer agents currently implemented on KnowNet:

6.4.1 SIPA

The Session Initiation Protocol Agent (SIPA) provides access to the platform via the FastAGI interface on the Asterisk PBX (Fig. 6.16). This allows for a telephony interface into KnowNet and for audio based interaction from the platform.

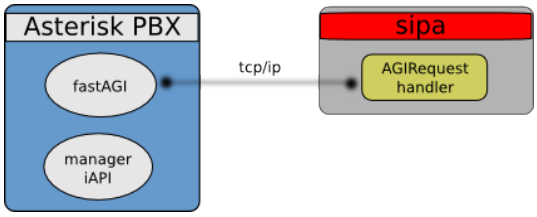


Figure 6.16: SIPA agent

The fastAGI interface on Asterisk provides a way to implement scripts on a remote machine (a fastAGI server) which get executed when a corresponding extension is dialed. The *sipa* is therefore in essence a fastAGI server that dynamically generates the scripts that are executed on Asterisk (which is the actual transport layer entity in the setup). An extension is defined within the *extension.conf* file on the Asterisk server as show in Listing 6.2, and a mapping is created on the platform server to map piask.agi to the sipa agent (Listing 6.3).

Listing 6.2: SIPA agent Asterisk extension definition

```
exten => 7777,1,AGI(agi://146.231.121.160/piask.agi)
```

Listing 6.3: fastagi-mapping.properties file on the KnowNet platform

```
piask.agi = com.dwesa.piask.SIPA
```

The *sipa* agent also interacts with the *sipma* agent in handling KnowNet initiated events to the end-user extensions. An example of such an event is a content that has to be communicated to a user, or a request for input from the user (Fig. 6.17). Such an event can initiate from any of the agents in the platform (e.g. an agent that implements a scheduling service initiating a reminder event to a user's extension) and gets handled by the interaction agent:

1. The *interaction agent* sends a message to the *sipma agent* - the message encapsulates the extension to initiate a call to and the message to communicate to the user
2. The *sipma agent* initiates a call to the stated extension, and stores the message in a message queue for *sipa agents*
3. An incoming call from the extension is handled by the *sipa agent*
4. The *sipa agent* sends a message to the *sipma agent* requesting the queued messages for the calling extension
5. The *sipma agent* sends the queued message back to the *sipa agent*
6. The *sipa agent* plays the message to the user and sends back a response to the *sipma agent.* For a notification event, the response is either success or failure, but for events requiring input from the user, the response is the requested user input.
7. The response is forwarded back to the requesting agent via the *interaction agent*

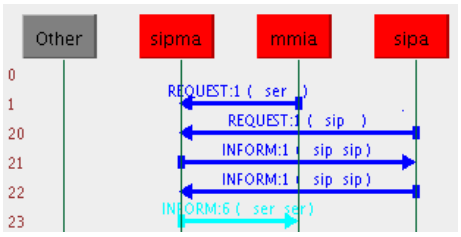


Figure 6.17: SIPMA and SIPA agents interaction

6.4.2 SIPMA

The SIP Manager Agent (SIPMA) allows access to the manager interface of the Asterisk VOIP PBX (Fig. 6.18). The manager interface allows for execution of commands on Asterisk and for handling events initiated from the PBX. This provides such functionality as initiation of calls from within KnowNet to different end user extensions (Fig. 6.17), to allowing for implementation of specific functionality in response to an Asterisk event of an extension registration. The SIPMA agent allows KnowNet to have a low level control over the PBX in a manner that enhances the execution of the user's requests, also providing an interface to the rich set of manager commands that can be executed on the PBX. A simple utilization of the SIPMA agent, in the context of collaboration with the MMIA agent and SIPA agents is depicted above (Fig. 6.17).

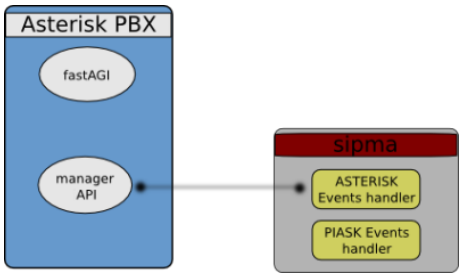


Figure 6.18: SIPMA agent

6.4.3 HTTPA

The HTTP transport layer is implemented and handled within KnowNet by the Hyper Text Transfer Protocol Agent (HTTPA) (Fig. 6.19). The *httpa* handles request from web browsers but also handles requests from the VXI VXML browser, which runs as an Asterisk application [196]. The *httpa* agent in essence provides a web server service that receives requests from any HTTP compliant browsers. The *httpa* agent embeds a Jetty web server, which is an open source, full-featured, scalable and robust server implemented in Java [197].

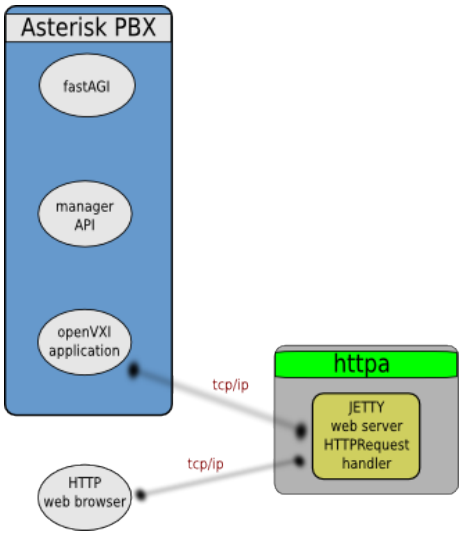


Figure 6.19: HTTPA agent

Besides handling the HTTP requests from web browsers, *httpa* also handles VXI VXML browser requests from Asterisk. This provides an alternative access from Asterisk and also a richer set of constructs available within VXML mark-up language which are not available in AGI scripting. While the VXI request is handled akin to any other HTTP request, on the Asterisk side

a mapping has to be made from an extension to the platform in the *extensions.conf* file (Listing 6.4)

Listing 6.4: VXML browser extension on Asterisk

```
exten => 6663,1,Answer
exten => 6663,2,Wait(3)
exten => 6663,3,Vxml(http://146.231.121.160:8080)
exten => 6663,4,Hangup
```

The HTTPA agent is responsible for the device specific session handling across device requests. This means that the HTTPA agent maintains a list of the devices that are accessing the platform and makes an association between the different request, in this case using a simple unique ID. This is different from session handling as provided by the MMIA agent, the HTTPA agent provides the more end-devices/request associated session handling, while the MMIA agent provides more of the user specific session handling, which could spreads across different devices.

6.4.4 XMPPA

IM has become a prominent communication channel for online users, and also a key mechanism for the implementation of presence services. The eXtensible Messaging and Presence Protocol Agent (XMPPA) provides the primary access mechanism for the provision of IM based interaction with the knowledge platform (Fig. 6.20). The XMPPA agent is implemented via the SMACK XMPP API, which allows for the association with an XMPP based IM servers (e.g. jabber.org, Google Talk) [198]. Alternatively the XMPPA agent can associate with a local IM server, as is the case in the current implementation of KnowNet, wherein a local OpenFIRE server has been deployed. Each of the different access layer agents have specific areas and types of interaction that they are suited for, and also different interaction constraints within which to operate. XMPP is best suited for handling simple instructions to the knowledge platform from the users, such as requests to send a file to a user, or to establish a call to an extension. It is not suited, for example, for browsing a large amount of data encapsulated in the knowledge platform.

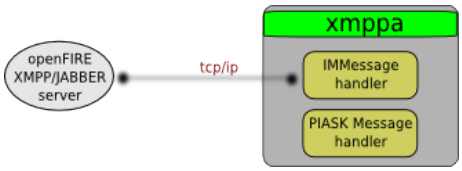


Figure 6.20: XMPPA agent

The XMPPA agent provides valuable presence information to the platform that is used by the MMIA agent in establishing a communication channel with a user. In cases where the user's status is *available* or *busy* as ascertained from the XMPPA agent, the MMIA agent can send information directly to the user chat client, or choose an alternative channel (e.g. email) respectively.

6.4.5 SMTPA

The Simple Mail Transfer Protocol Agent (SMTPA) implements a channel for communication with the users via email. The SMTPA agent (Fig. 6.21) agent provides a complementary channel for the knowledge platform, that is utilized predominantly for sending requested information files as attachments and handling basic notifications to the users. While the current implementation of SMTPA agent provides a largely uni-directional interaction from KnowNet to the users, the SMTPA agent can forward IM type instructions through the MMIA agent for processing by the NLP unit within the KBA agent. The SMTPA agent handles and provides asynchronous communication between the platform and the users.

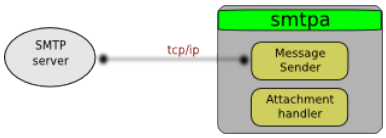


Figure 6.21: SMTPA agent

6.4.6 PWSA

One of the key requirements for the knowledge platform within the context of the ICT4D research undertaken in Dwesa is to provide a knowledge layer within the deployment stack for eServices. This allows for the development of services that exploit the added benefits of the knowledge platform, such as the ability within the eCommerce service portal for local arts and crafts to access the available knowledge about the artifacts for sale and about the artists and the entrepreneurs.

Within the service provision domain a number of standards have evolved for service description, service discovery and message transportation between the service providers and the service consumers. These key standards are Web Service Description Language (WSDL), Universal Description, Discovery and Integration (UDDI) and Simple Object Access Protocol (SOAP) respectively. WSDL allows for a description of a service, specifying the actions that are offered by

the service, the messages that are passed to the different actions, and the specification of the service address. UDDI is a standard that allows for the discovery of the different services based on their WSDL. SOAP messages encapsulate the request and responses in the process of executing a service.

Within the JADE MAS platform, there exists a mechanism, through a Web Services Integration Gateway (WSIG) add-on, to expose the behaviours of different agents as web services. This is achieved through the operation of the WSIG servlet and the WSIG agent. The WSIG servlet handles the requests from web services (WS) consumers, unpacks the SOAP messages and forwards them to the WSIG agent. The WSIG agent on the other hand, joins the JADE MAS and registers with the DF to be notified when agents join and leave the platform. The behaviours of the agents on the platform are exposed through an agent ontology (Appendix D) that describes the actions (*AgentAction*) that are handled by the agents. The WSIG agent is responsible for handling the communication with the WSIG servlet, parsing the messages and forwarding encoded *ACLMessages* to the appropriate agents on the MAS. The WSIG agent is also responsible for creating a WSDL for the different agent actions that are defined and registered in the agent ontology.

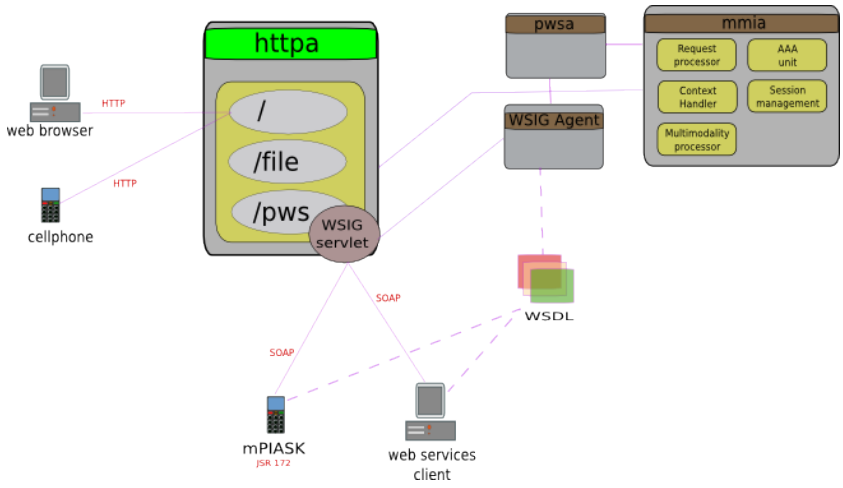


Figure 6.22: KnowNet Web Service architecture

The WSIG add-on had been utilized within KnowNet to expose the behaviours available in the platform agents (Fig. 6.22). The HTTPA agent defines a number of key server contexts that are available on the platform. The root context (”/”) handles the request from web clients and from the VXI browser for knowledge services (Section 6.4.3). The file context (”/file”) allows for accessing of public files from the platform that are utilized by the requesting devices (e.g. the background graphics that are utilized for an HTML interface). The PIASK web services (PWS)

context with path (”/pws”) allows for handling of web services requests. The PWS context is defined as a web application context and primarily operates as a servlet container for the WSIG servlet. On handling the communication with the WSIG servlet, the WSIG agent interfaces with the PIASK Web Services Agent (PWSA).

The PWSA agent plays a critical role in providing an interface into the different services that are available within the platform. The PWSA agent is currently the only agent that makes its ontology (i.e. *PIASKOntology*) available for processing by the WSIG agent. The PWSA agent aggregates the functionality that is available within the platform, primarily through the MMIA agent, and exposes that for a WSDL to be generated from, and for the services to be requested from. The PWSA agent therefore handles all the WS requests into KnowNet (as standard ACLMessages defined around the *PIASKOntology*), generates the corresponding platform data messages (Section 6.7) and forwards them to the MMIA agent which provides the primary logic for handling of requests on KnowNet (Section 6.3). On receiving the responses from the MMIA agent, the PWSA agent unpacks the message and encodes it as per the *PIASKOntology* and sends it back to the WSIG agent. The PWSA agent, within the role of handling communication with Web Service clients, therefore operates within the Access tier of the PIASK architecture.

The WSDL of the available KnowNet actions is accessible to WS clients through a request to the PWS context on the HTTPA agent. Within the current setup of the platform, there is no use of directory services through UDDI, and therefore the WS clients directly access the WSDL on the platform and formats the requests accordingly. This mechanism allows for the knowledge services available within the platform to be accessed as web services for utilization in the deployment of eServices portals. A WSDL specification of the KnowNet services exposed through the PWSA agent is in Appendix E.

6.5 Social networking agents

The social networking agents play the pivotal role of positioning the knowledge platform within the social context by integrating the existing social dynamics within the community. The role of social networking agents is pivotal in the knowledge platform towards the overall objective of achieving both a context sensitive and culture sensitive knowledge platform. The other key role of social networking agents is based on the realization of the distinct knowledge types that are available, and of the fact that some knowledge is better contained in computer based knowledge bases, while other knowledge is still best carried by the human agent. For example, tacit knowledge is currently only available through human beings and not storable in external knowledge

bases. The platform's interface with the human agents (which in one sense are the users of the system, but in another sense also active participants in the life of the community of agents) is facilitated by the social networking agents. The current implementation of the KnowNet knowledge platform consists of a single social networking agent (SNA).

6.5.1 SNA

The SNA agent interacts primarily with the MMIA agent from which it typically receives requests associated with the users of the platform (Fig. 6.23). Examples of such interactions include the request from the MMIA agent to process an authentication request, or the determination of specific profile information of the users. For example, if a user's session was through a HTTPA agent (i.e. via a web browser) and the MMIA agent determines to reroute the user's interaction to a VOIP session, the request will come through to the SNA agent to provide the associated user's VOIP extension.

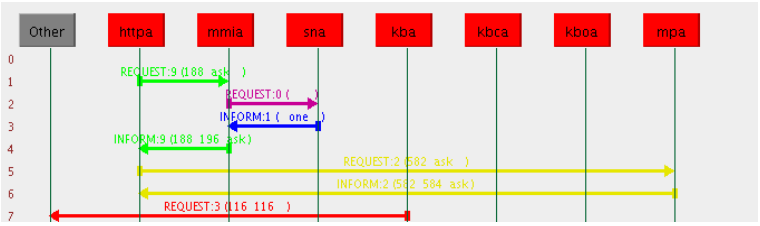


Figure 6.23: SNA agent interactions

The SNA agent provides the primary community/user orientation in the platform by being aware of:

1. The users of the platform.
2. The individual preferences of the users associated with the UI components and interaction modalities (e.g. themes, UI language).
3. The relationships between the users.

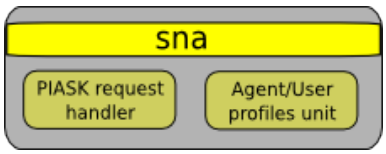


Figure 6.24: SNA agent

The SNA agent therefore encapsulates two key modules, the request handler module, and the user's profile module (Fig. 6.24). The user profile module directly interfaces with the underlying knowledge base layer agents to query the user profile ontologies. This agent is therefore functionally dependent on the knowledge base agents which are discussed in the next section.

6.6 Knowledge base agents

One of the core layers of the knowledge platform is the knowledge base layer, which is the end-point of most requests that are handled by KnowNet. The agents that have been implemented within this layer are primarily responsible to interfacing with different knowledge bases and knowledge repositories, accessing the requested knowledge and forwarding it back to the requesting users. While most of the knowledge that KnowNet is targeted to handle is encapsulated in ontologies, there is also a large amount of information that is stored in other types of knowledge bases: knowledge encapsulated in folksonomies, DataBase Management System (DBMS) based knowledge, and native file system based knowledge. The required flexibility on the knowledge platform, necessitates the implementation of agents that naturally handle the different knowledge bases (Fig. 6.25).

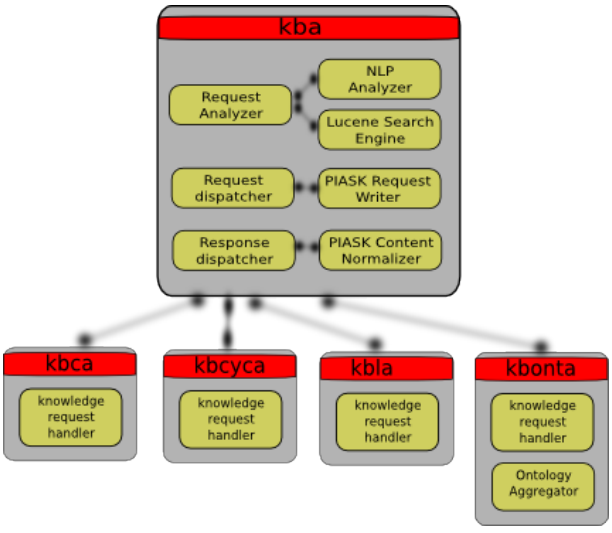


Figure 6.25: Knowledge base layer agents

6.6.1 KBA

The Knowledge Base Agent (KBA) acts as the manager agent for the rest of the knowledge base layer agents. All the requests that are send from the MMIA agent to the knowledge base layer are send directly to the KBA agent, which processes the request to determine the appropriate agent to forward the request to. The following are the modules within the KBA agent:

1. Request analyzer: The request analyzer module is the first to receive the requests from the MMIA agent. Its main responsibility is to process the requests, which are heterogeneous in nature. First of all it has to determine the nature of the knowledge that is requested and which ontology encapsulates the required information. This directly determines which agent the request gets forwarded to. It achieves this through interaction with the NLP analyzer module and a search engine module. The request analyzer also extracts the specific instructions that have been encapsulated into the request message by the MMIA agent and forwards this for processing to the request dispatcher.

2. NLP analyzer: Due to the heterogeneity of the requests that come into KnowNet, it is necessary to provide a multiplicity of request handlers. Some of the requests that KnowNet handles are through the XMPPA access agent which provides an IM interface into the knowledge platform. The NLP analyzer allows for the processing of natural language requests, with the end goal of determining the best agents to forward the requests to. The function of the NLP analyzer is executed in close operation with the search engine module.

3. Search engine: The large amount of knowledge that is available within KnowNet needs to be indexed for easy and quick processing by the knowledge agents. The search engine module utilizes the Lucene search API to implement an index of all the data that KnowNet has access to (this is discussed in detail in Section 6.6.5). This module is responsible for building an index for every new information that is added and also for searching the index to find the best hits for the requests.

4. Request dispatcher: This module is responsible for sending the request to the appropriate agents. This module forwards appropriately formatted requests to the underlying ontologies. The requests need to be formatted to the specific inter-agent message format, that the other knowledge agents can process (Section 6.7). This module therefore operates in close proximity with the PIASK request writer module.

5. Response dispatcher: In handling the request for information, the KBA agent at times needs to forward the request to a number of agents. The response dispatcher aggregates all

the responses that have been received, gets them normalized accordingly and sends them back to the MMIA agent.

6. PIASK request writer: The underlying knowledge agents in KnowNet handle messages that have been formatted in a specific manner. An example formatted request for the KBONTA agent would encapsulate the specification of the ontology from which the information is requested, the subject requested and sometimes the associated predicate as mapped to the <*s,p,o*> struct of the triples in the ontology. It is the responsibility of this module to format the requests properly.

7. PIASK content normalizer: This module formats the content that has been received from the knowledge layer agents so that it is processable by the Media Presentation Agent (Section 6.2). The fulfillment of the need for the platform to handle multi-media content that can be transcoded to many different formats, is dependent on this preliminary mechanism of normalizing all the content to a *lingua franca* defined within KnowNet.

An example of handling users requests and how the different KB modules work is shown in Figure 6.26.

The request analyzer receives a natural language request from the MMIA agent (possibly initially from the XMPPA agent), which gets stripped down into basic tokens by the NLP Analyzer. The search engine identifies the associated ontology, the subject within that ontology and, where available, the required predicate. The request dispatcher then sends the formatted (by the request writer) request to the knowledge agents. Once a response is available from the knowledge agents, it get normalized to platform standard content format, and then send back to the requesting MMIA agent.

6.6.2 KBONTA

This is an agent that aggregates access to all the ontologies that have been loaded onto the knowledge platform. This agent abstracts access to the knowledge encapsulated in the different ontologies and is the primary handler of user requests. The Knowledge Base ONTology Agent (KBONTA) comprises two basic modules: the ontology aggregator - which automatically searches the underlying knowledge base for new ontologies; and the request handler - which receives the requests from the KBA agent, determines the required ontology, subject and, where available, the predicate.

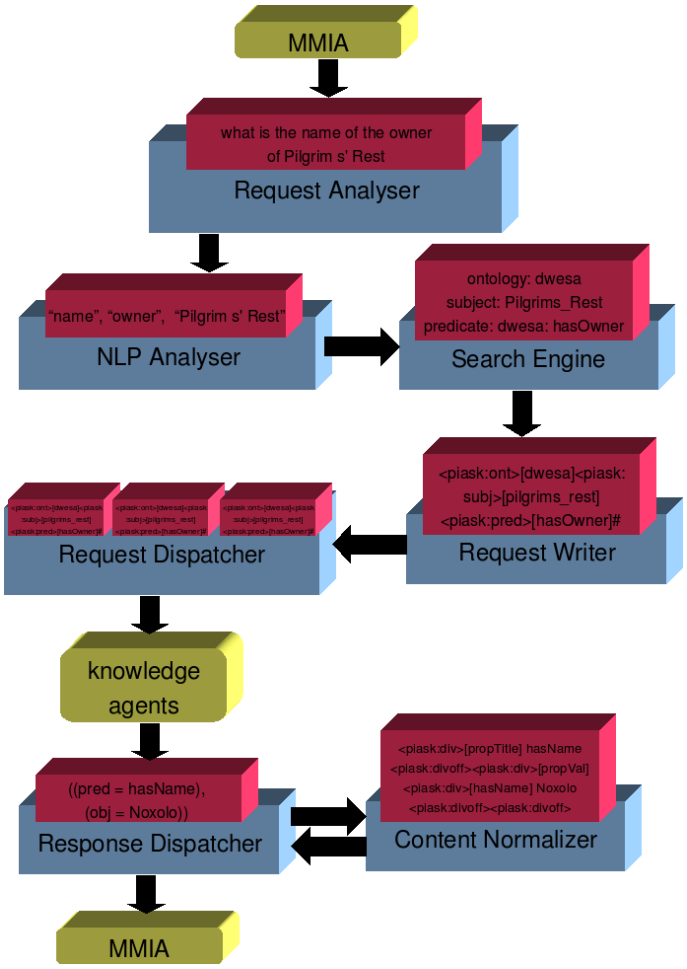


Figure 6.26: KBA agent modules

6.6.3 KBLA

The Knowledge Base Local Agent (KBLA) handles the processing of files associated with the local knowledge on the platform. This agent provides access to the local files (i.e. documents, images, audio, and video) and allows for adding new files, modifying and updating the existing files, and deleting files from the knowledge platform. The KBLA agent interacts with the AAA module of the MMIA agent to implement access control, and authorization permissions on the underlying knowledge in the system. The KBLA agent uses the deployment platform's native file-system to store the files, and the PIASK ontology (Section 5.1.5) to store the information associated with the file.

6.6.4 Inference on the knowledge base

One of the key motivations and advantages of utilizing ontologies for the encapsulation of knowledge within the platform is the ability to infer implicit knowledge from the underlying explicit knowledge. The ontology layer in KnowNet is implemented through the JENA API. JENA provides the following reasoners for inference [199]:

1. Transitive reasoner - this reasoner provides the functionality to traverse a class based on the transitive and symmetric properties of *rdfs:subPropertyOf* and *rdfs:subClassOf* properties.
2. RDFS rule reasoner - allows for reasoning based on RDFS entailments.
3. OWL reasoner - implements the OWL-Lite rules for inference.
4. DAML micro reasoner - a support of legacy DAML inference.
5. Generic rule reasoner - this is a generic reasoner that allows for user-defined rules.

The full and effective utilization of these reasoners is dependent on the underlying formalization of the ontologies and this has been specified in Section 5.1. The added advantage of the reasoner can be illustrated in the following example.

Within the PIASK ontology which is used by the KBLA agent (Section 6.6.3), the different documents can be of types *dwesa:Audio*, *dwesa:Video*, *dwesa:Image*, and *dwesa:Text* where dwesa is namespace *http://www.dwesa.com/piask*. These resource types are all subclassed from the *dwesa:Document* resource (e.g. <*dwesa:Audio rdfs:subClassOf dwesa:Document*>) (Section 5.1.5). Querying the ontology for all the documents that have been added, the following RDQL statement (implemented in JAVA) would be constructed (Listing 6.5).

Listing 6.5: RDQL PIASK ontology statement

```
String statement = piask.defaultprefixes +
                   "select ?subj " +
                   "where {?subj rdf:type dwesa:Document}";
```

This statement returns the value of the subject for the triples where the property is *rdf:type* and the object is *dwesa:Document* (Listing 6.6). The *piask.defaulprefixes* variable is an enumeration of the ontology namespace prefixes that are used within KnowNet (e.g. rdf:, owl:, dwesa:).

Listing 6.6: RDQL query results without inference

```
?subj = <http://www.dwesa.com/piask/file.tmp>
?subj = <http://www.dwesa.com/piask/frog3>
?subj = <http://www.dwesa.com/piask/3file.tmp>
?subj = <http://www.dwesa.com/piask/4file.tmp>
?subj = <http://www.dwesa.com/piask/5file.tmp>
?subj = <http://www.dwesa.com/piask/6file.tmp>
```

The above results show the 6 files for which the type has been explicitly specified as *dwesa:Document*. This is implemented to cater for cases where the document type cannot be ascertained accurately.

Initializing an inference model on the platform is achieved as shown in Listing 6.7. The first line in the code gets the ontology model from an *arraylist* of all the models that have been loaded on the platform. An inference model is then created from that ontology model (the last line in the code).

Listing 6.7: KnowNet - initializing inference model

```
/* initializing the models */
Model piask = ((ontModel)knows.get(knows.indexOf(new
                ontModel("piask", null)))).getModel();
Model infpiask = ModelFactory.createRDFSModel(piask);
```

The results that are observed after executing the same RDQL statement (Listing 6.5) on the inference model are indicated in Listing 6.8.

Listing 6.8: RDQL results with inference

```
?subj = <http://www.dwesa.com/piask/file.tmp>
?subj = <http://www.dwesa.com/piask/frog3>
?subj = <http://www.dwesa.com/piask/3file.tmp>
?subj = <http://www.dwesa.com/piask/4file.tmp>
...
?subj = <http://www.dwesa.com/piask/circles.png>
?subj = <http://www.dwesa.com/piask/facebook.png>
?subj = <http://www.dwesa.com/piask/new.jpg>
?subj = <http://www.dwesa.com/piask/another.jpg>
```

```
?subj = <http://www.dwesa.com/piask/0wedmap.jpg>
?subj = <http://www.dwesa.com/piask/unaml.jpg>
?subj = <http://www.dwesa.com/piask/w001.jpg>
?subj = <http://www.dwesa.com/piask/sleep-eating.jpg>
...
```

The query on the RDFS inference ontology returns 27 documents, which is the number of documents that have been added to the platform. In this case, the execution of the statement is being performed on a model based on the *rdfs:subClassOf* property inference.

6.6.5 NLP and ontology indexing

Part of the process of handling user requests is passing them through a NLP engine and a search engine. This process allows for faster determination of the associated ontology, subject and predicate. Before this functionality can be accessed, an index of the underlying ontologies has to be created and updated every time new information is added to the platform. KnowNet utilizes the Lucene search engine both for the purpose of NLP and ontology indexing.

6.6.5.1 Lucene search engine

Lucene is a text search engine implemented in JAVA to provide search functionality within applications [200]. The process of implementing the search functionality with Lucene begins with the building of an *Index*. An *Index* is a collection of *Documents* which consist of one or more *Fields* [201]. Each *Field* is a <*name, value*> pair *(<n,v>)* that represents the searchable elements of the index (Fig. 6.27). Constructing an index involves transforming the data that needs to be indexed into *Documents* that a Lucene *Indexwriter* can process. The source of this data can be a text file, a database, a multimedia file or any content that can be organized into a set of *Fields* within a *Document*.

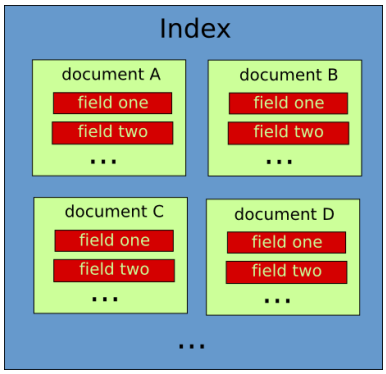


Figure 6.27: Lucene index structure

6.6.5.2 Ontology to search index mapping

In building an index for the knowledge that is being processed within the KnowNet platform, there is need to do a mapping of the underlying ontologies onto the Lucene *Index* structure. The basic structure of every ontology that is being utilized in KnowNet is a collection of <*subject, predicate, object*> *(<s,p,o>)* triples (Fig. 6.28).

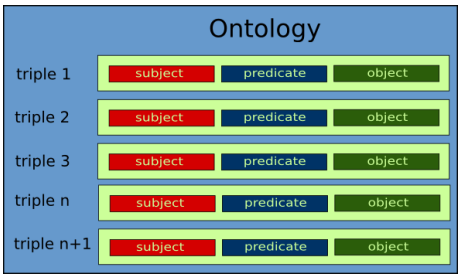


Figure 6.28: An Ontology Structure

There are various alternatives to mapping the <*s,p,o*> triple structure of the ontologies to the <*n,v*> structure for the *Index*. One such alternative would be to adopt an object-oriented view (in contrast to the statement/triples view) of the underlying knowledge (Fig. 6.29). In this view, every *Document* would map to a subject on the ontology, and the associated *Fields* per document would map to the predicates associated with a subject, a field name would therefore be the value of the associated predicate, and the value would be the associated object.

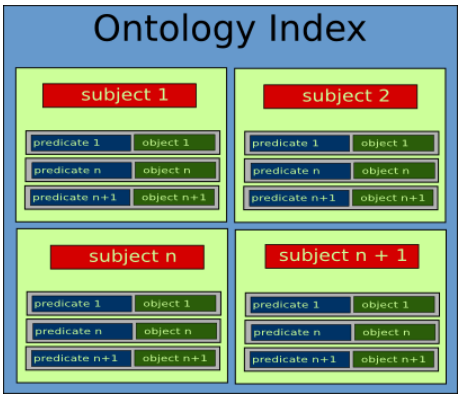


Figure 6.29: "Object-Oriented" Ontology mapping

Another mapping strategy and one that has been implemented in the platform is the statement-centric view of the ontology. In this view every triple maps onto a *Document*, and the <*s,p,o*> elements map onto the *Fields* of the document (Fig. 6.30). This strategy is implemented in KnowNet as recommended from the Lucene documentation [200].

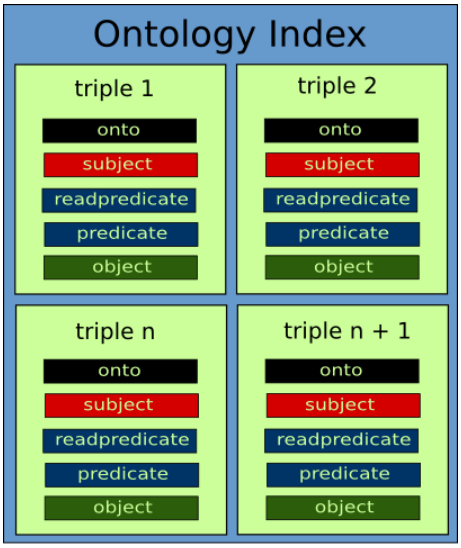


Figure 6.30: Statement-Centric Ontology mapping

This index structure therefore goes through all the triples and creates a *Document* per triple, with the associated *Fields*. The extra *Fields* that have been added for each *Document* are:

onto this field stores the value of the current ontology that the triple occurs in.

readpredicate this field allows for the tokenization of predicates into units of processable strings. This helps in the execution of searches on the predicate field of the ontology. For example if one has a predicate *http://www.dwesa.com /piask/hasEmailAddress*, which associates an email address literal to a resource of type *http://www.dwesa.com/piask/Person* then the predicate field will evaluate to "*http://www.dwesa.com/ piask/hasEmailAddress*" and the readpredicate field evaluates to the two tokens "*email*" and "*address*". The searches on the index then get performed on the readpredicate field, and still allows access to the URI of the fully qualified predicate.

The implementation of this ontology to index mapping is achieved as follows:

```
ResultSet results = qe.execSelect();
while(results.hasNext()){
   QuerySolution binding = results.nextSolution();
   Document doc = new Document();
   Field one = new Field("onto", "dwesa", Field.Store.YES,
                                          Field.Index.TOKENIZED);
   doc.add(one);
   one = new Field("subject", binding.get("subj").toString(),
                     Field.Store.YES, Field.Index.TOKENIZED);
   doc.add(one);
   String pre = binding.get("pred").toString();
   pre = pre.replaceAll("http://www.dwesa.com/piask/", "");
   one = new Field("predicate", pre, Field.Store.YES,
                                     Field.Index.UN_TOKENIZED);
   doc.add(one);
   pre = pre.replaceAll("_", " ");
   one = new Field("readpredicate", pre, Field.Store.YES,
                                    Field.Index.TOKENIZED);
   doc.add(one);
   one = new Field("object", binding.get("obj").toString(),
                       Field.Store.NO, Field.Index.TOKENIZED);
   doc.add(one);
   writer.addDocument(doc);
}
```

Figure 6.31: Implementation of the ontology mapping

The creation of the *Index* allows for use of the following Field options, which provide the flexibility to control the size of the *Index* and the level of tokenization in the *Index*.

1. Field.Store.YES - The current field gets stored in the Index
2. Field.Store.NO - The current field does not get stored in the Index
3. Field.Index.TOKENIZED - The field gets tokenized before being processed. In some cases it is important to keep the field intact before processing, and in those cases the option would be Field.Index.UN_TOKENIZED.

6.7 The communication protocol

The functioning of the platform is facilitated by the communication that happens within the community of agents. The JADE MAS platform inherently provides an Agent Communication Language (ACL) messaging mechanisms as part of the FIPA specification compliance. The KnowNet communication protocol is implemented on top of this message system, for the exchanging of platform specific messages between the agents. The FIPA specification provides a number of features for facilitation of communication between agents and for differentiating the messages processed:

1. Primarily message are send to specific agents by specifying the receiver associated with every ACLMessage (e.g. *aclmessage.addReciever(x)*).

2. The message performatives[3] can be used in a conversation to identify the different stages in the conversation. For example, the initial request for information that is sent from one agent to another could carry the *ACLMessage.REQUEST* performative, and the response to that request would have the *ACLMessage.INFORM* performative. Different filters based on the message performative, within different agent behaviours, can then be used to limit and identify the messages that are handled from the message queue.

3. The conversation ID can be used to identify messages that belong to a specific conversation between the agents. This ID can be used as well to filter the messages for handling by different agent behaviours.

4. The replyWith parameter specifies the string that the expected response should contain, thus allowing the requesting agent to identify the messages that are a response to a send request.

Within KnowNet, a combination of the above message identification features is used. Figure 6.32 is an illustration of message passing between some of the agents in the system. For example, the messages that are sent from the *httpa* agent to the *mmia* agent for handling user's requests would carry a request performative (i.e. *ACLMessage.REQUEST*), have the *replyWith* and *conversationID* parameters set with *aclmessage.setReplyWith("piask")* and *aclmessage.setConversationId("piask:httpa-<currenttime>")* respectively.

[3] A performative is defined within the FIPA specification as a parameter that "denotes the type of the communicative act of the ACL message" [202]

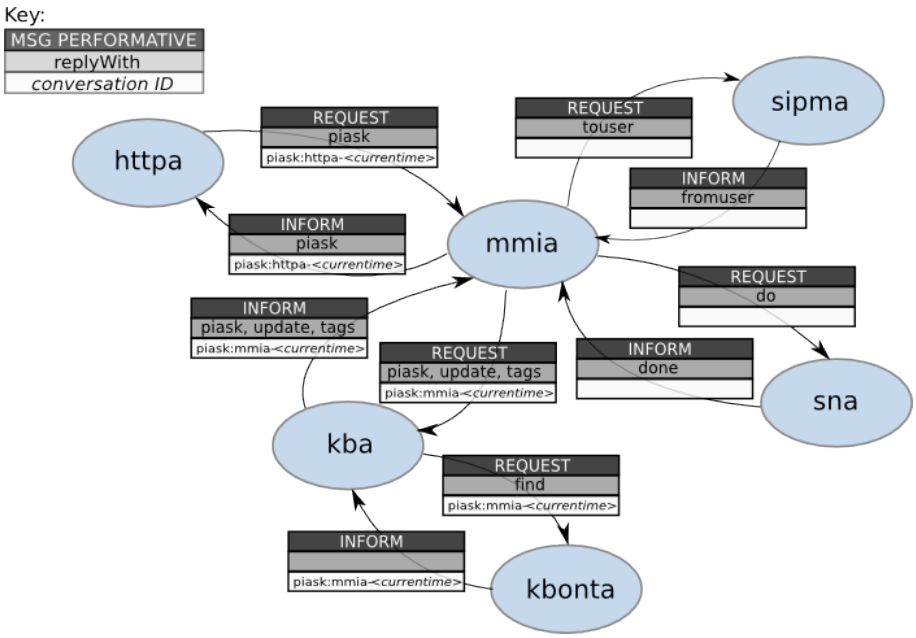


Figure 6.32: KnowNet message exchange

The actual messages that are sent between agents carry a formatting that is illustrated in Figure 6.33.

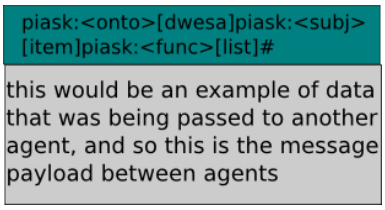


Figure 6.33: KnowNet message payload

The message comprises two sections which are separated by a hash (”#”) character. The first part of the message contains KnowNet specific parameters for passing between the agents, and the second part is the actual message content that is to be processed by the agent. Passing parameters between agents is achieved through the use of platform specific tags. The tags play the role of specifying the message semantics in a manner that is understood by the platform agents. An alternative mechanism that achieves the same purpose would be the use of a platform vocabulary and an ontology that is shared between all the platform agents and utilized to wrap all the communication within the platform. Utilizing basic string tags affords the ease of deployment of new agents on the platform, ease of use, and computational efficiencies associated with the

fact that no processing is required to encode the content into ontology classes and to serialize the associated classes before communicating with the receiving agent.

6.7.1 KnowNet tags

The KnowNet tags are of the form *piask:<param>*, and they can also carry a value specified in the form *piask:<param>[value]*. This system of formatting parameters facilitates the differentiation of platform specific content from the actual content to be processed by the agents, which would normally contain different markup language tags. The following are some of the tags used within KnowNet:

1. *<piask:onto>* - This tag is used to specify the ontology associated with the message that is being communicated. For example, when the KBA agent sends a message to the KBONTA agent, it specifies the ontology associated with the request by including it as the value of this parameter. Therefore in Figure 6.33, the request is associated with the ontology named "dwesa" (*<piask:onto>[dwesa]*).

2. *<piask:subj>* - This tag specifies the subject referred to in the encapsulated message. In Figure 6.33, the subject in the "dwesa" ontology is "item" (*<piask:subj>[item]*).

3. *<piask:pred>* - This is the tag for specifying the predicate associated with the message and the user request. For example, in a case of a user request for "the name of the owner of the Resting Place BnB", this tag would carry the value "hasOwner" (*<piask:pred>[hasOwner]*) within the message passed to the relevant ontology.

4. *<piask:func>* - For the cases where a specific function associated with an agent is required, this tag is utilized. In the communication between the MMIA agent and the SNA agent, this tag can be used as *<piask:func>[set]* or *<piask:func>[get]* to specify whether the SNA agent should update or return the information associated with a subject respectively. In Figure 6.33, the KBA agent is requesting the KBONTA agent to list (*<piask:func>[list]*) the associated triples in the ontology.

5. *<piask:var>* - This is used for the specification of a variable that is communicated between the agents. For example, the session ID is passed between the access layer agents and the MMIA agent through this tag.

The full specification of the tags that KnowNet utilizes is in Appendix D.

6.7.2 KnowNet vocabulary and message ontology

JADE provides a number of ways of facilitating communication between agents on the platform based on the FIPA specification. The messages between the agents can be passed as Strings (see Section 6.7 on the communication between KnowNet agents) that are parsed and processed by the agents. This strategy has the main advantage of being the most direct and easiest to implement. The inherent drawback with it is that it limits the flexibility of the developed MAS, due to the fact that the syntactic and semantic meaning is implicit in the messages between the agents. For example, the KnowNet message tags system (Section 6.7.1) necessitates that the agents communicating understand the syntax of the messages and what the different tags mean. An agent introduced to the system that does not understand these meanings will not be able to communicate with the rest of the agents in the platform. Another strategy that can be utilized in the communication between the agents, is through explicitly specifying the vocabulary and the ontology encapsulating the semantic meaning associated with a MAS platform. Once a vocabulary and an ontology have been defined, the agents simply use that in their communication. This allows other agents (by different developers, in different platforms) to interact within this community of agents. The agents are also able to filter messages and handle only those that are encoded in an ontology that they understand.

On KnowNet, the majority of the interaction between the agents is based on the simple passing of appropriately encoded String messages. A limited vocabulary and an ontology have been defined as an alternative communication protocol within the platform. An example of an interaction that utilizes this ontology is between the PWSA agent and the MMIA agent (described in Section 6.4.6) and the full specification of the ontology is in Appendix D.

6.8 Conclusion

The development of the knowledge platform has provided the initial validation of the viability of the PIASK architecture. The knowledge platform has been realized as a MAS, and the role of different agents and their interaction in achieving the required platform objectives has been discussed. The flexibility achieved in KnowNet platform to handle multi-protocol, heterogeneous requests, also positions it as an adequate and viable technology integration platform. A detailed discussion of the PIASK architectural pattern and the validation of KnowNet knowledge platform is presented in the following chapters.

Chapter 7

Functional and Technical Adequacy of the Platform

”Adequacy is sufficient” (Adam Osborne)

The functional requirement of the KnowNet is to provide a context-sensitive and ethnocentric knowledge-based service platform for use in rural marginalized communities. In this chapter we discuss the ways in which the platform meets this requirement, by highlighting: the operation of handling user requests on the platform; the measures that have been taken towards ensuring confidentiality, robustness and reliability of the platform to meet the service level expectations of the users (Section **??**). We also report the result of the performance evaluation of KnowNet, providing an adequacy validation of the platform as far as the technical and functional requirements are concerned. The last section of the chapter discusses the mechanisms of deploying new services on the platform.

7.1 End-to-end request handling on KnowNet

This section highlights the interaction between the agents on KnowNet, to validate the 5-tier architectural pattern for its adequacy in being a template for the provisioning of complete end-to-end eServices.

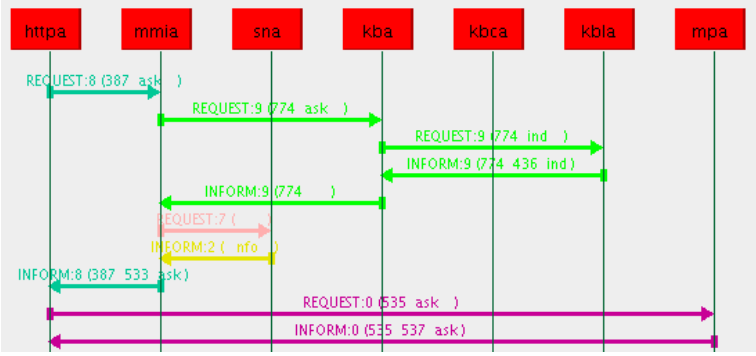


Figure 7.1: End-to-end process handling in KnowNet

The process in focus here is simple browsing of content via a web interface. Other events that have already taken place include the loading of different UI themes, styles and available languages by the MPA agent, and the determination of the user's profile and preferences which are maintained as part of a user session details by the MMIA agent (Section 6.3.1.5). The user has naturally registered on the platform and been authenticated on the current interaction session. The request is handled by the agents as follows (Fig. 7.1):

1. The HTTPA agent receives a request from the user browser, specifying the requested item and an ID associated with the user session. The HTTPA agent determines if the request is a Multipart HTTP message (for cases of uploading files onto KnowNet), extracts the KnowNet associated variables from the request string (in this case, the *piaskid* session variable), and then forwards the request to the MMIA agent.

2. The MMIA agent processes the request to determine the authentication status of the user. Since the user is already authenticated, the *request processor* module within the MMIA agent passes the request to the associated context handler (in this case the *know.piask* handler). The handler determines the availability of relevant agents to handle the request and forwards it to the primary KBA agent.

3. The KBA agent strips all the KnowNet related variables from the MMIA agent request to determine the operation parameters. The possible operations here include updating content, or deleting information from the knowledge base. On determining that the request is for retrieving information, the request gets processed via the *search engine* module to get the relevant ontology, the subject and possibly the predicate. The *request dispatcher* module of the KBA agent then forwards the request to the specific domain agent (in this case

the local file system agent - KBLA) or to all the relevant and available agents for further processing.

4. The KBLA agent receives the request, unpacks the associated variables, and fetches the requested information (i.e. from the file system ontology and the local data store). The fetched information is then formatted into the platform data format and sent back to the main KBA agent.

5. The KBA agent aggregates all the relevant responses from the domain specific agents and sends these back to the MMIA agent.

6. Bundled with the response from the KBA agent is the meta information regarding the requested information (e.g. the owner, the permissions, and the properties). The MMIA agent extracts this meta information and determines the permissions associated with the returned knowledge. The permissions within KnowNet are defined on the basis of the relationship between the author of the information and the person requesting the information (Section 6.3.1.2). The MMIA agent therefore sends a request to the SNA agent to determine the associated relationship.

7. Since the SNA agent has access to the profiles of the users of the system, it sends back the information to the MMIA agent specifying the type of relationship between the author and the person requesting the knowledge.

8. The MMIA agent then appends the associated interaction parameters (e.g. the different interaction paths from this point (Section 6.3.1.6), the standard menu options, or queued notifications for the user) to the content, and sends the content back to the HTTPA agent.

9. The HTTPA agent appends the information identifying the requesting device (which determines the device specifications and capabilities) and sends these to the MPA agent.

10. The MPA agent strips off KnowNet control messages from the content. Based on the specifications in the control messages, the MPA agent passes the content to the relevant content renderer, applying the specified theme and style, in the available language preferred by the user. On completion of the rendering, the MPA agent sends the formatted content back to the HTTPA agent.

11. The HTTPA agent sends the content back to the requesting user device.

Within this process, the clearly defined and distinct roles of the agents in the platform are illustrated. As highlighted in Section 6.1, the agents exist as autonomous but functionally interdependent members of the KnowNet community of agents. This autonomy gives the architecture a modular and scalable configuration that can evolve and be arranged according to the environmental and contextual constraints.

7.2 Security and confidentiality

In Section 4.5, we discussed the confidentiality requirements associated with different types of knowledge as defined in the OSCA knowledge matrix. Measures have been added to provide the required level of security and confidentiality in the platform: the critical communication channels between agents are encrypted. One of such links is between the MMIA and the SNA agents. The integrity, confidentiality and accuracy of this channel messages is important for the following reasons:

1. Authentication on the system is executed through messages passed from the MMIA agent to the SNA agent which is the only agent that has access to user's credentials and preferences.

2. Authorization on the platform, which is undertaken by the MMIA agent, is done in dialog with the SNA agent which has the relationship types (Section 6.3.1.2) that define the different authorization levels.

3. New user registrations are also handled through this channel by the MMIA agent.

KnowNet utilizes an asymmetric encryption based on the RSA algorithm, with a 1024 bytes key size. The encryption public keys are pre-shared between the agents, and the associated *Cipher* objects are initialized during the agent setup sequence. The encryption occurs as a last stage before the content is added to the *ACLMessage*, and the decryption is the first process after the content has been read from the incoming message stream (see Section 6.7 for a detailed discussion of the communication protocol).

7.3 Robustness and fault-tolerance

The robustness of KnowNet is a crucial factor especially in the context in which the platform is deployed. The environmental factors in Dwesa introduce key constraints that have to be taken

into consideration in the deployment of ICT based services in the area, and highlights the key differences in the solutions developed for a rural marginalized community and those developed for a community anywhere else (e.g. first world, urban community). Two of the factors motivating for high levels of robustness and fault-tolerance are:

1. The infrastructural services in Dwesa are intermittent, particularly the electricity supply. The ICT infrastructure that we have deployed is geared particularly for handling the sudden losses in electricity, and the *brownouts* that occur. The critical network components are powered through Uninterruptible Power Supply (UPSs) which filter out the voltage fluctuations and allow for graceful shutting down of network services in cases of power loss. This fact of intermittence in Dwesa (and other rural marginalized communities) necessitates that measures and mechanisms are integrated at all the levels in the service provisioning stack, to be able to recover from interruptions, and to tolerate faults in the supporting infrastructure.

2. The other constraint as far as handling faults and interruptions in the network is concerned is the remoteness of deployment site and the lack of technical expertise in the region. As far as possible the services should automatically recover and self-heal without the need for a technical expert's intervention.

Fault-tolerance (a.k.a. fail-safe design) is a feature of systems and applications to continue working in the context of different system components failing or not being available. One of the common strategies to provide fault-tolerance in a system is to introduce redundancy in the critical components. An important motivation for the implementation of the knowledge platform as a MAS is the inherent ability of agent architectures to provide a redundant and distributed operation environment. The autonomy and behavioural completeness of agents have two key advantages with regards to robustness and performance. As far as robustness is concerned, the critical platform operations can be isolated and handled by a group of redundant agents, and with regards to performance, those agents that handle operations that become bottlenecks in the system, can be replicated to farm out the task processing among the agents.

The MMIA agents play a fundamentally crucial role in handling procedural tasks on the platform (Section 6.3). A level of robustness is therefore ensured by duplicating the MMIA agent within the different containers that are available on the MAS platform. Greater levels of fault-tolerance are achieved by distributing the platform containers across different hosts.

The main container is another important component in the life of a MAS platform: because the main container hosts the agents that handle critical platform operations (e.g. AMS, DF)

(Section 6.1).

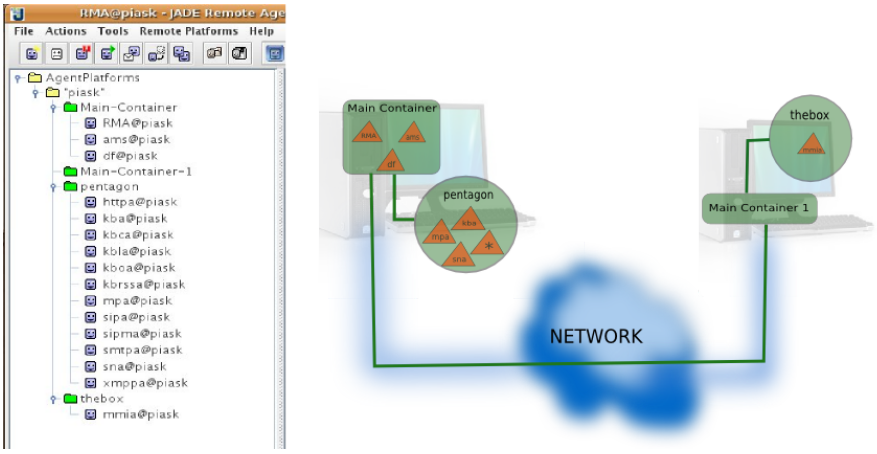


Figure 7.2: KnowNet redundant main container

The ultimate configuration for KnowNet as far as robustness and fault-tolerance is concerned, is one in which the critical Main Container role is duplicated on a different host (Fig. 7.2). Within this configuration, a backup Main Container is initiated on a different host. The role of providing platform specific services (i.e. through the AMS and DF agents) is still handled by a single Main Container. In the case of the Main Container failing, the surviving backup Main Container takes up the position as a leader in the MAS platform, and the peripheral containers reconnect themselves to the new leader (Fig. 7.3). As an illustration of this, the KnowNet platform was setup as per the configuration in Figure 7.2. The Main Container was then intentionally terminated (through the Remote Management Agent (RMA) interface). The KnowNet logs that show the process of platform recovery are in Appendix G.

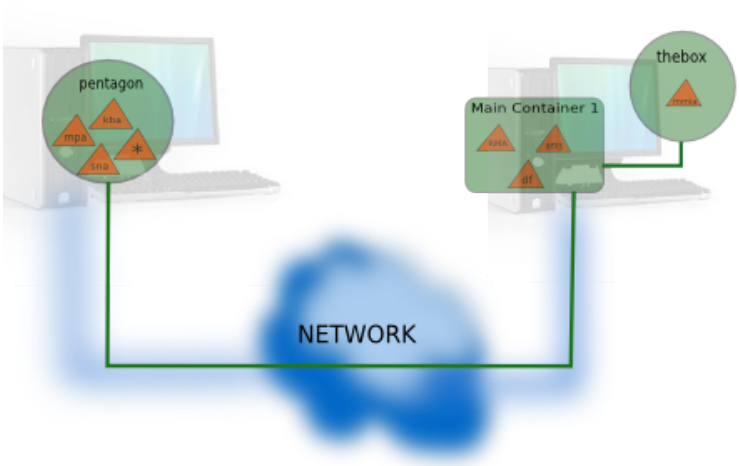


Figure 7.3: KnowNet Fault Recovery

7.4 The platform management agent

The different agents on KnowNet perform the functions specified within the five tiers of the architecture. There are also agents (e.g. the DF and the AMS agents) that are part of the infrastructure of a JADE MAS framework. We have implemented an agent that manages the KnowNet platform to provide an effective operation environment. This agent, PIASK Management Agent (PManA) ensures an end-to-end, secure, efficient and robust service provision. The key responsibilities of the PManA are:

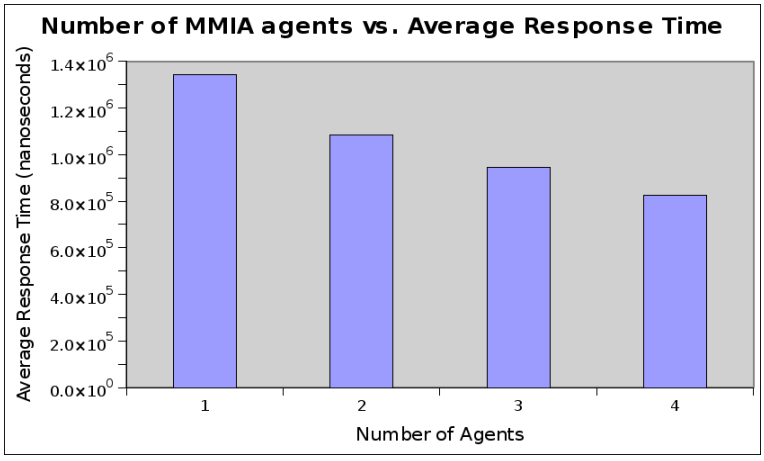


Figure 7.4: PManA adding more agents to KnowNet platform

1. Initializing agents and adding them to the platform. The PManA loads the available agents from the file system and initializes them on the MAS. In other words, when starting the platform, the only agent initialized is the PManA, and this subsequently populates the whole MAS with the platform agents.

2. Creating redundant containers on the primary KnowNet host. The PManA also creates peripheral containers that associate with the Main Container of the platform.

3. Identifying the other hosts that are part of the platform, and distributing execution across the different hosts for load balancing purposes.

4. Ensuring the optimal performance of the platform by adding new agents to the platform under heavy load. The different agents within KnowNet monitor performance of the "next leg" agents and give feedback to the PManA which compares the reported performance with the established thresholds for the different agents. For example, the HTTPA agent monitors the performance of the MMIA agent over an average of requests and sends an update to the PManA. On receiving the updates, the PManA compares it with the established MMIA agent performance thresholds and current platform conditions (e.g. the number of MMIA agents already on the platform, the performance of the agents further down the execution path) and determines whether to add a new agent to the platform or not. The effect of adding MMIA agents to the platform over the average response time is depicted in Figure 7.4 and a juxtapositioning of platform performance with and without the PManA is shown in Figure 7.5. With the PManA active on the platform, obvious performance improvements are observed on the platform (e.g. lower average response times, minimized variance across all the requests, etc).

5. Monitoring the status of the different agents on the platform and resuscitating the agents that have erroneously terminated. The PManA constantly polls the AMS agent to determine the agents that are registered with the platform. If any of the agents terminate unexpectedly on the platform, the PManA relaunches those agents. Figure 7.6 illustrates the activities around this process. Process *A*, *E* and *J* represent the PManA polling the AMS for the list of current agents on the platform. As discussed in Section 6.2, the handling of user requests (process *B* and *C*) goes from the access agents (*httpa6*), through the interaction and knowledge agents and ultimately to the presentation agent (*mpa17*) for rendering. For the sake of this demonstration, after process *E*, the *mpa17* agent is intentionally terminated from the platform, and the PManA launches a new presentation agent (*mpa18*).

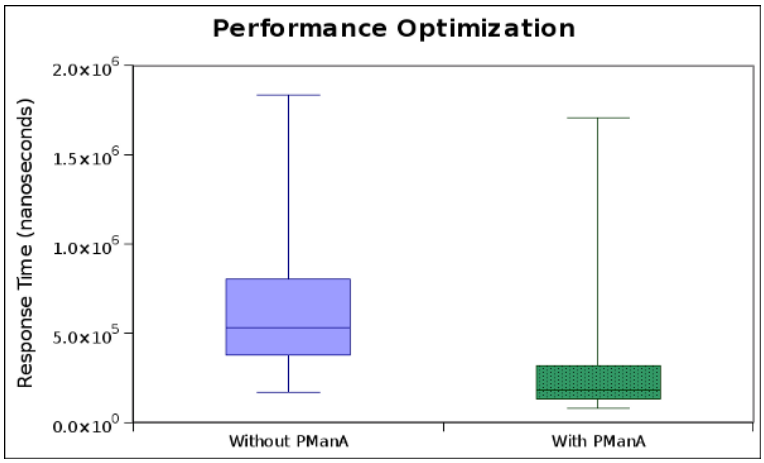


Figure 7.5: PManA performance optimization

Subsequent requests for content rendering are now handled by the new presentation agent (process *G* and *I*). Process *H* represents the performance updating that the access agent is making to the PManA, based on the performance of the interaction and knowledge layer agents.

6. Providing redundancy of the critical components in the platform. As highlighted in the previous section, some of the critical components include the Main Container with the associated DF and AMS agents, and the MMIA agents which provide the crucial logical implementation of the services on the platform.

While the KnowNet agents can exist and be part of a community without the PManA, the role of the PManA within a MAS ensures a more robust platform, which adapts to the environmental conditions (e.g. platform usage load, and faults in the related platform hardware). The PManA is therefore a critical agent in the provisioning of a robust knowledge platform.

7.5 Technical validation of KnowNet

The technical validation of the KnowNet platform is undertaken to determine the adequacy of the platform as far as the technical aspects associated with the operation and the performance of the platform are concerned. The experiments have been undertaken focusing at different levels of KnowNet's processes, to validate implementation decisions made in the platform.

Some of the overall technical objectives in the implementation of the platform and associated with increasing the performance of the platform are:

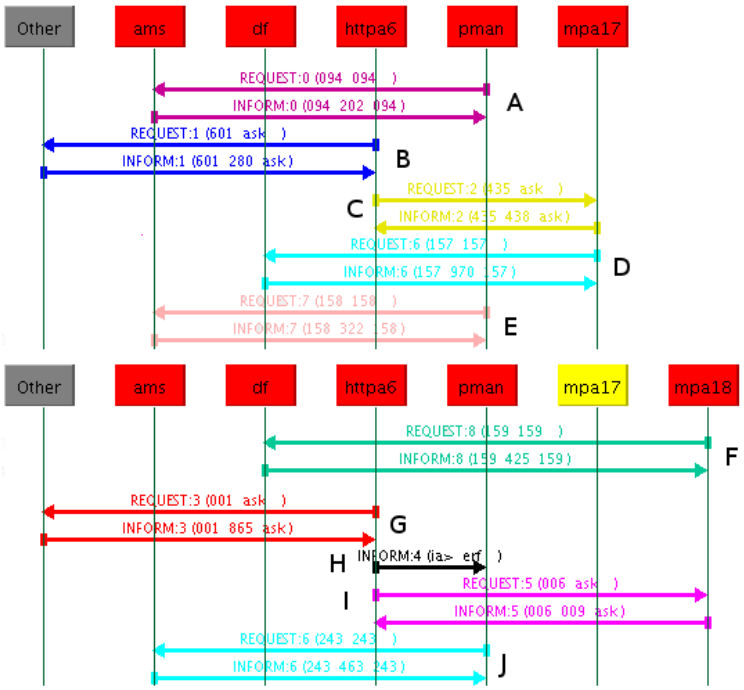


Figure 7.6: PManA relaunching a dead agent

1. Increase availability - the availability of the platform services is a key objective that equates to implementing a robust and resilient platform that at worst degrades gracefully under stress. Some of the measures that have been implemented in the platform towards ensuring robustness have been discussed in Section 7.3, in particular focusing on the architectural adequacy of MAS environments and the role of the PManA agent within the platform that is solely responsible for monitoring the platform and reacting to environmental and system variations to ensure greater service availability and robustness.

2. Minimize latency and maximize throughput - the user's experience in utilizing the platform is one of the key factors taken into consideration in the decisions made during the implementation of the platform. The higher the throughput that can be realized in the platform, the better the end user's experience. Throughput is measured in terms of the number of requests that can be handled in the platform per second. The capability in the platform to serve all user requests with a minimum latency between requests and the responses, contributes to a better user experience on the platform.

3. Minimize resource utilization - the infrastructural constraints in rural marginalized communities necessitate that scarce resources are managed effectively, and resource utilization

minimized to achieve greater overall efficiency in the network. The resources that have to be managed in the implementation of the platform include: hardware resources (e.g. memory, using Java garbage collection, application threads, database connection pooling) and network resources (e.g. bandwidth, socket connection pooling).

7.5.1 Environment parameters

The environment in which the performance testing of KnowNet was undertaken is described in the next three sections.

7.5.1.1 Hardware and networking

The specifications of the KnowNet server and network switch are reported in Table 7.1.

CPU	Intel Core 2 Quad 2.40 GHz 1066FSB
Memory	1024 MB DDR2 800 MHz DIMM
Hard Drive	250 GB 7200 RPM SATA
NIC	Intel 82566DM-2 onboard Gigabit Adapter
Switch	CheetaHub Power 3016A 16-port 100Mbps

Table 7.1: Hardware specifications

For the performance testing, KnowNet was deployed in the simplest configuration: as a single Main Container platform on a single host. The production deployment is more distributed and with more redundancy to ensure a robust, fault tolerant platform, and therefore its efficiency is reduced as a result of handling messaging between different agents, across the network to different hosts.

7.5.1.2 Software

The software utilized in the implementation of KnowNet is reported in Table 7.2:

7.5.1.3 Operational parameters

The environment was setup to meet the minimum requirement of the usage reality in Dwesa, the community site of research. In this community, we have setup five computer labs, four of which are connected to the local WiMAX network. These labs are the initial point of presence for the Internet in the community.

Software	Description
Ubuntu 8.04 (hardy)	The operating system (OS) used in the implementation of the platform and on test environments
Java SE Runtime 1.6	The application runtime environment
KnowNet 1.0	The KnowNet platform
JADE 3.6	The MAS environment
JENA 2.4	JAVA ontology API
JETTY 6.1	JAVA-based embeddable web server and servlet container
Grinder	A load testing and network load simulation framework.
JMeter	An Apache stress testing application
Apache Benchmark (ab)	An Apache web server load testing application

Table 7.2: Software specification

The load on the Dwesa network is currently minimal but it can only be expected to grow in the future, as more labs are setup and more added services deployed. Based on the total number (approximately twenty five) of computers that have been deployed among these schools, the current maximum number of expected simultaneous users on the network services can safely be estimated at less than fifty. Due to the fact that the computing infrastructure is hosted at schools, there is a need to fit the community usage of the resources within the normal schedule of the schools, and therefore this limits the average number of hours of high usage loads on the network to approximately four hours a day: in the afternoons.

These metrics provide an indication of the minimum parameters that have to be taken into consideration in the performance evaluation of the platform. The number of users on the network informs the level of concurrency that must be considered in the evaluations. The daily usage pattern is not immediately considered in undertaking the performance evaluation: however, it is an important indication of the level of uptime required on the platform. Most of the experiments described in the next sections, simulate much higher usage scenarios than expected, at least at present.

7.5.2 Profiling the platform's execution

The processes that take place on the platform are diverse, as indicated by the numerous platform contexts that have been defined as service handlers within the MMIA agent (Section 6.3.1.4). Still, there are regular or more frequent processes on the platform one of which is browsing and accessing the knowledge encapsulated in the knowledge base agents. This process is executed through a minimum of five agents: an access layer agent, an interaction agent, two knowledge

base layer agents and a presentation layer agent. For different types of knowledge, the process can involve up to 6 agents on the platform. For example, a request for knowledge residing on a generic ontology (i.e. accessed through the KBONTA agent) and having access permissions associated with it (i.e. requiring the SNA agent for determining the scope and the level of the permissions).

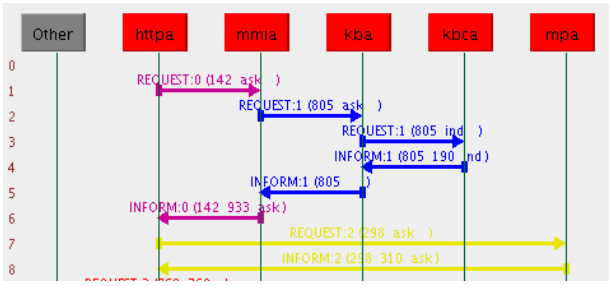


Figure 7.7: RMA messages for KnowNet processes

The basic profiling of the processes involved, in terms of the interaction between the agents and the temporal characteristics of the different legs of execution, is shown in Figure 7.7 and Figure 7.8 respectively. The HTTPA agent receives requests from the user's browser, these requests are passed to the MMIA which forwards them to the KBA. The HTTPA afterwards forwards the request to the presentation agent (i.e. MPA) to have the knowledge rendered accordingly.

The different agents were responsible for recording the time taken by the next leg. For example, the MMIA agent is responsible for timing how long the KBA takes to process the request, from the moment the ACLMessage is send to the KBA to the moment a response is received.

From Figure 7.8, the observation (as expected) is that the slowest leg of the platform process handling is the KBA leg. This leg of execution is typically associated with CPU intensive I/O processes of accessing the underlying knowledge base, querying the ontology index and compiling the knowledge into a standard platform data format (Section 6.7).

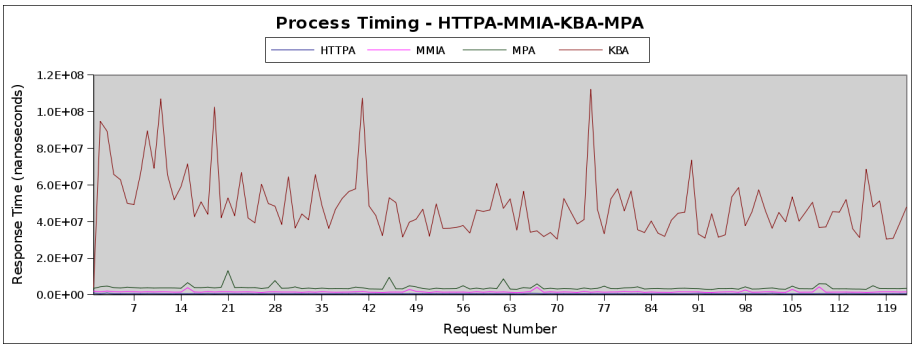


Figure 7.8: Temporal profile of different KnowNet request handling legs
The graph shows the time taken for the individual legs: HTTPA is the time from the user device to the HTTPA agent, MMIA is the time from the HTTPA to the MMIA agent, MPA is the time from the HTTPA agent to the MPA, and KBA is the time from the MMIA agent to the KBA agent.

Figure 7.8 also gives an indication of the relative performance of the different legs on the platform in order to highlight possible bottlenecks in the platform. This observation, of the possible bottleneck associated with the KBA leg of request processing, has fed into the following platform optimization decisions:

1. The performance optimization strategy of the PManA agent (Section 7.4), resulting in the addition of more KBA agents with increasing usage loads on the platform. Figure 7.9 is a box-plot that shows the effect of adding more critical agents (e.g. KBA agents) to the KnowNet platform for handling a simulation of three simultaneous users each with with 500 consecutive requests. There is an increase in the performance of the platform (i.e. decreased average response time) up to three agents added to the platform. Subsequent agents (i.e. fourth and fifth) only increase the variability of the response times, without a significant reduction in the average time per request. The performance increases up to three agents because the load from the three users gets distributed among the agents until each agent is, on average, handling requests from a single user. Adding more agents to the platform (i.e. the forth and the fifth agents) does not change the average response time significantly. This is because the user requests get handled immediately (i.e. without being queued up per agent) and therefore get the best response time, and at the same time more agents being added increase the load on the platform resources, which can lead to the variability that is observed.

2. Optimization of the processes within the KBA agent leg, through database connection pooling and in memory caching of JENA ontology models that are utilized by the KBONTA

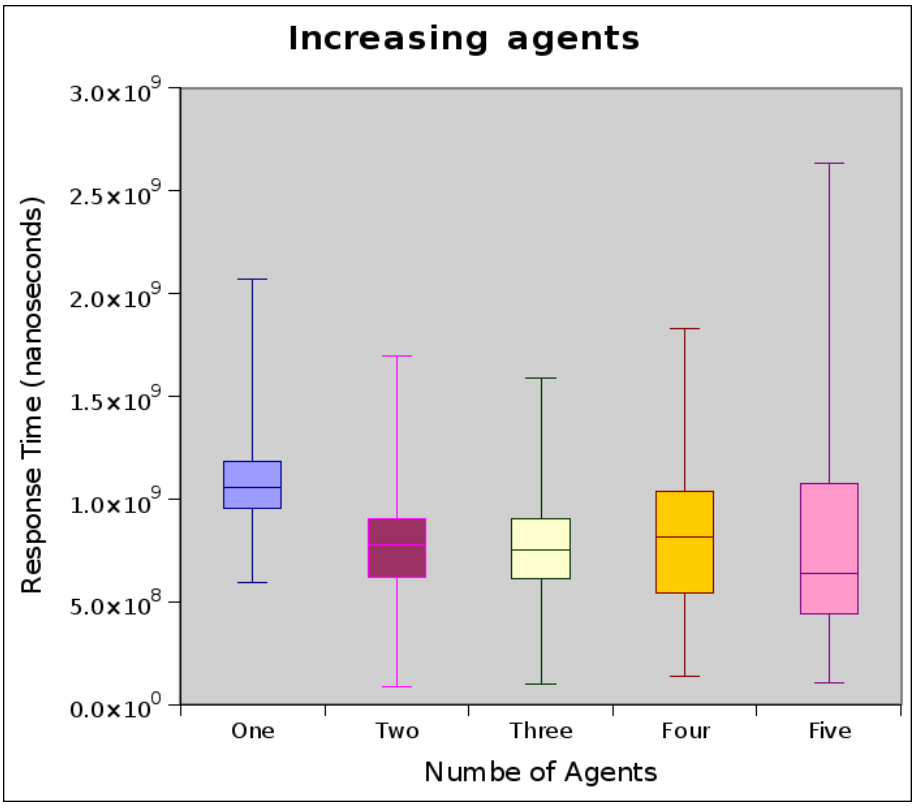


Figure 7.9: Performance optimization box-plot
In the box-plot, the top and the bottom of the box represent the top and lower quartiles (i.e. 75th and 25th percentiles) respectively and the line that cuts across the box is the median. The top and the bottom whiskers represents the maximum and the minimum values respectively.

(Section 6.6.2).

3. The integration of the Lucene search engine within the KBA agent process handling (Section 6.6.5) to decrease the time for ontology querying process.

7.5.3 Performance profiling

In profiling performance of KnowNet, the key observation is how the performance scales with an increasing number of concurrent users on the platform. In the test setup, the requests are for a listing of the content contained in one of the ontologies (i.e. listing of all the countries as contained in the world factbook ontology). The processing of these requests is handled by the HTTPA agent, MMIA agent, KBA agent, KB_C_A agent (a specialized knowledge base agent for the world fact book ontology) and the MPA agent. After the content has been rendered by the MPA agent, the document length is 37998 bytes. For the actual simulation of requests, the

Apache Benchmark (ab) tool was used [203].

For the first test, an increasing number of concurrent users was simulated on the platform (i.e. 5, 10, 20, 40, ...) with each user doing a single request. For each number of concurrent users, the simulation was performed 10 times and the average requests per second recorded. For this test, the success rate for the handling of requests was 100%. Figure 7.10 shows the observed throughput of the platform: an increase in the number of requests handled per second up to 80 concurrent users and then a gradual reduction on handling requests above 160. Zooming in on the graph within the ranges of the estimated usage loads on the platform (Section 7.5.1.3), we can observe an increase in Requests Per Second (RPS), leveling off at approximately 18 RPS for 80 concurrent users. What is important to observe from the graph is the response of KnowNet after it reaches its peak at 80 concurrent users: because this shows how the platform responds when it is overloaded. How fast the platform reaches the peak and the actual values at which it peaks are not of much interest as they depend on the test setup. Therefore, within the usage parameters of the platform, KnowNet scales to the increase in the number of users, and even in higher usage loads the performance degradation is smooth and gradual, which means the platform has predictable responsiveness.

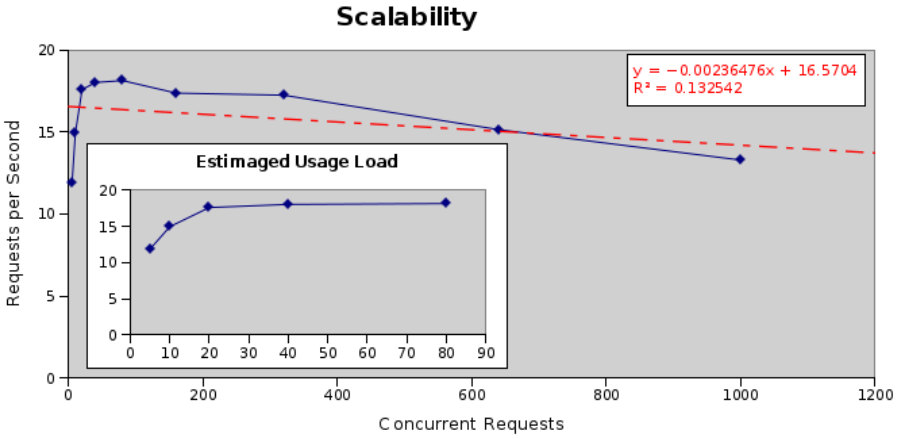


Figure 7.10: KnowNet throughput - Requests per Second

The Request Per Second (RPS) metric shows the number of requests processed by the KnowNet platform per second.

The second test on the platform also simulated an increase in the concurrent number of users on the platform. The observed metric in this case is the Total Request Time (TRT) (i.e. the total time to process all the concurrent requests). This gives a clearer and more accurate (i.e. a high coefficient of correlation of 0.999) picture of the response of the platform, and shows a linear

increase as the number of users increases (Fig. 7.11).

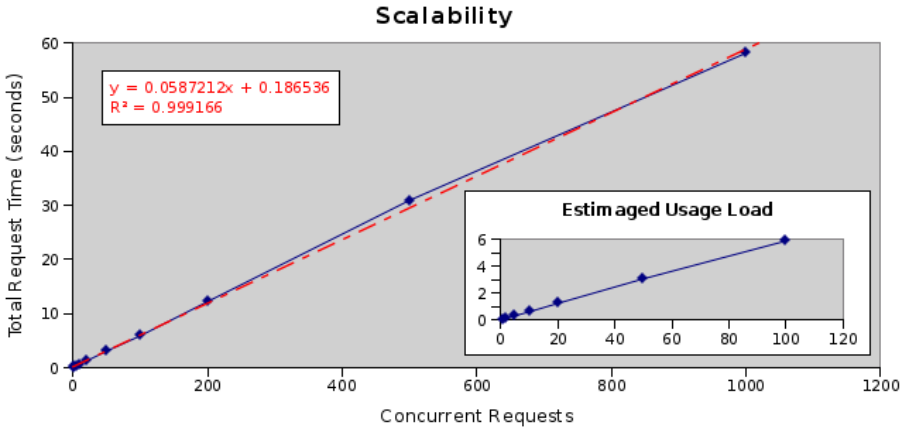


Figure 7.11: KnowNet scalability - Total Request Time

The Total Request Time (TRT) metric represents the time taken to process all the concurrent requests.

The exact values observed in these tests are relatively insignificant as they are dependent on test environment factors (e.g. the rendered document size, the bandwidth available on the network, the hardware involved in the testing). The key observations that validate the platform for performance adequacy are the linear scaling to handle an increase in the number of concurrent users (Fig. 7.11), and a smooth and gradual performance degradation under high usage loads (Fig. 7.10). While a large part of this observed behaviour is influenced by the performance of the embedded Jetty web server, it still highlights and validates the overall performance of the agents within the platform.

7.6 Developing services on KnowNet

The importance of integrating local knowledge in the services developed for rural, marginalized communities has already been discussed in Section *2.2*. KnowNet primarily provides a knowledge platform for these communities in a manner that is flexible and that can be ethnographically contextualized to their environment. The platform, however, also provides an environment for deployment of knowledge based services which can be developed in the ways described in the following sections:

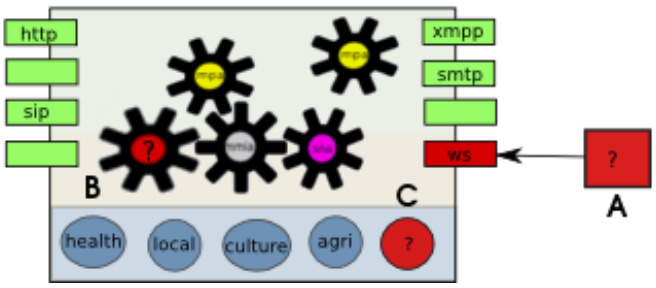


Figure 7.12: Developing services on the KnowNet platform
The three alternative ways of implementing services on the KnowNet platform are represented by the question mark ("?").

7.6.1 A web service client

Section 6.4.6 described how the platform has been complemented with the WSIG add-on that is part of the JADE environment to expose the selected behaviours within the agents as web services ports. In Figure 7.12, component A shows this web services interaction with the platform. The provision of web services is handled as any other access layer interfaces that facilitates the client connection into the platform. The web service requests are handled initially by the PWSA agent and then the platform agents, with the support of the interaction agents.

To develop services using this interface, the developer does not need to be aware of the internal processes within KnowNet: this interface shields the developer from the details of the implementation. Because of the nature of web services, the information that a developer requires is made available through the platform WSDL (see a sample in Appendix E), which specifies the services provided, the format of the messages passed to the platform and the service binding parameters associated with the platform.

The services developed in this manner only access and utilize the knowledge and the business logic that is encapsulated in the platform. They are limited from utilizing the access layer components directly as interfaces to the developed service. For example, the flexibility of utilizing different access technologies (e.g. HTTP, VOIP, XMPP) would not be available for the developed services. This is because the role of the platform in the service deployment architecture is as a service provider and not a consumer.

An example of utilizing this interface is the development of the mPIASK J2ME application. The development of the application involved utilizing the Wireless Toolkit libraries to generate application stubs, from the WSDL, that are utilized to access the functionality available in the platform. Some application development environments, such as NetBeans [204], provide features for processing the WSDL to extract and generate the stubs that can be utilized in service

execution.

7.6.1.1 mPIASK

mPIASK is a tool that leverages the proliferation of cell phones in the rural marginalized communities, as indicated in Section 4.1.2. Cellphone are easily accessible to every person in the Dwesa community and as such provide an immediate interaction device for the users in the community. KnowNet handles interaction from cellphones currently in two ways:

1. The users can use their cellphone web browsers to access the platform. In this case the requests are handled directly by the HTTPA agent (Section 6.4.3).

2. The users are also able to access the platform by initiating a call to a KnowNet phone extension. The call is handled by the Asterisk PBX, which processes the request via an VXML browser. The interaction between KnowNet and the VXML browser is then handled by the HTTPA agent, with the VXML rendering done by the MPA agent (Section 6.2).

mPIASK has been developed as a J2ME application to provide an interface into the platform for the provisioning of the following functionality:

1. Accessing the information regarding a user's friends. This allows the user to view friends' profiles, add new friends, delete friends, and to communicate with the friends (Fig. 7.13 *c*).

2. Searching for specific knowledge on the platform (Fig. 7.13 *f*).

3. Managing the content that the user has authored. This is in terms of specifying the permissions on the content and sharing the content with specific people (see Sections 6.3.1.2 and 8.1.3).

4. Accessing the notifications from the platform. This includes incoming requests for friendships, request from people to access certain protected content, KnowNet messages, reminders, and community notices (Fig. 7.13 *d*).

5. Managing a user's profile. This includes modifying the user's details, changing passwords, and specifying preferences.

6. Registering new users on the knowledge platform (Fig. 7.13 *e*).

Figure 7.13: mPIASK interface

The mPIASK application is developed as a Web Service client into the knowledge platform (Fig. 6.22). mPIASK is therefore implemented through a set of APIs complying with the JSR172 specification, and is deployable on handsets that support the specification. mPIASK is developed around a number of classes *(*i.e. web services stubs generated through the Wireless ToolKit - WTK2.5.2*)* that are generated based on the WSDL that is created by the WSIG agent (Section 6.4.6).

On execution of a request from the mPIASK application, a SOAP message is sent to the WSIG servlet contained in the HTTPA agent. The servlet forwards the message to the WSIG agent which then sends a message to the PWSA agent. This interaction between the WSIG servlet, WSIG agent and the PWSA agent occurs within the Access layer of the KnowNet platform. The subsequent interaction is handled through the MMIA agent and executed as described in Section 6.3. The last leg of a typical execution (i.e. the rendering of the content through the

MPA agent) is left to the requesting web service clients to handle.

7.6.2 A platform agent

The most functionally integrated method of offering services on KnowNet, in a manner that exploits the full functionality provided is by creating service agents that become part of the agent community. This scenario is illustrated in Figure 7.12 (B). In this scenario, the agent can be implemented at the interaction layer of the architecture to provide the business logic for the provisioning of a new service on the platform.

The developed agent becomes part of the MAS and implements the communication protocols and ontologies defined within the KnowNet MAS. It must be positioned at the right layer, and associate with the relevant agents in the platform. Such an agent will generally associate with the MMIA agent which is responsible for forking out the incoming service requests to the relevant agents. Since the MMIA agent aggregates all the functionality within the platform, a full service agent can be implemented that only needs to associate with the MMIA agent.

As an example of utilizing this method to provide extra services within the platform, we have developed an agent that provides a service to access RSS feeds within the platform (Fig. 7.14). This agent is developed as an knowledge layer agent due to the functional association with the knowledge base layer, as a means of accessing an alternative source of knowledge (e.g. RSS feeds). The key processes that are involved in implementing a new agent on the platform are: associating with the relevant agents on the platform, handling requests in the MAS, and dispatching responses to the requesting agents.

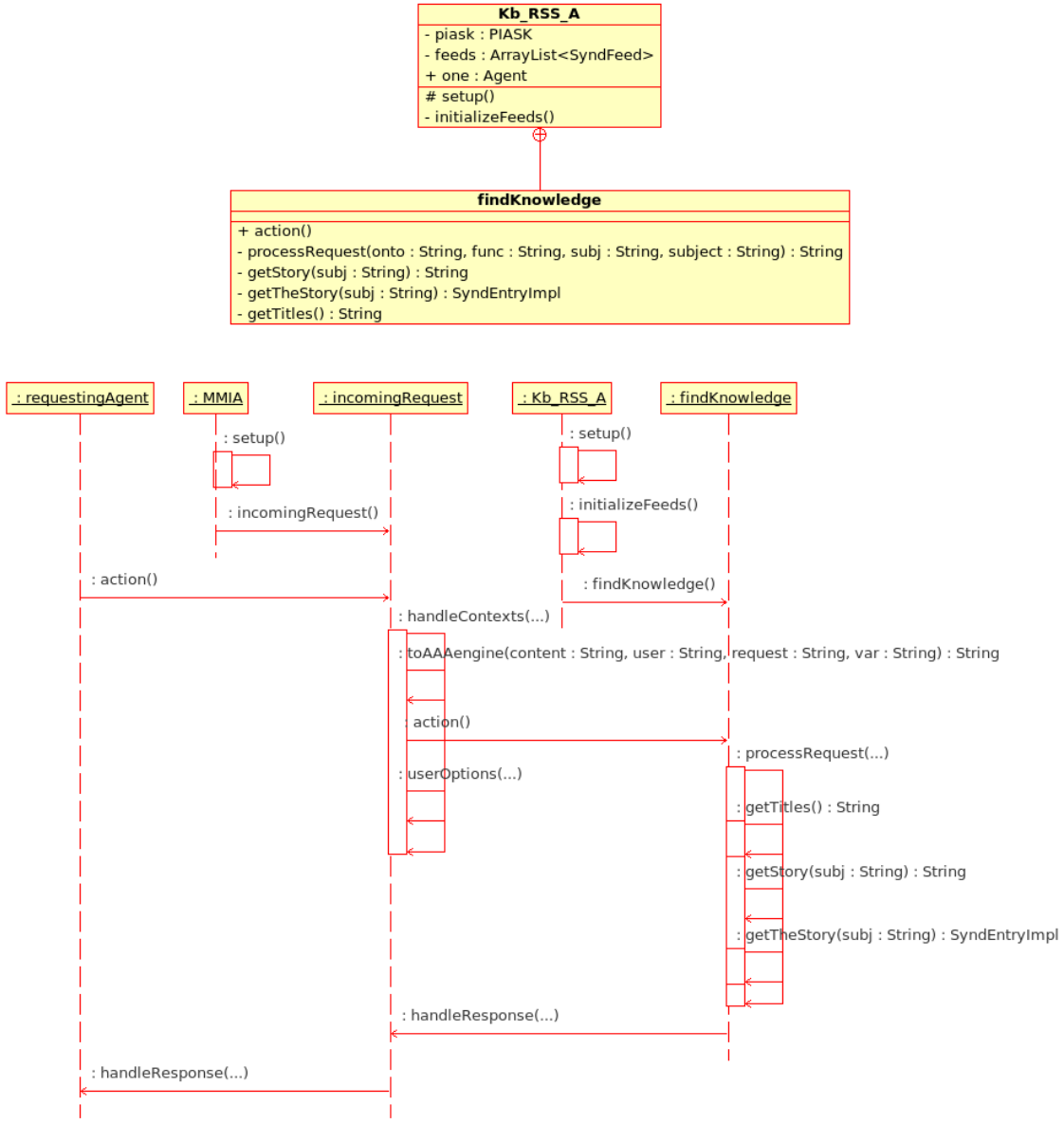


Figure 7.14: New platform agent - RSS Agent

This shows the class diagram and the sequence diagram of the new agent.

7.6.2.1 Association

A new agent on KnowNet joins a specific container on the platform (Section 6.1). On joining the container, the new agent associates with the Main Container that hosts the AMS and DF agents within the platform. The primary action by the new agent joining the platform is to advertise the

services provided, with the DF agent. This ensures that the other agents on the platform are aware of the new services and able to route requests to the new agent. Within the platform, services are advertised with reference to the layer (on the PIASK architecture) in which the specific agent resides. For knowledge layer agents, the associated agent service description object would have the preset name of "PIASK-Knowledge" and for interaction agents the service name would be "PIASK-Interaction". So the RSS agent implemented registers with the DF and describes the services it provides with the "PIASK-Knowledge" name (within the *setup* procedure in Fig. 7.14).

The added agent also queries the DF for the list of agents that implement specific functionality required. The RSS agent is the last agent in a request handling sequence, and therefore it does not need to be aware of any other agents besides the ones requesting services from it. In most cases, the requests are from MMIA agents. The Agent ID (AID) of the specific agent (in case of a multiple interaction agents on the platform) is extracted from the request ACLMessage.

7.6.2.2 Handling requests

The process of handling requests on KnowNet relies on the platform communication protocols. In order to participate in the community of agents, the new agent has to implement the vocabulary and the communication ontology defined within KnowNet or be able to process the platform communication tags (Section 6.7). In handling the requests, the agent extracts the platform control messages from the ACLMessage payload, parses them to determine the specified variables, and then calls the corresponding processes to handle the request.

In Figure 7.14, the RSS agent receives requests from the MMIA agent within the overridden *action()* method of the *CyclicBehaviour* superclass. The message is then passed to *processRequest()* which parses the message to determine variables from the MMIA agent. The internal methods: *getTitles()*, *getStory()* and *getTheStory()* are then called to process the request as necessary.

7.6.2.3 Dispatching responses

The data communicated between the agents adheres to a specific format to facilitate the rendering of the content by the MPA agent (Section 6.2). The agents added on KnowNet are therefore responsible for dispatching the responses in the correct platform format. This is one of the key features that must be implemented by all the agents that join the platform as add-on services.

7.6.3 A platform ontology

The third manner in which services can be created is through extending the platform knowledge base layer by adding new ontologies. The new ontologies encapsulate the data that is then made available within the platform. These ontologies are processed primarily by the KBONTA agent (Section 6.6) which is responsible for accessing the platform's ontologies. For every ontology added to the platform, the KBONTA agent requires the URI of the root class from which the main relevant concepts in the ontology are sub-typed (i.e. associated by the *rdf:type* relationship). The KBONTA agent provides the logic for handling the data level ontologies, thus providing a mechanisms for adding simple content-based services within the platform.

The number of ontologies defined within the platform provide an example of how this can be achieved and integrated into the overall operation of the platform.

Another way in which services can be added onto the platform is by combining the three methods discussed above and developing an agent that provides a web service interface, and processes data stored in its ontology.

7.7 Conclusion

This chapter has highlighted the functional adequacy of the platform for the implementation of knowledge based eServices. In addition to the provision of a complete knowledge service, the platform enables service level extensibility through the following three mechanisms: implementing an added service agent, interacting with the platform through the web services interface, and extending the underlying platform ontologies. Besides the basic function of service provisioning, features have been developed in the platform towards security, robustness and reliability. The technical profiling of KnowNet has illustrated the scalability of the platform for service provision in the context of increasing usage loads.

Chapter 8

Social Context Adequacy of the Platform

> ”Computing is not about computers any more. It is about living” (Nicholas Negroponte)

The previous chapters have provided an overview of the PIASK architecture (Chapter 5), a proof-of-concept realization of the architecture (Chapter 6), and a functional and technical validation of the platform (Chapter 7). This chapter discusses the adequacy of the architecture and the platform to meet the objectives of flexibility for implementation in fundamentally different social contexts, and fitting into the knowledge system dynamics of the local community.

8.1 Culture sensitivity pre-validation

The development of ICT solutions embeds the specific cultural orientation of the developers and the development environment within the solution developed. This embedding can occur at the underlying content level, the UI level through embedding themes, styles and the language, or at the usage metaphor level. While this resultant cultural dynamic is largely beneficial (due to greater cultural interaction and exchange), it can also be detrimental to less prominent, marginalized communities. Through the embedding of a specific culture, the technology can lead to the assimilation of those marginalized culture's expressions and values [5]. So, PIASK needs to be evaluated against the key objective of its design, which is culture and context sensitivity (besides the validation in terms of the technological and practical viability for provisioning of knowledge based services). The developed solution must integrate a level of flexibility that allows for a culture-level customization and localization of the solutions implemented.

The architecture needs to be evaluated on the basis of how well it makes explicit the culture specific factors in a manner that allows for its realization in different contexts. The initial pre-

validation of PIASK is therefore undertaken from a philosophical perspective using Herman Dooyeweerd's Theory of Modal aspects (TMA) [43].

8.1.1 Philosophical validation of PIASK

In order to undertake a cultural validation of the PIASK architecture, the cultural influences in the tools and the methods used must be acknowledged or the tools must be culturally transcendental (i.e. not culturally biased). Dooyeweerd's philosophy is considered a critical and transcendental philosophy in part due to its derivation from the law and meaning side of reality (versus the entity side) and also due to its concern with aspects that pertain to general human existence [205]. TMA is proposed as a "philosophically sound basis for diversity, coherence, and interdiscplinarity" [206]. In TMA, Dooyeweerd identifies 15 spheres of reality (i.e. the modal aspects) in which we operate within our experiential horizon [43]. The modal aspects encapsulate the different factors that are at play in the totality of human activity. Even though Dooyeweerd himself did not claim the 15 aspects to be exhaustive, they seem to encompass adequately the different dimensions in everyday experience.

The 15 modal aspects are: quantitative, spatial, kinematic, physical, biotic, sensitive, analytical, formative/cultural, lingual, social, economic, aesthetic, juridical, ethical and pistic/credal. An example of the interplay of some of these different aspects in the activity of developing a website is as follows:

1. Economic aspect: the costs associated with developing the web site, the utilization of the limited resources (e.g. available space on a web browser screen).
2. Aesthetic aspect: the overall look and structural beauty of the pages, the graphics used.
3. Spatial aspect: the layout of the different page elements, the distances between the graphics on the page.
4. Ethical aspect: the type of content to be posted on the website.
5. Lingual aspect: the elements on the web page for communicating with the users (e.g. icons, links).
6. Social aspect: the interactions between the developers of the web page, and the users.

The modal aspects are not just categorizations of different human activities. They also provide meaning for different activities by stipulating the laws of operation within the specific modal

aspect (e.g. the logical laws within the analytical aspect, the gravitational laws within the physical aspect, the moral laws within the ethical aspect, the spiritual laws within the credal aspect, language grammar within the lingual aspect). A key principle in Dooyeweerd's philosophy is the *Shalom hypothesis* or the *simultaneous realization of norms* principle. The *Shalom principle* highlights the need to operate within the laws of the different modal aspects and to take into consideration the balancing of activities across all the different aspects, in order to "maintain sustainability and deep, rounded, rich well-being" [206, 207]. This principle is the basis for Aspectual Analysis (AA), the tool that has been used to validate the effectiveness of PIASK as a culture-sensitive architectural pattern [5]. AA allows for the analysis of an activity, or a system in a manner that highlights the resultant effects across all the 15 aspects.

In analyzing the PIASK architecture, we have made use of nine aspects (which are: analytical, formative, lingual, social, economic, aesthetic, juridical, ethical and credal) that are identified by Aay and Langevelde as directly contributing to the essence of different cultures (i.e. the key defining and differentiating aspects for specific cultures) [208]. The different aspects were mapped into the layers in which they are addressed, specifying a key manner in which the specific aspect applies to that layer (Tab. 8.1).

PIASK Layer	Modal Aspect	
Presentation	Aesthetic	The harmony in the interface design
	Economic	Effective use of available screen space
	Lingual	Use of symbols for UI
Interaction	Social	Interaction with the Socially Intelligent Agents
	Aesthetic	Intuitive interaction modalities
Access	Economic	Limited resources utilization (bandwidth, device capabilities)
Social Networking	Social	Communities and groups among the users
	Formative	Cultural power dynamics in the society
	Juridical	User's roles and responsibilities
	Ethical	Relational dynamics between the users
Knowledge Base	Lingual	Knowledge representation formalisms
	Analytical	Logical reasoning, distinction of ontology elements
	Credal	Ontological commitments

Table 8.1: Basic Aspectual Analysis of PIASK

The modal aspects mapped per PIASK layer are not exhaustive (i.e. more aspects can still be mapped to the different layers), however, the following observations can be made:

1. The primary interaction of the users with a PIASK based system is through the Presentation

Layer (i.e. the web browser interface, the IM client interface, VOIP VXML browser). Some of the aspects which are operational in this layer and which are culturally sensitive include: the Aesthetic aspect, in which different cultures will have different notions of harmony and beauty; and the Lingual aspect, wherein the elements of the UI are influenced by the underlying symbolism in the culture.

2. The social networking layer allows for a mapping of the existing social dynamics in the community to the users' interaction on the PIASK architecture. The aspects that come into play in everyday interactions (e.g. the social, juridical, and ethical) are operational and supported in this layer.

3. For each of the nine aspects, there is a mapping to a specific layer within PIASK that provides the flexibility to handle features associated with the aspect. For example, at the presentation layer, and with regards to the aesthetic aspect, there is the flexibility to develop new themes and styles for the UI components (Section 6.2).

4. The six other aspects (i.e. biotic, physical, sensitive, spatial, quantitative and kinematic) are not directly mapped to any of the layers of the architecture, however, they come into play in the overall utilization of PIASK based systems. For example, the biotic aspect which has as its kernel meaning life functions that are common to all animals, would come into play in the consideration of usage patterns associated with the architecture. Therefore, healthy (e.g. that do not result in Repetitive Stress Injury (RSI)) interaction mechanism would have to be considered in the implementation of PIASK systems.

From the aspectual analysis of PIASK, it can be noted that the architecture is sensitive to all the nine modal aspects that are closely coupled and relatively unique to different cultures. This means that modifications and customizations can be undertaken to align a realization of PIASK along all the culturally relevant aspects (e.g. a knowledge formalism that closely embodies the ontological views (i.e. analytical modal aspect) within a community can be implemented at the knowledge base layer, and UI components can be developed at the presentation layer that reflect the symbolism (i.e. lingual aspect) prevalent within a culture). This analysis, therefore, provides the initial indication that the PIASK architecture can be utilized for culture-sensitive applications.

8.1.2 Folksonomies and IK

Folksonomies and ontologies represent two positions on the spectrum of KR. On one end is the structuring of knowledge in organic, bottom up folksonomies in which the structure of the

underlying knowledge emerges as *the users* associate content with different semantic tags. On the other end is the formalized, top-down, structured ontologies in which the knowledge gets populated into the ontologies that have been design by *the experts*. These two approaches to knowledge management both have their advantages and disadvantages (Section 3.7 and Section 3.5), which are applicable in the context of the knowledge platform.

The implementation of support for folksonomies within the platform is presented in Section 5.1.6. The users of the platform have the ability to define tags that are associated with the different IK that is added and available on the platform (Fig. 8.4). The weighting of these tags results in an emergent classification of the knowledge in a manner that represents the users' understanding of the structure of the knowledge. This emergent structure as a result of tagging the knowledge items, can be utilized to inform the revision of the associated platform ontologies . The tags associated with content can also add a weighting on the index searches that are performed on the platform. The utilization of folksonomies within the platform therefore allows for KR that is contextualized, due to being informed by the users' ontological views.

8.1.3 Confidentiality requirements for different knowledge

One of the culture sensitive aspects that have been taken into account in the development of KnowNet is the confidentiality considerations associated with the different types of knowledge in Dwesa (and generally in rural marginalized communities). In Section 4.5, we discussed the OSCA knowledge matrix, which highlights the different types of knowledge and the associated levels of confidentiality in the community. The knowledge platform provides permissions control mechanisms to enable the specifying of the different confidentiality requirements. These requirements are handled as follows:

1. For the *specialist knowledge* (type D in Fig. 4.2), there is highest requirement for confidentiality and this kind of knowledge is only shared between a few specific individuals. This is achieved in KnowNet through an option to allow the owner of the knowledge to share it with a specific person already on their list of friends (i.e. related by a *dwesa:knows* or sub-class thereof) and to limit access from the rest of the users of the platform (Fig. 8.1 (a) and (b)).

2. The *specific group knowledge* (type C in Fig. 4.2) is shared between a small group of individuals and the group membership is typically based on a relational association between the members. Within the platform, this is enabled through permissions that are associated with a relationship type (Fig. 8.1 (c)). For example, one can set the read flag only for the

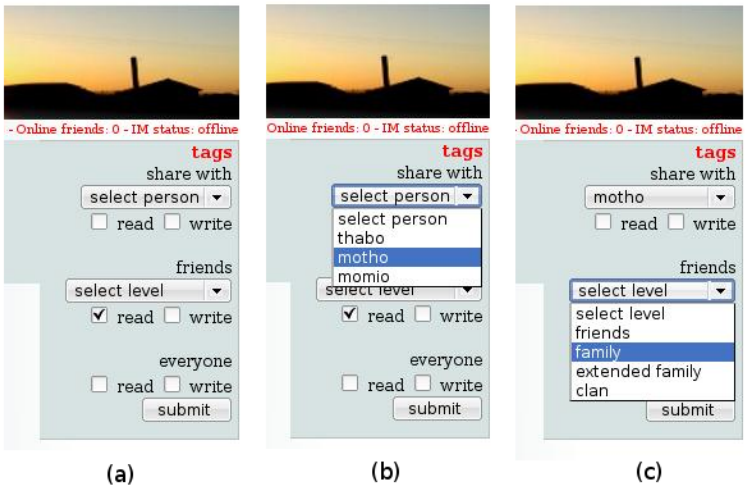


Figure 8.1: KnowNet confidentiality and permissions

people within the same family (i.e. associated by the *dwesa:isFamily* predicate) or people within the same clan group.

3. A key consideration for the *shared cultural knowledge* is the establishment of the ownership of the knowledge (Section 4.5). On the KnowNet platform this is achieved through the association of every unit of knowledge authored with a specific user, or a specific cultural group. The actual mechanisms of enforcing the protection of the knowledge (from exploitation and abuse) are within the legislative domain in terms of copyright laws and policies.

4. The *common knowledge* is accessible and available to every user of the platform and therefore such knowledge can be authored with the *read* and *write* permissions set *On* for the group *everyone* (Fig. 8.1).

KnowNet implements access and permission control mechanisms that offer flexibility to cater for the different relational groupings within the community of users. The implementation of these mechanisms is possible due to the functional separation of content (i.e. knowledge base tier) from the domain logic (i.e. interaction tier) and the integration of the relational dynamics (through the social networking tier) as articulated through the PIASK architecture. The platform is therefore validated as adequate for the purpose of encapsulating different types of knowledge, from the point of view of implementing the necessary confidentiality and ownership requirements as per the OSCA knowledge matrix.

8.2 PIASK and SECI

Nonaka's SECI framework for knowledge systems provides an overview of the structures that exist and the processes that occur within a community in the context of knowledge creation, utilization and exchange (Section 2.6.2). This section discusses the developed architecture for knowledge based applications and implemented knowledge platform based on the PIASK architecture in terms of their alignment with the SECI framework.

8.2.1 Knowledge processes and types of knowledge

The four processes in the SECI model are iterative, continuous processes in which individuals engage to create knowledge. These are processes that are observable in any knowledge-based environment. Figure 8.2 shows how PIASK maps into the SECI model, and how the platform provides the handling of some of the community knowledge processes.

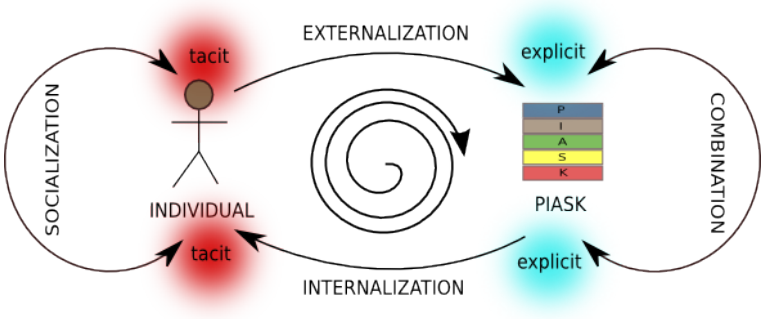


Figure 8.2: PIASK within SECI

1. The process of socialization occurs between individuals in a close context and where there is an exchange of tacit knowledge. This process does not interface with the KnowNet platform as it happens completely around tacit knowledge.

2. Externalization involves the explicit conversion of tacit knowledge into explicit knowledge, and this done by the individuals articulating the specific knowledge and formalizing it. KnowNet facilitates this process by allowing for the capturing of new knowledge into the platform. This is the primary way in which knowledge is created on the platform.

3. Once the knowledge has been explicitly codified on the knowledge platform, individuals have access to that knowledge and through the browsing of different knowledge, they are

able to create new knowledge through the process of combination. Within the Knowledge Base layer of the PIASK architecture, the platform agents are able to engage in the combination process and create new knowledge by executing inference rules on the explicit knowledge in the ontologies. This handling of the combination process within KnowNet validates the motivation for utilizing ontologies (versus other data stores and knowledge repositories) in the knowledge base layer of the platform. The combination process represents the secondary way in which knowledge is created on the platform.

4. The last process in which the platform gets involved, is the internalization process. In this process, the individuals convert explicit knowledge into implicit, tacit knowledge. This process is supported in KnowNet through the intrinsic functionality of allowing the knowledge on the platform to be accessed and exchanged with the users of the systems, taking into consideration the access permissions that have been set for the different types of knowledge.

The knowledge platform therefore provides an architecture that effectively supports the knowledge creation and exchange processes, associated with explicit knowledge, as enunciated in Nonaka's knowledge framework.

8.2.2 The *Ba* in PIASK

The process of knowledge creation necessarily occurs in a relational context. Within Nonaka's knowledge framework, the concept of *'Ba'* refers to a space where different contexts are shared to create meaning and new knowledge. A context provides an objective position from which to create meaning and to make interpretations.

> "without context to specify time, place, and relationship with others, knowledge becomes just information" [87]

The individuals' contexts therefore provide the time, the place and the relationship within which the SECI processes can occur. *Ba* can be developed between individuals, and in a context of a firm, it can grow to be between departments, and up to being between firms and organizations. One of the key factors that Nonaka *et al.* highlight as crucial in energizing and activating *Ba* is the provision of "love, care, trust and commitment" [87]. These factors are intrinsically ingrained in relationships between individuals and within groups.

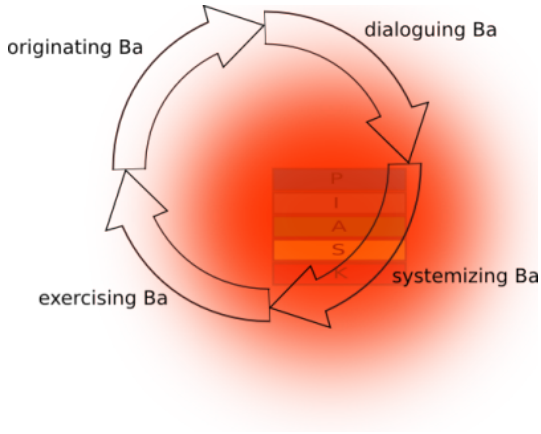


Figure 8.3: Ba in PIASK

KnowNet in essence is a provisioning of a virtual *Dialoguing Ba*, *Systemizing Ba* and *Excercising Ba* (Section 2.6.2) for the community within which the platform is deployed (Fig. 8.3). The context interactions that can happen on the platform are predominantly within the *Systemizing Ba* wherein there is a collective interaction between the different users of the platform to create explicit knowledge. The social networking layer in the PIASK architecture under-girds the interaction of different individuals' contexts in undertaking SECI processes on the platform. The PIASK architecture through the social networking layer, therefore facilitates the realization of the "love, care, trust and commitment" sentiment that is necessary for organic and effective knowledge creation.

The overall mapping of the PIASK architecture into the SECI framework provides an essential validation of the Social Networking and the Knowledge Base layers in terms of how they play distinct roles in providing the relational context for the *dialoguing ba* and *systemizing ba*, and in facilitating the execution of the *combination* process respectively.

8.3 PIASK and the IK life cycle

While the PIASK architecture is designed as generic and applicable in different contexts, the realization of the platform has been with a direct focus on being situated within the context of rural, marginalized areas as typified by Dwesa. Some of the key differences with such contexts are associated with the social dynamics, the socio-technological constraints, and the infrastructural limitations. The notion of IK is predominantly (within literature) applied to these kind of rural contexts, although in reality every community and society has knowledge that is indigenous within that community. The processes associated with IK are therefore universal and apply

equally in different contexts. The life cycle discussed in Section 2.5, encapsulates the key processes that are associated with IK. Although the life cycle is discussed from the understanding of IK being a special, different type of knowledge (i.e. not universally available) and from an external perspective (i.e. from the perspective of someone studying the IK of a specific community), it still highlights the crucial factors in the utilization of IK.

These processes are handled within the knowledge platform as follows:

1. Recognition and identification - the first stage in the knowledge life cycle is the recognition and the identification of the knowledge sources that are available in a community. This process in a sense occurs within the *externalization* process in Nonaka's SECI framework (Section 2.6.2). Externalization results in the knowledge being available in a codified form. This process is therefore facilitated in KnowNet by the ability to add new content on the platform in an explicit format (e.g. a recording of a folktale, or authoring a story).

2. Validation - Once the knowledge is available in an explicit form, the next process involves validating that knowledge along the dimensions of significance, relevance, and reliability. This process is intrinsically a community process, in that the significance, relevance and reliability can only be ascertained in the context of people accessing the knowledge and commenting on it. KnowNet provides a feature for the users of the platform to validate the different content that is available on the platform. The users are able to give a vote on the content, in terms of its reliability, relevance and accuracy (Fig. 8.4). The accumulated voting for the content is then calculated and made available to inform the usage of the content. The validation weights from the users are also useful in ranking the search results on the platform. The net effect from this validation mechanism is that the content that is most reliable, relevant and accurate will hopefully get the highest weighting on the platform and increased availability, and the content that the users do not find reliable, relevant and accurate gets the low weightings.

3. Documentation - The process of documenting the IK primarily contributes to the externalization of the knowledge and therefore this is handled through the mechanism to add new knowledge on the platform.

4. Storage - The knowledge base layer is primarily responsible for the storage of the knowledge that is available on the platform.

5. Transfer and Dissemination - Intrinsic in the platform is the facilitation of the exchange of knowledge between different people and making the knowledge available. KnowNet

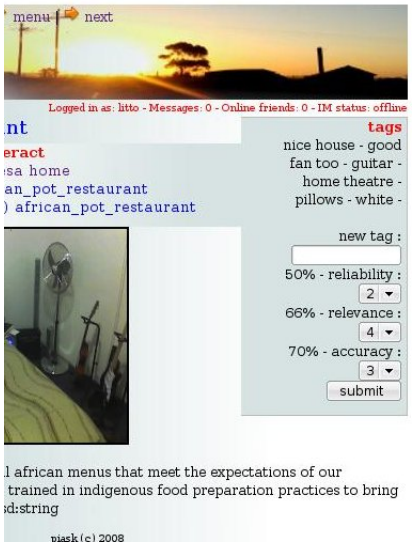


Figure 8.4: PIASK folksonomies for IK validation

provides features for the knowledge to be accessed via any of the channels (implemented via the Access layer agents) that are available on the platform, thus increasing the accessibility of the knowledge. For example, a user can call into the platform, browse the specific content and request that the associated file be emailed to them.

The support for the different processes in the IK life-cycle aligns the platform for effective integration into the communities as far as representing and encapsulating the local knowledge (i.e. IK) within that community.

8.4 Usability assessment of PIASK

The usability of a system is generally considered a measure of the quality of the interaction of the users with the system [209, 210]. Despite the lack of a clear consensus on the definition of usability, there are agreed attributes that pertain to the quality of usability of a system. Traditionally usability has always been purely understood and undertaken within the context of the UI and interaction components of the developed systems. This notion of usability has been enriched, by Folmer and Bosch, by proposing that the quality of usability does not only pertain to the developed and implemented applications, but rather that usability is a quality that is ingrained deeply in systems' architectures [209]. Their proposition is supported by the observation that it is easier to implement good usability features (enumerated through usability patterns) in some architectural frameworks than in others (e.g. the *progress indication* usability pattern would be easy to

implement in architectures where the progression of task execution is easily fed back from the business logic components to the UI components).

Some of the commonly accepted metrics of usability are: learnability, efficiency, memorability, reliability in use, user satisfaction, understandability, operability, attractiveness, low error rate and flexibility [210, 211, 212]. These metrics provide a qualitative and quantitative measure of usability of systems. Some of the associated tools and methods for determining these metrics include: questionnaires, observations, cognitive walk-through, and checklists. These tools are predominantly used for the evaluation of the usability of developed systems and applications on the basis of the interface design and the interaction with the user. Folmer and Bosch have developed a framework for the evaluation of software architectures as far as usability is concerned. This framework, called the Scenario based Architecture Level UsabiliTy Analysis (SALUTA) allows for the assessment of software architectures by highlighting the different task scenarios that are associated with utilizing the software, and then doing a weighting of four usability attributes for the task scenarios highlighted [213]. The four usability attributes in the framework are satisfaction, learnability, efficiency and reliability.

This assessment method has been used in the evaluation of the usability flexibility and sensitivity of the PIASK architecture. The first part of the evaluation is focused on the PIASK architecture (Section 8.4.1). This is because the usability of the implemented knowledge platform is dependent and predicated on the usability sensitivity of the architecture, and also because the usability of the platform is specific and contextualized (i.e. the UI components are already customized with specific themes, linguistic localization). In the second part (Section **??**), we undertaken an assessment based on the usage scenarios on the platform.

8.4.1 Usability profile of PIASK

In undertaking the usability profiling of the PIASK architecture, the SALUTA assessment framework was utilized to establish a usability factor weighting of the five layers of PIASK in terms of the usability aspects handled by each layer. This process involved the following steps:

1. For each usability pattern in the framework (Fig. 8.5), the PIASK layer which supports the pattern is determined.

2. Based on the associated usability property [1], the usability attributes are then mapped according to the SALUTA framework.

[1] Folmer *et al* make a distinction between usability *property* and usability *attribute* [213]. This distinction can be seen in Figure 8.5.

3. For each layer, the total number of usability attributes per pattern are tallied to discover the layers on which usability is most dependent and in which it is most supported.

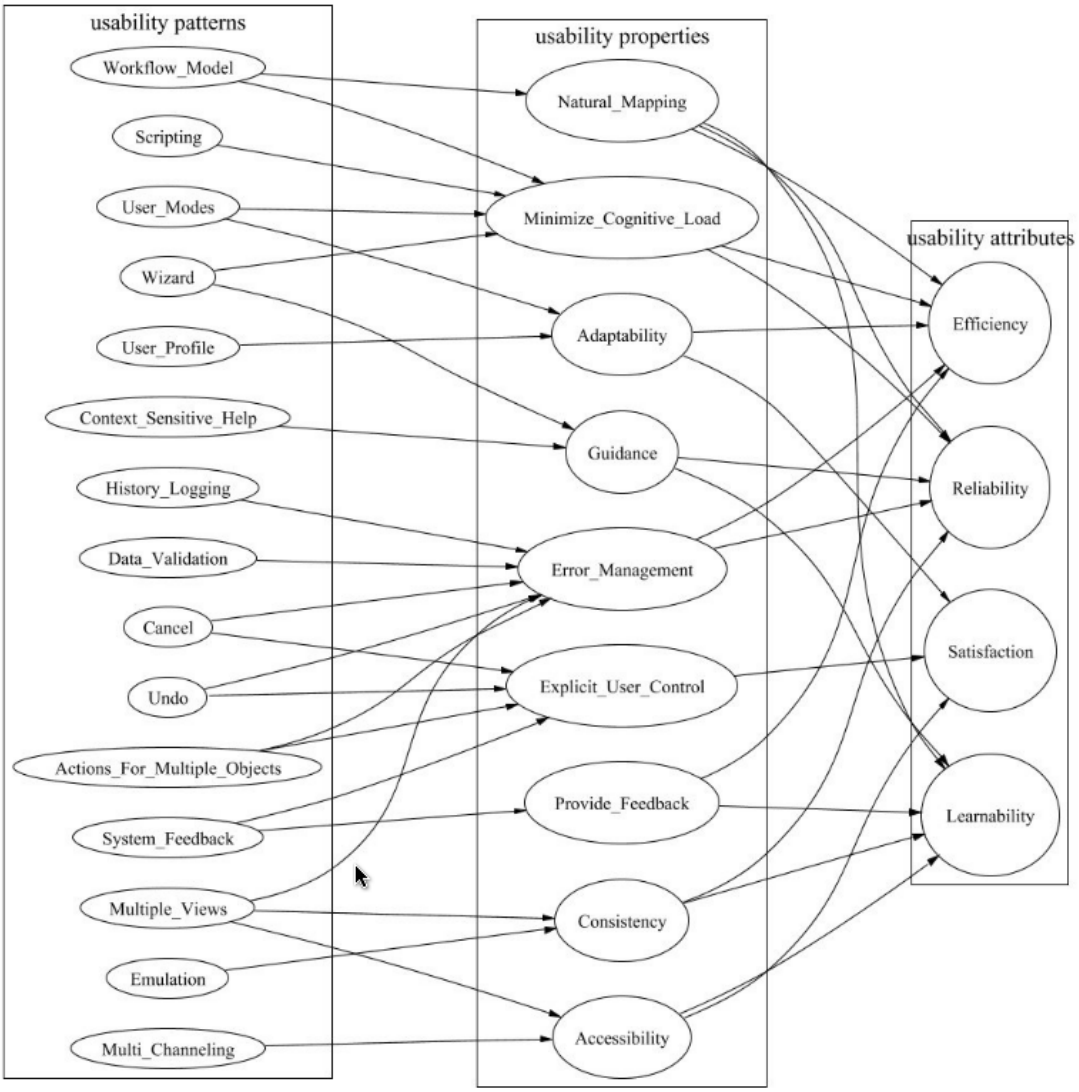


Figure 8.5: Folmer *et al*. Usability Framework[213]

The following example illustrates the steps for the *history logging* pattern (specified in Fig. 8.5):

1. The essence of this pattern is to enable the logging of the user activity on the system. In the case of the PIASK architecture, the functionality of usage accounting (i.e. logging

user activity) is handled by the interaction layer components (see Section 6.3.1.2 on the discussion of the AAA functionality of the MMIA agent).

2. This pattern maps onto the *efficiency* and r*eliability* usability attributes through the *error management* property. Therefore, within the interaction layer of PIASK, the two attributes are highlighted.

3. Finally, the usability attribute count for the interaction layer is incremented by two.

PIASK Layer	Usability Pattern	Satisfaction	Learnability	Efficiency	Reliability	Layer total
Presentation	Multiple Views	●	●●	●	●●	8
	Emulation		●		●	
Interaction	History Logging			●	●	27
	Cancel	●		●	●	
	Undo	●		●	●	
	System Feedback	●	●	●		
	Context Sensitive Help		●		●	
	Wizard		●	●	●●	
	Workflow		●	●●	●●	
	Actions for multipel objects	●		●	●	
	Scripting			●	●	
Access	Multi-channeling	●	●			2
Social Networking	User Modes	○		○○	○	6
	User Profile	○		○		
Knowledge Base	Data Validation			●	●	2
Usability Attribute Totals		8	8	14	15	

Table 8.2: PIASK usability profile

The results of this process over all the usability patterns are depicted in Table 8.2. The results indicate the level to which each of the five layers of the PIASK architecture contribute to the usability of the systems implemented based on this architecture. It is clear that usability is provided for in PIASK predominantly through the Interaction, Presentation and Social Networking layers of the platform (in that order). This is due to the fact that usability is primarily an aspect of the quality of the interaction of the user with the software system, and in the case of the PIASK architecture, the interaction layer components primarily provide the event handling for the interaction with the users, and the presentation agents provide the interface through which the users interact with the applications. The social networking components allow for the consideration of user-specific usability preferences.

This basic analysis of the usability of the PIASK architecture shows the adequacy and the flexibility of the architecture to integrate different usability patterns in a manner that would increase the satisfaction, learnability, efficiency and reliability of the developed applications. We hereafter detail the support implemented, for five of the usability patterns, within the platform to illustrate the mapping of the usability patterns to the PIASK layers.

8.4.1.1 Consistency - Multiple Views pattern

The multiple views pattern contributes towards the usability property of consistency of the developed systems. The essence of the multiple views pattern is to enable the presentation of same data objects through a number of different UI elements. This is achievable through the fact that the PIASK architecture abstracts the UI components into a separate presentation layer. In the implementation of KnowNet, the *mpa* agents demonstrates the implementation of this pattern and the flexibility of the architecture toward the integration of consistency features into the developed applications. In a generic sense therefore, the consistency of the presentation is handled through the presentation components and the consistency of the interaction (i.e. consistent menus, interaction paths) is handled through the interaction components.

A detailed discussion of the presentation and interaction components is undertaken in Sections 6.2 and 6.3 respectively. The multiple views that have been realized (through the different renderers) in the implementation of the knowledge platform include: HTML based presentation, VXML and simple text based content presentation.

8.4.1.2 Error Management - Data Validation pattern

The data validation pattern leads directly to the error management usability property. Data validation is a quality feature of a system that allows for the information, to be stored onto the system, to be validated for integrity and accuracy before being committed into the system. The separation of the data objects in the architecture into the knowledge base components that interact with the interaction components, allows for the implementation of the data validation pattern in the PIASK architecture. The knowledge base layer predominantly handles the validation of the knowledge that is added to the knowledge repositories. For example, in KnowNet, the KBA agent validates the new content updates to determine the accuracy of the content and the alignment with the defined ontology. The underlying ontology API, Jena, also undertakes the validation of the content data types before committing to the knowledge store. The back-end database (in this case MySQL) also undertakes internal data validation (e.g. consistent data types, referential integrity) before the new rows are added to the database tables.

Within the PIASK architecture, therefore, the key aspect that allows for the achievement and the flexibility to implement the data validation pattern, is the separation of the architecture data objects from the rest of the components in the architecture.

8.4.1.3 Feedback - System Feedback pattern

The interaction components on the PIASK architecture provide the business logic in the system. These components are at the core of the interaction between the users and the system, because they are attached to the UI elements, the different computational elements, and the underlying data objects in the system. One of the usability patterns which leads to the general feedback property of systems is the system feedback pattern. This pattern can be implemented entirely within the interaction components level on the PIASK architecture. Within the current realization of KnowNet, a number of context handlers have been defined that are associated with providing specific logical functions on the platform. The progress in the execution of these context handlers can be communicated to the users through the current channel of interaction of the user, or through the channels that the users are available on (by utilizing the multi-modality features provided in the platform - Section 6.3.1.5). For example, when the user has requested a file to be emailed to their inbox, the interaction component (i.e. *mmia* agent) determines the task execution status from the access component (*smtpa agent*), and wraps the status in a feedback response message sent to the user.

8.4.1.4 Adaptability - User Profile pattern

The ability of software applications to adapt to different end-user preferences is a usability property that can be achieved through the user profile pattern. The essence of the user profile patterns is to enable user level and user specific customization of UI components based on the users' profile. Within the PIASK architecture, the Social Networking components are directly responsible for this aspect of being aware of the users (e.g. aware of their profiles, preferences) and in interaction with the presentation tier components and interaction tier components, to provide the users with their preferred specific UI themes, styles and interaction modalities.

In the KnowNet platform, the presentation agent (*mpa*) renders the content based on the information regarding the users' preferences as communicated from the social networking agents (*sna*), through the access level agents (*httpa*) (Section 6.2).

8.4.1.5 Accessibility - Multi-Channeling pattern

The multi-channeling usability pattern contributes towards greater accessibility of the developed application and systems. Within the PIASK architecture, the access layer is developed particularly to provide the access specific functionality that leads to multiple interaction channels on the developed systems. Some of the channels that have been realized in the platform though

the access layer components, include the standard HTTP channel, the VXML channel and the XMPP channel. The details of these access level components are discussed in Section 6.4. This separation of the transport specific operations into the access layer components has enabled the realization of multiple extensible channels into the architecture and the platform in a manner that is independent from the business logic and the presentation considerations in the architecture.

8.5 Conclusion

In this chapter we have illustrated the different dimensions that validate the adequacy of the PIASK architecture. At one level, the extent to which the architecture accommodates various aspects of a culture, towards being an ethnocentric and culture-sensitive architecture, is validated through the basic aspectual analysis of the architecture based on Dooyeweerd's theory of modal aspects. At another level, we have shown the sufficiency of the architecture in terms of: its alignment with the standard knowledge systems theory as articulated in Nonaka's SECI knowledge dynamics and the knowledge context of *Ba*; the feature of the platform to support the IK life cycle processes; and the positioning of the platform to encapsulate the different types of knowledge as enunciated in the OSCA knowledge matrix.

PIASK provides a modular architecture by making explicit the social networking and the access components of systems into separate layers. The effectiveness of this separation of the access and social networking components is validated through the SALUTA based usability profiling of the architecture, in which the flexibility afforded by this separation is shown to be adequate for the implementation of various usability patterns.

Chapter 9

Conclusion and Future Work

> "The possession of knowledge does not kill the sense of wonder and mystery. There is always more mystery" (Anias Nin)

In Chapter 1 we set the context and direction for the research. This chapter concludes the dissertation by summarizing the work done. We present the primary contributions that we have made, discuss the issues that arise out of this research, and conclude by exploring future research directions.

9.1 Summary of the thesis

The main research context in which this work took place is that of ICT4D. The proliferation of ICT4D interventions is a testimony to the extent to which ICT is thought to be an enabler for development. The role of ICT in community development has to be understood in the light of the different paradigms used for formalizing the problem of poverty and under-development. As described in Section 2.1.2, some of the formal paradigms from which ICT4D interventions can be understood are: technological, economic, structural and cultural. Within each of these paradigms, there are specific targets and solutions that can be developed, and within each paradigm, there are pitfalls that must be avoided, such as, the technological determinism that plagues many ICT4D projects undertaken within the technological paradigm. We have adopted and positioned this research within a knowledge-centric approach to ICT4D as a primary paradigm for undertaking the intervention. We argue that the full potential of ICT for development is in enabling the communities to appropriate the technology to their context and their situation (Section 2.3), in a manner that allows for a heightened sense of ownership and par-

ticipation in the intervention, increase in the quality of life, and a synergy out of integrating the technology into their life systems and their local knowledge into the global knowledge systems.

An important requirement for ICT4D is to be ethnocentric and context-sensitive, as articulated in part, within the discipline of ethnocomputing. There are a number of different levels at which context-sensitivity can be implemented in ICT solutions: at the UI level, through linguistic and cultural customization of the interface components, and through implementing interaction modalities aligned with the reality of the users; and at the level of the underlying knowledge, by integrating mechanisms for encapsulating the local knowledge (IK).

In this thesis, we have positioned context-sensitivity as the core focus, right from the conceptualization of application architectures for services deployment. We formalized an architectural framework, called PIASK, which is informed by the requirements derived from a profile of rural, marginalized communities. In profiling the deployment context, we explored the nature of the knowledge and the associated system dynamics within these communities, and developed the OSCA knowledge matrix to highlight the key requirements for the different knowledge categories. The PIASK architectural framework is also informed by the current trends and developments around service oriented architectures, web services architectural developments, semantic computing technologies, MAS application development, and Web 3.0 developments (specifically the notion of the Giant Global Graph (GGG) - Section 3.8.1, with the emphasis on the interconnected nature of knowledge).

We developed KnowNet, a proof-of-concept ontology-based knowledge platform to validate the PIASK architecture. The platform is realized as a MAS based on the JADE development environment. While undertaking the development of the platform, we discussed the key considerations and mechanism towards providing flexibility at the application development level: the use of an intrinsically decoupled development platform; implementing mechanism of encapsulating different domain ontologies through knowledge base layer agents; and modeling through the social networking agents the aspects of utilizing the platform that are associated with the social dynamics within the community.

We then evaluated PIASK and the KnowNet platform from the point of view of meeting the specified objectives of context-sensitivity and of being culturally situated, and for adequacy as a platform for service deployment, particularly in rural, marginalized communities.

9.2 Addressing the research objectives

We set five research objectives in the introduction chapter that encapsulated the solution to the problem areas identified. This is how the objectives have been met.

1. *Objective 1 - Investigate and formalize an architectural framework for software that is sensitive to the socio-technical context of deployment.*

   We have proposed the PIASK architectural framework which provides a template for the implementation of context-sensitive, knowledge based applications. The PIASK architecture is informed by the investigation of the current web service architectures, the current trends associated with Web 3.0 and social applications, and also by the requirements extracted from the profile of marginalized communities. Context-sensitivity is provided for in PIASK through: defining a social networking tier that models the social system dynamics (i.e. relationships, trust factors, group memberships) into the architecture; separating and making explicit the access components for handling requests from a multiplicity of application layer protocols (on the OSI model); and allowing the flexibility of a modular, functionally loosely coupled architecture.

2. *Objective 2 - Explore the knowledge system dynamics and the social context in marginalized rural communities*.

   The understanding of the knowledge system dynamics in rural communities informed (through the community profile) the conceptualization of the PIASK architecture. The key knowledge dynamics that have been observed as important to consider in such architectures have been presented through the the OSCA knowledge matrix, which is a presentation of different types of knowledge mapped along the dimensions of ownership, social advantage, confidentiality, and accessibility.

3. *Objective 3 - Develop a proof-of-concept knowledge platform based on the architecture.*

   We have developed KnowNet, a PIASK based MAS platform. KnowNet provides support for environment heterogeneity through presentation agents that handle content transcoding and rendering for different devices, interaction agents that provide multi-modal interaction with the users on the platform, access agents that provide service access for a multiplicity of devices based on the underlying communication protocol, and social networking agents

that position the platform within the social context of deployment. In the realization of KnowNet, knowledge base layer agents have been implemented that handle the processing of local ontologies.

4. *Objective 4 - Undertake a pre-validation of the architecture and the developed platform.*

   The theoretical validation of the architecture has been undertaken, through Dooyeweerd's TMA, the SALUTA based usability analysis and through Nonaka's SECI processes (and within the context of *Ba*), to establish the level of social context sensitivity. The KnowNet platform has also been validated for functional and technical adequacy, as a scalable, reliable (as far as predictable performance in overloaded conditions) and flexible service deployment platform.

5. *Objective 5 - Develop tools and services*

   KnowNet has not been developed only as a stand-alone knowledge platform, but also as a service provision platform. We have demonstrated three mechanisms of deploying services on the platform: through the web service interface, utilizing the services exposed through the platform WSDL; developing an agent that joins the KnowNet MAS; and ultimately by extending the underlying ontologies encapsulated in the platform. We have also developed mPIASK, a tool that allows access to the platform from the end users' mobile phones, in a way harnessing the proliferation of cellphones in marginal rural realities.

9.3 Contributions

This research has the following as its specific contributions within the domains of ICT4D, application architectural frameworks, ethnocomputing and (HCI).

9.3.1 Knowledge-centric paradigm for ICT4D interventions

We have explored an ICT4D intervention based on encapsulating and integrating local knowledge within the developed systems

ICT4D interventions can adopt one or a hybrid of the following paradigms (discussed in Section 2.1.2): economic, structural, technological or cultural. In this research we have explored

ICT4D primarily from knowledge-centric paradigm, in which the essence of the developed solution (in this case the PIASK architectural framework and the KnowNet knowledge platform) is to encapsulate the local knowledge (a.k.a indigenous knowledge), and to facilitate the processing and exchange of the knowledge. We observe that community activities are structured around different types of knowledge, and therefore maintain that the full effectiveness of ICT4D interventions can be realized as knowledge networking solutions (on which further specific eServices can then defined and implemented) are developed, as opposed to restricted, narrowly defined service interventions (Section 2.2.3).

This perspective has been validated in our experiences in Dwesa (Section 4.1), where the community has managed to realize (for themselves and by themselves) the different activities that they could undertake as a result of the availability of the knowledge network. The primary conceptualization of the multi-functional, distributed communication platform presented in Section 4.4 is developed from this knowledge-centric paradigm to provide a distributed platform that is inherently built around the IK in the community.

9.3.2 Web 3.0 in third world context

We have shown the applicability of high-end technological developments for the 'low-end' marginalized third world rurality context.

This research has been undertaken within a domain (i.e. ICT4D) that is relatively new within the systems and applications development domain. This context presents unique challenges, from a systems development perspective, associated with both the effects of culture on technology and of technology on culture. We have articulated these challenges (e.g. need for context-sensitivity, multi-modality, system-wide flexibility towards situating the solution within the cultural environment) in a specific manner and highlighted the associated technological solutions. In the realization of the solutions, we utilized technologies (e.g. ontology and folksonomy knowledge tier based on current KR formalisms (e.g. RDF, OWL, RDFS), and Multi-Agent System platform) that are not typically explored for the rural marginalized community contexts, and shown the adequacy and particular applicability of these technologies for these contexts. These technologies are at the core of the latest development on the Internet around Web 3.0 [214]. A typical pessimistic perspective (as termed by Schon *et al*) would be to deploy high-end latest computing solutions within high-end contexts, and low-end, traditional ICT solutions in similarly low-end domains (in a manner that has become typical of ICT4D interventions based on the *technology dumping* attitude (Section 1.2)) [215].

This research work has also contributed ontologies that can be utilized for knowledge based applications developed in the context of rural marginalized communities.

9.3.3 PIASK architectural pattern

We have proposed a software architecture for context-sensitive, ethnocentric knowledge-based applications that exist in heterogeneous environments.

The PIASK architectural pattern (Section 5.3) builds on the principles that have been developed over the years in Object Oriented Design, Component Based Design and Service Orient Architectures to provide a framework that guides the development of context-sensitive, ethnocentric systems that operate in a heterogeneous contexts as far as the content, the user interaction modalities and end-user devices are concerned. Besides the decoupling of domain data, business logic and interface components (which is standard in many architectural patterns - e.g. the MVC pattern) the PIASK architecture abstracts the systems access specific functionality into a behaviourally complete and separate access tier. The architecture also provides an explicit mapping of the ethnocentric and social dynamics within a community through a separate social networking layer.

9.3.4 OSCA knowledge matrix

We have defined a tool for profiling knowledge in terms of ownership, social advantage, confidentiality and accessibility requirements.

The OSCA knowledge matrix (Section 4.5) allows for a mapping of knowledge resources within a domain of interest, into the distinct dimensions of ownership, social advantage, confidentiality and accessibility. This gives a clear indication of the requirements on the subsequently developed applications and systems for that specific domain, in terms of implementing measures to meet the four dimensions.

9.3.5 Mechanisms towards ethnocentricity and context-sensitivity

We have explored the integration of ethnocentricity into software applications through an explicit, inherent social networking layer.

Within the discipline of ethnocomputing, the need for developed applications and systems to be context-sensitive and ethnocentric is greatly emphasized. We have proposed a solution based on integrating a social networking layer into ICT systems and applications as a means to positioning the solutions within the social and cultural dynamics of a community. The second part of the proposed solution is to encapsulate the local IK within the developed knowledge-based application in a manner that is aligned with traditional IK life-cycles, and in a manner that allows for a synergy out of having both the local and the external knowledge available on the system (i.e. attaining a synergy associated with the process of combination of local and external knowledge, as outlined in Nonaka's SECI framework (Section 2.6.2)).

9.3.6 KnowNet platform

We have implemented a knowledge-based services platform, that is also functioning as a technology integration platform

The KnowNet platform (Section 6) allows for the deployment of multi-modal, context sensitive, heterogeneous knowledge portals. The platform also allows for deployment of knowledge services based on three different deployment strategies, which provide different levels of integration into the rest of the platform functionality, access to the behaviours defined within the platform agents, and access through the external interfaces defined in the platform. We have highlighted the adequacy of MAS environments for service deployment in the typically resource constrained, erratic rural and marginalized environments as far as providing a platform that is robust, fault-tolerant and resilient. We have also developed a platform that enables the processing of IK in a manner that is aligned with the IK life-cycle and that adequately maps into the SECI knowledge framework.

The KnowNet platform has also evolved as a service technology integration platform. The MAS environment has enabled the realization of a system that is multi-protocol and that easily integrates different technologies in an organic manner. The current deployment of KnowNet is integrating the following diverse tools, technologies and protocols: Asterisk PBX, JENA, voiceXML, Web Services, Jetty web server, XMPP, HTTP, SIP, Lucene, MySQL, FastAGI, JavaMAIL and many others, in a manner that is organic and that demonstrates the flexibility of MAS's.

9.3.7 TMA for ICT4D interventions

We have shown the applicability of TMA and aspectual analysis for the ICT4D domain.

The validation of the culture sensitivity and adequacy of PIASK necessitated an exploration of methods and techniques for evaluating how well a system addresses the different aspects of a culture. We have made use of Herman Dooyeweerd's philosophy as a framework for the validation of the culture sensitivity of ICT4D interventions. Dooyeweerd's Theory of Modal Aspects (TMA) philosophy has been explored extensively in different disciplines, and aspectual analysis has been utilized for validating the completeness and the *harmony* (derived from the *shalom principle*, or the *simultaneous realization of norms* principle) of different systems. We have contributed to showing the usefulness of the TMA and aspectual analysis as applied to ICT4D interventions (Section 8.1.1). Within ICT4D, TMA allows for the evaluation of the different interventions in terms of how complete and balanced they are in addressing the different aspects (i.e. as enunciated through the 15 modal aspects) of the community life.

9.3.8 Phone gestures

We have developed a phone gestures interaction technique.

We have developed an interaction technique and a mechanism for the VXML interface, that has been motivated by the mouse gestures (i.e. the use of the mouse pointer to draw patterns that are interpreted as commands for the application) to enable the users to navigate within a VXML interaction sessions. Phone gestures (Section 6.3.1.6) provide a visual, easy to remember mapping to the standard menu options associated with navigating a VXML document.

9.4 Discussion

This research makes its main contributions within the ICT4D domain, which is relatively in its infancy as far as theoretical models and implementation frameworks are concerned. In this section we discuss further perspectives and views around the contributions made in this research. On one hand, this allows for a critical consideration of the associated issues, and on the other hand allows for the appreciation of the complexities of this domain.

Unique features of the knowledge-centric ICT4D approach.

A key conceptual position and contribution made in this research is the knowledge-centric paradigm for ICT4D interventions. This paradigm has not been extensively formalized or articulated in the thesis, but has been illustrated through the investigation and subsequent implementation of the KnowNet knowledge platform. In one way, it can be argued that all ICT solutions process the knowledge that is available in the context of deployment, and therefore that all the ICT4D paradigms are to an extent inherently knowledge-centric. The main difference is that within the knowledge-centric paradigm, the solutions are developed around the available IK to provide an open (in compliance with the context accessibility constraints), unrestricted processing of the knowledge. The realization of specific services (which would then typically fit into the current ICT4D paradigms: eCommerce service within the economic paradigm, or eGovernment service within the structural paradigm, etc) is then undertaken on top of the knowledge based solution. This point is argued in greater detail in a paper published in 2008 [4].

KnowNet as a MAS.

We have realized the KnowNet platform as a JADE Multi-Agent System (MAS), and this is retrospectively motivated in Chapter 7. In and of itself, the implementation and realization of PIASK as a MAS is not an innovative contribution. The evaluation of KnowNet has however highlighted a key advantage and applicability of MAS environments for the ICT4D context of rurality as far as system robustness, flexibility and fault-tolerance are concerned. The context of rurality (in which ICT4D interventions are sometimes undertaken) is characterized by erratic network conditions, unstable infrastructure and is a generally resource constrained environment. The key features of MASs that have been observed as particularly suitable to these environments are: inherent agent mobility to position agents within different hosts as a response to environmental conditions; autonomy and redundancy of the agent which allows for greater levels of fault-tolerance; and the inherently distributed nature of the environment. Within KnowNet we have introduced a management agent that ensures greater levels of robustness and efficiency of the platform.

The complexities within the ICT4D domain.

The nature of the problems and challenges within ICT4D domain is that they are multi-

disciplinary and multi-faceted. The total solutions to these problems are in themselves multi-disciplinary. In this research we have articulated an overall conceptual approach to ICT4D (i.e. the knowledge-centric paradigm) that we have adopted, and within that, narrowed down the focus to the technical and technology based solutions to the identified problems (ineffective ICT4D interventions associated with lack of ethnocentricity and context-sensitivity).

We have undertaken the evaluation of the effectiveness of the proposed architectural framework and the implemented knowledge platform from a restricted perspective (i.e. the technical and functional evaluation of KnowNet and the social context sensitivity evaluation of PIASK). The evaluation of the full effectiveness of the proposed solutions (validated by the improved quality of life in the communities, increased codification of IK, greater integration of marginalized communities into the global knowledge networks) cannot be undertaken within the time period of this research, and therefore this becomes a continuous future aspect of this research.

9.5 Future work

The formalization of the PIASK architecture and the realization of the KnowNet platform have provided the first building blocks towards context-sensitive, ethnocentric and effective ICT4D interventions. Much work and research still remains to be undertaken within the ICT4D domain, and specifically within the ICT4D intervention we are undertaking in Dwesa. We have identified the following areas which can be a focus for further research.

9.5.1 Elaborate usability evaluation

In this thesis we have undertaken a theoretical evaluation of the context sensitivity and ethnocentricity of both the PIASK architecture and the KnowNet platform. While the full validation of the architecture is beyond the scope of this research, it is still a matter worthy of further investigation. The pre-evaluation undertaken in this research points to the potential of PIASK and KnowNet to meet the objectives of being sensitive to the deployment context.

9.5.2 TMA based ICT4D integration framework

ICT4D is an inherently multi-faceted and multi-disciplinary domain. The majoring of the ICT4D interventions are typically undertaken within a very limited and constrained perspective (e.g. technological, economic). At best, the different players (i.e. stakeholders) within the ICT4D

landscape are aware of the multi-disciplinary nature of the domain. What is still lacking is an integration framework for the different activities and interventions that are undertaken within the ICT4D, that allows for synergy and cohesion out of the different perspectives, different methodologies, and different paradigms. In this research, we have briefly explored Dooyeweerd's TMA as an integrated philosophical framework for evaluation of ICT4D interventions. The possibility that a framework based on TMA towards the integration of ICT4D activities might contribute significantly to the domain in general, merits further investigation.

9.5.3 Standardization of the inter-agent ontology

Within the current deployment of the KnowNet platform, the communication between the agents is undertaken based on platform specific tags. While this has the immediate benefits of efficient performance and ease of implementation, it does not exploit the advantages of open accessibility associated with utilizing platform vocabularies and ontologies to define the semantics of the inter-agent communication. It is worth exploring the problem of standardizing an ontology for communication between heterogeneous agents in an open MAS community within an ICT4D domain. Such an ontology would enable a far more efficient and extensible deployment of agents within ICT4D interventions.

9.5.4 Development of supporting tools

There is currently a lack of supporting tools and applications that would enable the full and effective deployment of the KnowNet platform within the contexts of marginalized rural communities. Examples of such tools include Automatic Speech Recognition engines and Text-To-Speech voices specifically for the languages in focus in this research (e.g. Xhosa).

9.6 Concluding remarks

While the work undertaken in this research provides an initial solution for the problem of situating developed ICT solutions within the ethnographic, cultural and social context of the focus community, much work still remains within the ICT4D domain to enable the solutions developed and the interventions undertaken to achieve full activation of communities into knowledge societies and into participating in the global knowledge society. The nature of the problem is intrinsically multi-disciplinary, and the solutions that are developed necessarily will have to be informed by perspectives from multiple disciplines. The technical solution we have presented

will become fully effective within a larger framework that addresses much more than the technical issues.

References

[1] S. Milgram. The small world problem. *Psychology Today*, 2(1):60–67, May 1967. USA.

[2] B. Skiba, A. Tamas, and K. Robinson. Web 2.0: Hype or Reality...and how will it play out? *White Paper*, February 2006. London.

[3] M. Tedre, E. Sutinen, E. Kähkönen, and P. Kommers. Ethnocomputing: ICT in cultural and social context. *Communications of the ACM*, 49(1):126–130, January 2006.

[4] M. Thinyane, L. Dalvit, A. Terzoli, and P. Clayton. The internet in rural communities: unrestricted and contextualized. *Proceedings of ICT Africa conference*, February 2008. Addis Ababa - Ethiopia.

[5] M. Thinyane, A. Terzoli, and P. Clayton. Transitions towards a knowledge society: Aspectual pre-evaluation of a culture-sensitive implementation framework. *Springer IFIP - Learning of live in the knowledge society*, 281:271–278, July 2008. Boston - USA.

[6] A. Flor. ICT and Poverty: The Indisputable Link. *Proceedings of Third Asia Development Forum on Regional Economic Cooperation in Asia and the Pacific*, pages 11–14, June 2001. Bangkok - Thailand.

[7] P. Inamdar. Computer skills development by children using 'hole in the wall' facilities in rural India. *Australasian Journal of Educational Technology*, 20(3):337–350, November 2004.

[8] H. Slay, P. Wentworth, and J. Locke. Bingbee, an information kiosk for social enablement in marginalized communities. *Proceedings of the ACM South African Institute of Computer Scientists and Information Technologists conference (SAICSIT)*, pages 107–116, October 2006. New York - USA.

[9] K. Gush, G. Cambridge, and R. Smith. The Digital Doorway-minimally invasive education in Africa. *ICT in Education Conference*, 2004.

[10] T. Berners-Lee. *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web*. HarperCollins Publishers, 1999. New York - USA.

[11] M.E. Pierce, G. Fox, H. Yuan, and Y. Deng. Cyberinfrastructure and Web 2.0. In *Proceedings of High Performance Computing 2006*, volume 16 of *Advances in Parallel Computing*. IOS Press, July 2006.

[12] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic Web. *Scientific American*, 284(5):34–43, May 2001.

[13] D. Fensel. Language standardization for the Semantic Web: The long way from OIL to OWL. *Lecture notes in computer science*, pages 215–227, 2002.

[14] J. Heflin and J. Hendler. Semantic Interoperability on the Web. *Extreme Markup Languages 2000*, 4(5):7, 2000.

[15] Y. Yao, H. Wang, Q. Hu, G. Xu, and D. Wu. The research of medical services sharing platform base on semantic web services. *Proceedings of 7th Asian-Pacific conference on medical and biological engineering (APCMBE)*, 19:584–587, November 2008. Beijing - China.

[16] LF Pau. Artificial intelligence and financial services. *IEEE Transactions on Knowledge and Data Engineering*, 3(2):137–148, June 1991.

[17] NV Findler and R. Lo. Distributed artificial intelligence approach to air traffic control. *IEEE Proceedings D on Control Theory and Applications*, 138(6):515–524, November 1991.

[18] M.E.I. Kipp and D.G. Campbell. Patterns and Inconsistencies in Collaborative Tagging Systems: An Examination of Tagging Practices. *Proceedings of the 2006 Annual Meeting of the American Society for Information Science and Technology (ASIST)*, November 2006. Austin - USA.

[19] Flickr. Online. http://www.flickr.com/.

[20] Del.icio.us. Online. http://delicious.com/.

[21] Facebook. Online. http://www.facebook.com/.

[22] Orkut. Online. http://www.orkut.com.

[23] Myspace. Online. http://www.myspace.com/.

[24] Amazon. Online. http://www.amazon.com/.

[25] ebay. Online. http://www.ebay.com.

[26] T. Sollazzo, S. Handschuh, S. Staab, and M. Frank. Semantic Web Service Architecture–Evolving Web Service Standards toward the Semantic Web. *Proceedings of the 15th International FLAIRS Conference*, pages 425–429, May 2002. Florida - USA.

[27] D. Martin, M. Paolucci, S. Mcllraith, M. Burstein, D. McDermott, D. McGuinness, B. Parsia, T. Payne, M. Sabou, M. Solanki, et al. Bringing Semantics to Web Services: The OWL-S Approach. *First International Workshop on Semantic Web Services and Web Process Composition (SWSWPC)*, pages 6–9, July 2005. San Diego - USA.

[28] A. Ankolekar, M. Burstein, J. Hobbs, O. Lassila, D. Martin, D. McDermott, S. McIlraith, S. Narayanan, M. Paolucci, T. Payne, et al. DAML-S: Web Service Description for the Semantic Web. *Proceedings of International Semantic Web Conference*, pages 348–363, July 2002. Italy.

[29] Gartner. Worldwide software market for soa, web services and web2.0. Technical report, Gartner, http://www.gartner.com, 2007.

[30] H.E. Hudson. Extending Access to the Digital Economy to Rural and Developing Regions. *Understanding the Digital Economy: Data, Tools, and Research*, pages 261–291, 2002.

[31] P. Conradie and SJ Jacobs. Bridging the digital divide. *Engineering Management Journal*, 13(1):30–33, Feb 2003.

[32] M. Thinyane, H. Slay, A. Terzoli, and P. Clayton. A preliminary investigation into the implementation of ICTs in marginalized communities. *Proceedings of Southern African Telecommunications Networks and Applications Conference (SATNAC)*, September 2006. Western Cape - South Africa.

[33] S. Dyakalashe, M. Thinyane, H. Muyingi, and A. Terzoli. Cultural and lingustic localization of the shop-owner interfaces to an e-commerce platform for rural development. *Proceedings of Southern African Telecommunications Networks and Applications Conference (SATNAC)*, September 2008. South Africa.

[34] R. Florida and M. Kenney. The New Age of Capitalism: Innovation-mediated production. *Futures: The journal of Forecasting and planning*, 25(6):637–652, July 1993.

[35] M. Castells. *Information technology, globalization and social development*. United Nations Research Institute for Social Development, September 1999. Geneva.

[36] J. Hite. The Thunen Model and the New Economic Geography as a Paradigm for Rural Development Policy. *Review of Agricultural Economics*, 19(2):230–240, 1997.

[37] M. Drabenstott. New Policies for a New Rural America. *International Regional Science Review*, 24(1):3, 2001.

[38] E.J. Malecki. Digital development in rural areas: potentials and pitfalls. *Journal of Rural Studies*, 19(2):201–214, 2003.

[39] S. Mitchell and D. Clark. Business adoption of information and communications technologies in the two-tier rural economy: some evidence from the South Midlands. *Journal of Rural Studies*, 15(4):447–455, 1999.

[40] R. Gibson, K. Rene, and P. Saeed. Governance for sustainable development - moving from theory to practice. *International Journal of Sustainable Development*, 8(1/2):12–30, 2005.

[41] G.H. Brundtland et al. *Our common future*. Oxford University Press, 1987.

[42] O. Hietanen, S. Heinonen, K. Kiiskilä, J. Lyytimäki, and U. Rosenström. Indicators of Sustainable Information Society. *Turun kauppakorkeakoulu, Tulevaisuuden tutkimuskeskus. Tutu-julkaisu*, 1:2004, 2004.

[43] H. Dooyeweerd. *A New Critique of Theoretical Thought*. Edwin Mellen Press Lewiston, February 1997. New York - USA.

[44] O. Hietanen. The Global Challenges of eDevelopment - From Digital Divides Towards Empowerment and Sustainable Global Information Society. In *Seminar of global perspectives of development communication*. University of Tampere, June 2004.

[45] D.W. Pearce, A. Markandya, and E. Barbier. *Blueprint for a Green Economy*. Earthscan Publications, January 1989. London - UK.

[46] W. Norman and C. MacDonald. Getting to the Bottom of "Triple Bottom Line". *Business Ethics Quarterly*, 14(2):243–262, April 2004.

[47] J.H. Spangenberg. Environmental space and the prism of sustainability: frameworks for indicators measuring sustainable development. *Elsevier - Ecological Indicators*, 2(3):295–309, December 2002.

[48] S. Bhatnagar. Information Technology and Development Foundation and Key Issues. *Information and Communication Technology in Rural development: Case Studies from India. World Bank Institute*, pages 1–12, 2000.

[49] J.G. Saxe. *The Poems of John Godfrey Saxe*. Ticknor and Fields, 1855.

[50] B. Singh. *Information Technology for Rural Development in India*. PhD thesis, State University of New York at Buffalo, 2002.

[51] S. Grimes. Rural areas in the information society: diminishing distance or increasing learning capacity? *Journal of Rural Studies*, 16(1):13–21, January 2000.

[52] H. Mitomo and H. Oniki. Information technology for sustainable societies–public policy perspectives in Japan: the case of telework. *The IPTS Report*, pages 24–31, 1932.

[53] B. Wellman, A.Q. Haase, J. Witte, and K. Hampton. Does the Internet Increase, Decrease, or Supplement Social Capital? Social Networks, Participation, and Community Commitment. *American Behavioral Scientist*, 45(3):436, 2001.

[54] W.F. Fox and S. Porca. Investing in Rural Infrastructure. *International Regional Science Review*, 24(1):103, 2001.

[55] J. Feather. *The information society: a study of continuity and change*. Library Association, 1994.

[56] S. Strover. Rural Internet Connectivity. *Telecommunications Policy*, 25(5):331–347, 2001.

[57] E. Lorenzo. Photovoltaic Rural Electrification. *Progress in Photovoltaics Research and Applications*, 5(1):3–27, 1997.

[58] N. Negroponte. One Laptop per Child (OLPC). Online, 2006. http://laptop.org.

[59] Aleutia Limited. The aleutia e2 : An 8 watt green pc for everyone. Online, 2008. http://www.aleutia.com/.

[60] M. Thinyane, L. Dalvit, A. Terzoli, and H. Muyingi. The deployment of an e-commerce platform and related projects in a rural area in south africa. *ACM International Journal of Computing and ICT Research*, 1(1):9–17, June 2007. Kampala - Uganda.

[61] R. Huggins. The Digital Divide and ICT Learning in Rural Communities: Examples of Good Practice Service Delivery. *Local Economy*, 17(2):111–122, 2002.

[62] M. Thinyane, L. Dalvit, A. Terzoli, and N. Isabirye. A case study on the teaching of computer literacy in a marginalized community. *Proceedings of Comparative Education Society of Europe Conference*, July 2006. Granada - Spain.

[63] K.H. Moahi. Health Information Networks for Telehealth in Africa–Challenges and Prospects: a Review of the Literature. *Libri(Copenhagen)*, 49(1):43–50, 1999.

[64] M. Chetty, WD Tucker, and EH Blake. Telemedicine using VoIP combined with a Store and Forward Approach. *Proceedings of South African Telecommunications Networks and Applications Conference (SATNAC)*, September 2004. Stellenbosch - South Africa.

[65] P. Corr, I. Couper, SJ Beningfield, and M. Mars. A simple telemedicine system using a digital camera. *Journal of Telemedicine and Telecare*, 6(4):233–236, 2000.

[66] J.C. Leatherman. Internet-Based Commerce: Implications for Rural Communities. *Reviews of Economic Development Literature and Practice*, 5, September 2000. Washington DC - USA.

[67] M. Thinyane, A. Terzoli, and P. Clayton. Exploring a novel service deployment framework, piask, through the design and implementation of an e-commerce function for a rural community. *Proceedings of Southern African Telecommunications Networks and Applications Conference (SATNAC)*, September 2007. Mauritius.

[68] M. Tedre, E. Sutinen, E. Kahkonen, and P. Kommers. Appreciating the knowledge of students in computer science education in developing countries. *Proceedings of International Conference on Information Technology Research and Education (ITRE2003)*, pages 174–178, August 2003. New Jersey - USA.

[69] L.A. Zadeh. Coping with the imprecision of the real world. *Communications of the ACM*, 27(4):304–311, 1984.

[70] Richard A. Bolt. "put-that-there": Voice and gesture at the graphics interface. *Proceedings of the 7th annual conference on Computer graphics and interactive techniques*, 14(3):262–270, July 1980. New York - USA.

[71] C. Stephanidis. User interfaces for all: New perspectives into human-computer interaction. *User Interfaces for All–concepts, methods and tools*, pages 3–17, 2001.

[72] V. Kaptelinin. Activity theory: Implications for human computer interaction. *NATO ASI Series Computer and Systems Sciences*, 129:5–5, 1994.

[73] G.A. Senf. The art of language. *Journal of Language and Linguistics*, 1(4):388–433, 2002.

[74] K. Smith. What is the 'Knowledge Economy'? Knowledge intensity and distributed knowledge bases. *Conference on The Learning Economy*, 2002. Norway.

[75] A.W. Khan. Towards knowledge societies. *World of Science*, 1(4):8–9, July-September 2003.

[76] J. Houghton and P. Sheehan. A Primer on the Knowledge Economy. *Centre for Strategic Economic Studies, Victoria University.*, 23, February 2000. Melborne - Australia.

[77] N. Gorjestani. Indigenous Knowledge for Development. *UNCTAD Conference on Traditional Knowledge*, November 2000. Geneva.

[78] R. Woytek, N. Gorjestani, and Africa Regional Office. *Indigenous Knowledge for Development: A Framework for Action*. World Bank, November 1998.

[79] Wikipedia. Wikipedia. Online. http://en.wikipedia.org.

[80] E. Gettier. Is Justified True Belief Knowledge? *Analysis*, 23(6):121–123, 1963.

[81] A.E. Lawson. How Do Humans Acquire Knowledge? And What Does That Imply About the Nature of Knowledge? *Science & Education*, 9(6):577–598, 2000.

[82] C.T. Fosnot. *Constructivism. Theory, Perspectives, and Practice.* Teachers College Press, New York - USA, 2nd edition edition, April 2005.

[83] A.S. Hwang. Positivist and constructivist persuasions in instructional development. *Instructional Science*, 24(5):343–356, 1996.

[84] J.F. Osborne. Beyond Constructivism. *Science Education*, 80(1):53–82, 1996.

[85] M. Polanyi. The Tacit Dimension. *Knowledge in Organizations*, pages 135–146, 1997. Boston - USA.

[86] J. McCarthy. Artificial intelligence, logic and formalizing common sense. *Philosophical Logic and Artificial Intelligence*, pages 161–190, 1989.

[87] I. Nonaka, R. Toyama, and N. Konno. SECI, Ba and Leadership: A Unified Model of Dynamic Knowledge Creation. *Elsevier - Knowledge Management*, 33(1):5–34, February 2005.

[88] Philippe Perez. Knowlege management models: a state of the art. Technical report, KnowledgeBoard, November 2002.

[89] Y. Engeström et al. Comment on Blackler et al. Activity Theory and the Social Construction of Knowledge: a Story of Four Umpires. *Organization*, 7(2):301–310, 2000.

[90] D. Snowden. A framework for creating a sustainable knowledge management program. *The Knowledge Management Yearbook 1999-2000*, pages 52–65, April 1999.

[91] M. Korotkiy and J. Top. Onto-SOA: From Ontology-enabled SOA to Service-enabled Ontologies. *Proceedings of the Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services*, page 124, February 2006. Guadeloupe - French Caribbean.

[92] B.C. Grau. A possible simplification of the semantic web architecture. *Proceedings of the 13th ACM international conference on World Wide Web*, pages 704–713, May 2004. New York - USA.

[93] S.K. Das. Role of semantic web in the changing context of digital environment. Online, February 2007. https://drtc.isibang.ac.in/handle/1849/397.

[94] P. Hayes and C. Menzel. A semantics for the knowledge interchange format. In *IJCAI 2001 Workshop on the IEEE Standard Upper Ontology*, Washington - USA, August 2001.

[95] T. Berners-Lee. WWW Past and Future. 1-hour talk plus 25 minutes of questions. *The Royal Society*, 2003. London - UK.

[96] M. Kifer, J. de Bruijn, H. Boley, and D. Fensel. A Realistic Architecture for the Semantic Web. *Proceedings of International Conference on Rules and Rule Markup Languages for the Semantic Web*, pages 17–29, November 2005. Galway - Ireland.

[97] P.F. Patel-Schneider, P. Hayes, and I. Horrocks. OWL Web Ontology Language Semantics and Abstract Syntax. *W3C Recommendation*, 10, 2004.

[98] I. Horrocks, P.F. Patel-Schneider, and F. van Harmelen. From SHIQ and RDF to OWL: The making of a web ontology language. *Journal of Web Semantics*, 1(1):7–26, 2003.

[99] I. Horrocks, B. Parsia, P. Patel-Schneider, and J. Hendler. Semantic Web Architecture: Stack or Two Towers? *Proceedings of Principles and Practice of Semantic Web Reasoning: Third International Workshop (PPSWR)*, September 2005. Germany.

[100] R. Davis, H. Shrobe, and P. Szolovits. What Is a Knowledge Representation? *AI Magazine*, 14(1):17–33, 1993.

[101] F. Baader. Logic-based knowledge representation. *Artificial Intelligence Today, Recent Trends and Developments*, pages 13–41, 1999.

[102] W3C. Comparing rule-based systems. Online, April 2003. http://www.w3.org/2000/10/swap/doc/rule-systems.

[103] J. Mylopoulos. An overview of Knowledge Representation. *Proceedings of the ACM workshop on Data abstraction, databases and conceptual modelling*, 11(2):5–12, February 1981. New York - USA.

[104] B. Nebel and G. Smolka. Attributive description formalisms... and the rest of the world. *Text Understanding in LILOG*, pages 439–452, 1991. New York.

[105] B. Nebel. Frame-Based Systems. *MIT Encyclopedia of the Cognitive Sciences, MIT Press,*, 1999. Massachusetts - USA.

[106] P. Gray. *Logic, algebra and databases*. John Wiley and Sons, Inc, New York - USA, 1984.

[107] M. Poget. Using semantic nets to model troubleshooting's knowledge. *Troubleshooting Professional Magazine [Online]*, 3(7), July 1999. http://www.troubleshooters.com/tpromag/199907/model.htm.

[108] E. Horvitz. Machine Learning, Reasoning, and Intelligence in Daily Life: Directions and Challenges. *Proceedings of ICML*, 1999.

[109] Garth Kemerling. Causal reasoning. In *Philosophy Pages*. http://www.philosophypages.com/lg/e14.htm, 2007.

[110] J.S. Mill. A system of Logic Ratiocinative and Inductive (1843). *Collected Works of John Stuart Mill*, 7, 1974.

[111] C. Picardi, P. Salles, and F. Wotawa. An introduction to model-based systems. *AI Communications*, 20(1):1–6, 2007.

[112] A. Aamodt and E. Plaza. Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. *AI Communications*, 7(1):39–59, March 1994.

[113] R. Dieng, O. Corby, A. Giboin, and M. Ribiere. Methods and tools for corporate knowledge management. *International Journal of Human-Computers Studies*, 51(3):567–598, 1999.

[114] B. Chandrasekaran, JR Josephson, and VR Benjamins. What are ontologies, and why do we need them? *IEEE Intelligent Systems and Their Applications*, 14(1):20–26, 1999.

[115] N.J. Davies, D. Fensel, and F. Van Harmelen. *Towards the Semantic Web: Ontology-Driven Knowledge Management*. John Wiley and Sons, 2003.

[116] D.B. Lenat, RV Guha, K. Pittman, D. Pratt, and M. Shepherd. Cyc: toward programs with common sense. *Communications of the ACM*, 33(8):30–49, 1990.

[117] C. Fellbaum. *Wordnet: an electronic lexical database*. MIT Press, 1998.

[118] M. Hwang, H. Kong, and P. Kim. The Design of the Ontology Retrieval System on the Web. *Proceedings of The 8th International Conference Advanced Communication Technology (ICACT)*, 3, 2006.

[119] J.Z. Pan and I. Horrocks. RDFS (FA): Connecting RDF (S) and OWL DL. *IEEE Transactions on Knowledge and Data Engineering*, 19(2):192–206, 2007.

[120] S. Bechhofer, F. van Harmelen, J. Hendler, I. Horrocks, D.L. McGuinness, P.F. Patel-Schneider, L.A. Stein, et al. OWL Web Ontology Language Reference. *W3C Candidate Recommendation*, August 2003. http://www. w3. org/TR/2003/CR-owl-ref-20030818.

[121] O. Corcho, M. Fernández-López, and A. Gómez-Pérez. Methodologies, tools and languages for building ontologies. Where is their meeting point? *Data & Knowledge Engineering*, 46(1):41–64, 2003.

[122] H. Knublauch, R.W. Fergerson, N.F. Noy, and M.A. Musen. The Protege OWL Plugin: An Open Development Environment for Semantic Web Applications. *Lecture Notes in Computer Science*, pages 229–243, 2004.

[123] S. Bechhofer, I. Horrocks, C. Goble, and R. Stevens. OilEd: A Reason-able Ontology Editor for the Semantic Web. *Lecture Notes in Computer Science*, pages 396–408, 2001.

[124] S. Auer. Powl: A web based platform for collaborative semantic web development. *Proceedings of the Workshop Scripting for the Semantic Web (SFSW)*, May 2005. Greece.

[125] C. Wagner. WIKI: A technology for conversational knowledge management and group collaboration. *Communications of the Association for Information Systems*, 13(9):265–289, 2004.

[126] F.K. Hussain, A.S. Sidhu, T.S. Dillon, and E. Chang. Engineering Trustworthy Ontologies: Case Study of Protein Ontology. *Proceedings of the 19th IEEE Symposium on Computer-Based Medical Systems*, pages 617–622, June 2006. Utah - USA.

[127] P. Hayes. RDF Semantics. *W3C Recommendation*, 10, 2004.

[128] R. Chapman. *Rogets International Thesaurus*. Collins, 1992.

[129] L. Urdang. *The Basic Book of Synonyms and Antonyms*. Signet, 1986.

[130] I. Niles and A. Pease. Towards a standard upper ontology. *Proceedings of the international conference on Formal Ontology in Information Systems-Volume 2001*, pages 2–9, 2001.

[131] G. Delanty. Challenging Knowledge: the university in the knowledge society. *Teaching in Higher Education*, 7(2), 2002.

[132] A. Dogac, G. Laleci, Y. Kabak, and I. Cingil. Exploiting Web Service Semantics: Taxonomies vs. Ontologies. *IEEE Data Engineering Bulletin*, 25(4):10–16, 2002.

[133] M. Ashburner, CA Ball, JA Blake, D. Botstein, H. Butler, JM Cherry, AP Davis, K. Dolinski, SS Dwight, JT Eppig, et al. Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. *Nat Genet*, 25(1):25–9, 2000.

[134] Swoogle - semantic web search. Online. http://swoogle.umbc.edu/.

[135] H. Kong, M. Hwang, and P. Kim. A New Methodology for Merging the Heterogeneous Domain Ontologies Based on the WordNet. *International Conference on Next Generation Web Services Practices (NWeSP)*, pages 235–240, 2005.

[136] Y. Kalfoglou and M. Schorlemmer. Ontology mapping: the state of the art. *The Knowledge Engineering Review*, 18(01):1–31, 2003.

[137] T. Milo and S. Zohar. Using schema matching to simplify heterogeneous data translation. *Proceedings of the International Conference on Very Large Databases (VLDB)*, 133, 1998.

[138] D.L. McGuinness, R. Fikes, J. Rice, and S. Wilder. An environment for merging and testing large ontologies. *Proceedings of the Seventh International Conference on Principles of Knowledge Representation and Reasoning (KR2000)*, pages 483–493, 2000.

[139] D. Calvanese, DG Giuseppe, and M. Lenzerini. Ontology of Integration and Integration of Ontologies. *Proceedings of the 2001 Description Logic Workshop (DL 2001)*, pages 10–19, 2001.

[140] J. Liebowitz. *Knowledge Management: Learning from Knowledge Engineering*. CRC Press, 2001.

[141] P. De Leenheer and S. Christiaens. Mind the gap!: Transcending the tunnel view on ontology engineering. *Proceedings of the 2nd international conference on Pragmatic web*, pages 75–82, 2007.

[142] A. Passant. Using Ontologies to Strengthen Folksonomies and Enrich Information Retrieval in Weblogs. *International Conference on Weblogs and Social Media*, March 2007.

[143] E. Neumann and L. Prusak. Knowledge networks in the age of semantic web. *Briefings in Bioinformatics*, 8(3), May 2007.

[144] H. Wu, M. Zubair, and K. Maly. Harvesting social knowledge from folksonomies. *Proceedings of the seventeenth conference on Hypertext and hypermedia*, pages 111–114, 2006.

[145] T. Berners-Lee. Giant global graph, November 2007. http://dig.csail.mit.edu/breadcrumbs/node/215.

[146] P. Kollock. The Economies of Online Cooperation: Gifts and Public Goods in Computer Communities. *Communities in cyberspace. New York: Routledge*, pages 221–239, 1997.

[147] A.J. Kim. *Community Building on the Web: Secret Strategies for Successful Online Communities*. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 2000.

[148] M.T. Maybury et al. *Intelligent multimedia interfaces*. MIT Press Cambridge, Massachusetts - USA, 1993.

[149] S. Oviatt. Designing robust multimodal systems for universal access. In *Proceedings of the 2001 EC/NSF workshop on Universal accessibility of ubiquitous computing*, pages 71–74, New York, NY, USA, 2001. ACM Press.

[150] B. Xiao, R. Lunsford, R. Coulston, M. Wesson, and S. Oviatt. Modeling multimodal integration patterns and performance in seniors: toward adaptive processing of individual differences. *Proceedings of the 5th international conference on Multimodal interfaces*, pages 265–272, 2003.

[151] S. Oviatt and P. Cohen. Perceptual user interfaces: multimodal interfaces that process what comes naturally. *Communications of the ACM*, 43(3):45–53, 2000.

[152] S. Oviatt, P. Cohen, L. Wu, J. Vergo, L. Duncan, B. Suhm, J. Bers, T. Holzman, T. Winograd, J. Landay, et al. Designing the user interface for multimodal speech and pen-based gesture applications: State-of-the-art systems and future research directions. *Human-Computer Interaction*, 15(4):263–322, 2000.

[153] George Cybenko, Robert Gray, Alexy Khrabrov, and Yunxin Wu. Information theoretic principles of agents. In Yannis Labrou and Tim Finin, editors, *Proceedings of the CIKM Workshop on Intelligent Information Agents, Third International Conference on Information and Knowledge Management (CIKM 94)*, Gaithersburg, MD, 1994.

[154] F. Leymann. Web Services: Distributed Applications without Limits. *Business, Technology and Web*, February 2003. Leipzig - Germany.

[155] R.T. Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, University of California, 2000.

[156] D. Fensel and C. Bussler. The Web Service Modeling Framework WSMF. *Electronic Commerce Research and Applications*, 1(2):113–137, 2002.

[157] L. Cabral, J. Domingue, E. Motta, T. Payne, and F. Hakimpour. Approaches to Semantic Web Services: An Overview and Comparisons. *Lecture Notes in Computer Science*, 3053:225–239, 2004.

[158] Terry R. Payne. Web services from an agent perspective. *IEEE Intelligent Systems*, 23(2):12–14, 2008.

[159] M. Nikraz, G. Caire, and PA Bahri. A methodology for the analysis and design of multi-agent systems using JADE. *International Journal of Computer Systems Science and Engineering*, 21(2), May 2006.

[160] R. Leszczyna. Evaluation of agent platforms. Technical report, Institute for the Protection and Security of the Citizen, European Commission, Joint Research Centre, June 2004. Italy.

[161] Java Community Process. Jsr 87: Java agent services. Online, 2002. http://jcp.org/en/jsr/detail?id=87.

[162] JINI Community. Jini. Online, March 2007. http://www.jini.org.

[163] JXTA Community Project. Jxta. Online, 2008. http://www.jxta.org/.

[164] World Wide Web Consortium. Simple object acess protocol (soap) 1.2. Online, April 2007. http://www.w3.org/TR/soap12/.

[165] A. Cheyer and D. Martin. The Open Agent Architecture. *Autonomous Agents and Multi-Agent Systems*, 4:143–148, 2001.

[166] F. Bellifemine, A. Poggi, and G. Rimassa. JADE: a FIPA2000 compliant agent development environment. In *Proceedings of the fifth international conference on Autonomous agents*, pages 216–217. ACM New York, NY, USA, 2001.

[167] F. Bergenti and A. Poggi. LEAP: A FIPA Platform for Handheld and Mobile Devices. *Lecture Notes in Computer Science*, pages 436–446, 2002.

[168] C. Baumer, M. Breugst, S. Choy, and T. Magedanz. Grasshopper - A universal agent platform based on OMG MASIF and FIPA standards. In *First International Workshop on Mobile Agents for Telecommunication Applications (MATA)*, pages 1–18, 1999.

[169] D.B. Lange and M. Oshima. Mobile agents with Java: The Aglet API. *World Wide Web*, 1(3):111–121, 1998.

[170] Protege. Online. http://protege.stanford.edu/.

[171] Ihmc cmaptools. Online. http://cmap.ihmc.us/.

[172] Jena - a semantic web framework for java. Online. http://jena.sourceforge.net/.

[173] Redland rdf libraries. Online. http://librdf.org/.

[174] Rap - rdf api for php. Online. http://rdfapi-php.sf.net.

[175] V.K. Chaudhri, A. Farquhar, R. Fikes, P.D. Karp, and J.P. Rice. OKBC: a programmatic foundation for knowledge base interoperability. In *Proceedings of the National Conference on Artificial Intelligence*, pages 600–607. American Association for Artificial Intelligence, 1998. USA.

[176] R. Palmer, H. Timmermans, and D. Fay. *From conflict to Negotiation: Nature-based development of South Africa's Wild Coast*. Human Sciences Research Council, 2002.

[177] W. Barber and A. Badre. Culturability: The Merging of Culture and Usability. *Proceedings of 4th Conference on Human Factors & the Web*, 1998.

[178] Wikipedia. Living lab. Online, 2008. http://en.wikipedia.org/wiki/Living lab.

[179] P. Tarwireyi, M. Thinyane, and A. Terzoli. Implementation of an internet access cost management system for disadvantaged communities. *Proceedings of Southern African Telecommunications Networks and Applications Conference (SATNAC)*, September 2007. Mauritius.

[180] O.F. Arnold and A. Bruce. Nursing Practice With Aboriginal Communities: Expanding Worldviews. *Nursing Science Quarterly*, 18(3):259, 2005.

[181] Michael Hart. Gutenberg: The history and philosophy of project guternberg. Online, August 1992. http://www.gutenberg.org/wiki/Gutenberg:The History and Philosophy of Project Gutenberg by Michael Hart.

[182] Software Engineering Standards Committee. Ieee std 830-1998 ieee recommended practice for software requirements specifications. *IEEE Computer Society*, October 1998.

[183] PB Kruchten. The 4+ 1 View Model of architecture. *Software, IEEE*, 12(6):42–50, 1995.

[184] N. Sangal, E. Jordan, V. Sinha, and D. Jackson. Using dependency models to manage complex software architecture. In *Proceedings of the 20th annual ACM SIGPLAN conference on Object oriented programming, systems, languages, and applications*, pages 167–176. ACM New York, NY, USA, 2005.

[185] T. Reenskaug. Mvc: Xerox parc 1978-79. Online, 2003. http://heim.ifi.uio.no/ trygver/themes/mvc/mvc-index.html.

[186] M. Potel. Mvp: Model view presenter, the taligent programming modelf for c++ and java. Online, 1996. http://www.wildcrest.com/Potel/Portfolio/mvp.pdf.

[187] J. Coutaz. PAC: an Implementation Model for Dialog Design. In *Proceedings of Interact*, volume 87, pages 431–436, 1987.

[188] R. Pawson and R. Matthews. Naked objects: a technique for designing more expressive systems. *ACM SIGPLAN Notices*, 36(12):61–67, December 2001. New York - USA.

[189] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal. *Pattern-oriented software architecture: a system of patterns*. John Wiley & Sons, Inc. New York, NY, USA, 1996.

[190] D. Garlan and M. Shaw. An Introduction to Software Architecture. *Advances in Software Engineering and Knowledge Engineering*, 1:1–40, 1993.

[191] A. Snyder. Encapsulation and inheritance in object-oriented programming languages. In *Conference on Object Oriented Programming Systems Languages and Applications*, pages 38–45. ACM New York, NY, USA, 1986.

[192] M. Sparling. Lessons learned through six years of component-based development. *Communications of the ACM*, 43(10):47–53, 2000.

[193] O. Nierstrasz, S. Gibbs, and D. Tsichritzis. Component-oriented software development. *Communications of the ACM*, 35(9):160–165, 1992.

[194] B. Schwarz. Asterisk open-source PBX system. *Linux Journal*, 2004(118), 2004.

[195] D.B. Moran, A.J. Cheyer, L.E. Julia, D.L. Martin, and S. Park. Multimodal user interfaces in the Open Agent Architecture. *Knowledge-Based Systems*, 10(5):295–303, 1998.

[196] i6net solutions and technologies. Vxi voicexml browser. Online. http://www.i6net.com/products/vxi/.

[197] Mortbay. Jetty web server. Online, December. http://www.mortbay.org/jetty/.

[198] Ignite Realtime. Smack api 3.1.0. Online, December 2008. http://www.igniterealtime.org/projects/smack/index.jsp.

[199] D. Reynolds. Jena 2 inference support. Online, December 2007. http://jena.sourceforge.net/inference/.

[200] Apache Software Foundation. Apache lucene. Online, 2008. http://lucene.apache.org/java/.

[201] O. Gospondnetic. Introduction to text indexing with apache jakarta lucene. Online - O'Reilly OnJava.com, January 2003. http://www.onjava.com/pub/a/onjava/2003/01/15/lucene.html.

[202] FIPA. Fipa acl message structure specification. Online, December 2002. http://www.fipa.org/specs/fipa00061/.

[203] Apache Software Foundation. ab - apache http server benchmarking tool. Online, 2008. http://httpd.apache.org/docs/2.0/programs/ab.html.

[204] Netbeans. Online, December 2008. http://www.netbeans.org/.

[205] A. Basden. The critical theory of Herman Dooyeweerd? *Journal of Information Technology*, 17(4):257–269, 2002.

[206] A. Basden. The dooyeweerd pages. Online, 2000. http://www.dooy.salford.ac.uk/.

[207] A. Basden and L. Patrizia. Environmental sustainability and information systems: The similarity. *Systemic Practise and Action Research*, 10(4):473–489, August 1997.

[208] H. Aay and A.B. Van Langevelde. A Dooyeweerd-based Approach to Regional Economic Development. *Tijdschrift voor Economische en Sociale Geografie*, 96(2):184–198, 2005.

[209] E. Folmer and J. Bosch. Usability Patterns in Software Architecture. *Human Computer Interaction: Theory and Practice*, 2003.

[210] E. Folmer and J. Bosch. Architecting for usability: a survey. *The Journal of Systems & Software*, 70(1-2):61–78, 2004.

[211] B. Shackel. Usability - context, framework, definition, design and evaluation. *Human Factors for informatics usability*, pages 21–37, 1991. New York - USA.

[212] J. Nielsen. *Usability Engineering*. Morgan Kaufmann, San Francisco - USA, 1993.

[213] E. Folmer, J. van Gurp, and J. Bosch. Software Architecture Analysis of Usability. *Springer - Lecture Notes in Computer Science*, 3425:38–58, July 2005.

[214] O. Lassila and J. Hendler. Embracing" Web 3.0". *IEEE Internet Computing*, pages 90–93, 2007.

[215] D.A. Schon, B. Sanyal, and W.J. Mitchell. *High Technology and Low-Income Communities: Prospects for the Positive Use of Advanced Information Technology*. MIT Press, 1999.

Appendix A

The ontologies

Figure A.1 is a graphical outline of the full ontology that is utilized in KnowNet for the encapsulation of the platform knowledge. This is followed by the RDF/OWL specification of the ontology.

Figure A.1: PIASK ontologies

```
<?xml version="1.0"?>


<!DOCTYPE rdf:RDF [
   <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
   <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
   <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
   <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
]>


<rdf:RDF xmlns="http://www.dwesa.com/piask/"
     xml:base="http://www.dwesa.com/piask/"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
    <owl:Ontology rdf:about=""/>
    <owl:Class rdf:about="accommodation">
        <rdfs:subClassOf rdf:resource="Service"/>
    </owl:Class>
    <owl:Class rdf:about="Accuracy">
        <rdfs:subClassOf rdf:resource="Weight"/>
    </owl:Class>
    <owl:Class rdf:about="Activity">
        <rdfs:subClassOf rdf:resource="Item"/>
    </owl:Class>
    <owl:Class rdf:about="Audio">
        <rdfs:subClassOf rdf:resource="Document"/>
    </owl:Class>
    <owl:Class rdf:about="Buyer">
        <rdfs:subClassOf rdf:resource="Person"/>
    </owl:Class>
    <owl:Class rdf:about="cardiovascular">
        <rdfs:subClassOf rdf:resource="Condition"/>
    </owl:Class>
    <owl:Class rdf:about="Ceremony">
        <rdfs:subClassOf rdf:resource="Activity"/>
    </owl:Class>
    <owl:Class rdf:about="church">
        <rdfs:subClassOf rdf:resource="Institution"/>
    </owl:Class>
    <owl:Class rdf:about="clinic">
        <rdfs:subClassOf rdf:resource="Institution"/>
    </owl:Class>
    <owl:Class rdf:about="Condition">
        <rdfs:subClassOf rdf:resource="Item"/>
    </owl:Class>
    <owl:Class rdf:about="cultural">
        <rdfs:subClassOf rdf:resource="Event"/>
    </owl:Class>
    <owl:DatatypeProperty rdf:about="date"/>
    <owl:Class rdf:about="dermatological">
        <rdfs:subClassOf rdf:resource="Condition"/>
    </owl:Class>
```

```
    <owl:Class rdf:about="district">
      <rdfs:subClassOf rdf:resource="Geographical"/>
    </owl:Class>
    <owl:Class rdf:about="Doctor">
      <rdfs:subClassOf rdf:resource="Person"/>
    </owl:Class>
    <owl:Class rdf:about="Document">
      <rdfs:subClassOf rdf:resource="Item"/>
    </owl:Class>
    <owl:Class rdf:about="drug">
      <rdfs:subClassOf rdf:resource="medication"/>
    </owl:Class>
    <owl:Class rdf:about="Event">
      <rdfs:subClassOf rdf:resource="Item"/>
    </owl:Class>
    <owl:Class rdf:about="excursion">
      <rdfs:subClassOf rdf:resource="Service"/>
    </owl:Class>
    <owl:Class rdf:about="Folklore">
      <rdfs:subClassOf rdf:resource="Item"/>
    </owl:Class>
    <owl:Class rdf:about="Folktale">
      <rdfs:subClassOf rdf:resource="Folklore"/>
    </owl:Class>
    <owl:Class rdf:about="food">
      <rdfs:subClassOf rdf:resource="Service"/>
    </owl:Class>
    <owl:Class rdf:about="Game">
      <rdfs:subClassOf rdf:resource="Activity"/>
    </owl:Class>
    <owl:Class rdf:about="Geographical">
      <rdfs:subClassOf rdf:resource="Location"/>
    </owl:Class>
    <owl:ObjectProperty rdf:about="hasAccessedBy">
      <rdfs:domain rdf:resource="Document"/>
      <rdfs:range rdf:resource="Person"/>
    </owl:ObjectProperty>
    <owl:DatatypeProperty rdf:about="hasAuthor">
      <rdfs:domain>
        <owl:Class>
          <owl:unionOf rdf:parseType="Collection">
            <owl:Class rdf:about="Document"/>
            <owl:Class rdf:about="Folklore"/>
          </owl:unionOf>
        </owl:Class>
      </rdfs:domain>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:about="hasCellNumber">
      <rdfs:domain rdf:resource="Person"/>
      <rdfs:range rdf:resource="&xsd;string"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:about="hasCharacters">
      <rdfs:domain rdf:resource="Folktale"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:about="hasCreated">
```

```
111         <rdfs:domain rdf:resource="Document"/>
112       </owl:DatatypeProperty>
113       <owl:DatatypeProperty rdf:about="hasDate">
114         <rdfs:domain>
115           <owl:Class>
116             <owl:unionOf rdf:parseType="Collection">
117               <owl:Class rdf:about="Event"/>
118               <owl:Class rdf:about="Tagging"/>
119             </owl:unionOf>
120           </owl:Class>
121         </rdfs:domain>
122         <rdfs:range rdf:resource="&xsd;dateTime"/>
123       </owl:DatatypeProperty>
124       <owl:DatatypeProperty rdf:about="hasDescription">
125         <rdfs:domain rdf:resource="Item"/>
126         <rdfs:range rdf:resource="&xsd;string"/>
127       </owl:DatatypeProperty>
128       <owl:DatatypeProperty rdf:about="hasEmailAddress">
129         <rdfs:domain>
130           <owl:Class>
131             <owl:unionOf rdf:parseType="Collection">
132               <owl:Class rdf:about="Buyer"/>
133               <owl:Class rdf:about="Manufacturer"/>
134               <owl:Class rdf:about="Owner"/>
135               <owl:Class rdf:about="Person"/>
136               <owl:Class rdf:about="Seller"/>
137             </owl:unionOf>
138           </owl:Class>
139         </rdfs:domain>
140         <rdfs:range rdf:resource="&xsd;string"/>
141       </owl:DatatypeProperty>
142       <owl:ObjectProperty rdf:about="hasEvents">
143         <rdfs:domain rdf:resource="Location"/>
144         <rdfs:range rdf:resource="Event"/>
145       </owl:ObjectProperty>
146       <owl:DatatypeProperty rdf:about="hasIcon">
147         <rdfs:domain rdf:resource="Document"/>
148       </owl:DatatypeProperty>
149       <owl:DatatypeProperty rdf:about="hasId">
150         <rdfs:domain rdf:resource="Document"/>
151       </owl:DatatypeProperty>
152       <owl:DatatypeProperty rdf:about="hasIdKey">
153         <rdfs:domain rdf:resource="Person"/>
154         <rdfs:range rdf:resource="&xsd;string"/>
155       </owl:DatatypeProperty>
156       <owl:DatatypeProperty rdf:about="hasIMAddress">
157         <rdfs:domain>
158           <owl:Class>
159             <owl:unionOf rdf:parseType="Collection">
160               <owl:Class rdf:about="Buyer"/>
161               <owl:Class rdf:about="Manufacturer"/>
162               <owl:Class rdf:about="Owner"/>
163               <owl:Class rdf:about="Person"/>
164               <owl:Class rdf:about="Seller"/>
165             </owl:unionOf>
```

```
166           </owl:Class>
167         </rdfs:domain>
168       </owl:DatatypeProperty>
169       <owl:DatatypeProperty rdf:about="hasImage">
170         <rdfs:domain rdf:resource="Item"/>
171         <rdfs:range rdf:resource="&xsd;string"/>
172       </owl:DatatypeProperty>
173       <owl:DatatypeProperty rdf:about="hasLastAccess">
174         <rdfs:domain rdf:resource="Document"/>
175       </owl:DatatypeProperty>
176       <owl:DatatypeProperty rdf:about="hasLastModify">
177         <rdfs:domain rdf:resource="Document"/>
178       </owl:DatatypeProperty>
179       <owl:DatatypeProperty rdf:about="hasLocation"/>
180       <owl:DatatypeProperty rdf:about="hasMass">
181         <rdfs:domain rdf:resource="Product"/>
182         <rdfs:range rdf:resource="&xsd;decimal"/>
183       </owl:DatatypeProperty>
184       <owl:DatatypeProperty rdf:about="hasName">
185         <rdfs:domain>
186           <owl:Class>
187             <owl:unionOf rdf:parseType="Collection">
188               <owl:Class rdf:about="Item"/>
189               <owl:Class rdf:about="Person"/>
190               <owl:Class rdf:about="Tag"/>
191             </owl:unionOf>
192           </owl:Class>
193         </rdfs:domain>
194         <rdfs:range rdf:resource="&xsd;string"/>
195       </owl:DatatypeProperty>
196       <owl:ObjectProperty rdf:about="hasOwner">
197         <rdfs:domain rdf:resource="Document"/>
198         <rdfs:range rdf:resource="Person"/>
199       </owl:ObjectProperty>
200       <owl:DatatypeProperty rdf:about="hasPermission">
201         <rdfs:domain rdf:resource="Document"/>
202       </owl:DatatypeProperty>
203       <owl:DatatypeProperty rdf:about="hasPhoneNumber">
204         <rdfs:domain rdf:resource="Person"/>
205         <rdfs:range rdf:resource="&xsd;string"/>
206       </owl:DatatypeProperty>
207       <owl:DatatypeProperty rdf:about="hasPrice">
208         <rdfs:domain rdf:resource="Item"/>
209         <rdfs:range rdf:resource="&xsd;decimal"/>
210       </owl:DatatypeProperty>
211       <owl:ObjectProperty rdf:about="hasProducts">
212         <rdfs:range rdf:resource="Product"/>
213       </owl:ObjectProperty>
214       <owl:ObjectProperty rdf:about="hasTagging">
215         <rdfs:domain rdf:resource="Tag"/>
216         <rdfs:range rdf:resource="Tagging"/>
217       </owl:ObjectProperty>
218       <owl:Class rdf:about="herb">
219         <rdfs:subClassOf rdf:resource="medication"/>
220       </owl:Class>
```

```
    <owl:Class rdf:about="historic">
      <rdfs:subClassOf rdf:resource="Event"/>
    </owl:Class>
    <owl:Class rdf:about="hospital">
      <rdfs:subClassOf rdf:resource="Institution"/>
    </owl:Class>
    <owl:Class rdf:about="Image">
      <rdfs:subClassOf rdf:resource="Document"/>
    </owl:Class>
    <owl:Class rdf:about="individual">
      <rdfs:subClassOf rdf:resource="Event"/>
    </owl:Class>
    <owl:Class rdf:about="infection">
      <rdfs:subClassOf rdf:resource="Condition"/>
    </owl:Class>
    <owl:Class rdf:about="Institution">
      <rdfs:subClassOf rdf:resource="Location"/>
    </owl:Class>
    <owl:ObjectProperty rdf:about="isManufacturedBy">
      <rdfs:domain rdf:resource="Product"/>
      <rdfs:range rdf:resource="Manufacturer"/>
    </owl:ObjectProperty>
    <owl:ObjectProperty rdf:about="isOwnedBy">
      <rdfs:domain rdf:resource="Item"/>
      <rdfs:range rdf:resource="Person"/>
    </owl:ObjectProperty>
    <owl:Class rdf:about="Item"/>
    <owl:ObjectProperty rdf:about="knows">
      <rdfs:domain rdf:resource="Person"/>
      <rdfs:range rdf:resource="Person"/>
    </owl:ObjectProperty>
    <owl:Class rdf:about="Location"/>
    <owl:Class rdf:about="Manufacturer">
      <rdfs:subClassOf rdf:resource="Person"/>
    </owl:Class>
    <owl:ObjectProperty rdf:about="manufactures">
      <rdfs:domain rdf:resource="Manufacturer"/>
      <rdfs:range rdf:resource="Product"/>
    </owl:ObjectProperty>
    <owl:Class rdf:about="mascular">
      <rdfs:subClassOf rdf:resource="Condition"/>
    </owl:Class>
    <owl:Class rdf:about="medication">
      <rdfs:subClassOf rdf:resource="Treatment"/>
    </owl:Class>
    <owl:Class rdf:about="medicine">
      <rdfs:subClassOf rdf:resource="medication"/>
    </owl:Class>
    <owl:Class rdf:about="metabolism">
      <rdfs:subClassOf rdf:resource="Condition"/>
    </owl:Class>
    <owl:Class rdf:about="nervous">
      <rdfs:subClassOf rdf:resource="Condition"/>
    </owl:Class>
    <owl:Class rdf:about="Nurse">
```

```
276         <rdfs:subClassOf rdf:resource="Person"/>
277       </owl:Class>
278       <owl:ObjectProperty rdf:about="occursAt">
279         <rdfs:domain>
280           <owl:Class>
281             <owl:unionOf rdf:parseType="Collection">
282               <owl:Class rdf:about="Activity"/>
283               <owl:Class rdf:about="Event"/>
284             </owl:unionOf>
285           </owl:Class>
286         </rdfs:domain>
287         <rdfs:range rdf:resource="Location"/>
288       </owl:ObjectProperty>
289       <owl:Class rdf:about="Owner">
290         <rdfs:subClassOf rdf:resource="Person"/>
291       </owl:Class>
292       <owl:ObjectProperty rdf:about="owns">
293         <rdfs:domain rdf:resource="Manufacturer"/>
294         <rdfs:range rdf:resource="Item"/>
295       </owl:ObjectProperty>
296       <owl:DatatypeProperty rdf:about="permElse">
297         <rdfs:domain rdf:resource="Document"/>
298       </owl:DatatypeProperty>
299       <owl:Class rdf:about="Permission"/>
300       <owl:DatatypeProperty rdf:about="permKnows">
301         <rdfs:domain rdf:resource="Document"/>
302       </owl:DatatypeProperty>
303       <owl:Class rdf:about="Person"/>
304       <owl:Class rdf:about="Poem">
305         <rdfs:subClassOf rdf:resource="Folklore"/>
306       </owl:Class>
307       <owl:Class rdf:about="Product">
308         <rdfs:subClassOf rdf:resource="Item"/>
309       </owl:Class>
310       <owl:Class rdf:about="Proverb">
311         <rdfs:subClassOf rdf:resource="Folklore"/>
312       </owl:Class>
313       <owl:Class rdf:about="psychological">
314         <rdfs:subClassOf rdf:resource="Condition"/>
315       </owl:Class>
316       <owl:Class rdf:about="region">
317         <rdfs:subClassOf rdf:resource="Geographical"/>
318       </owl:Class>
319       <owl:Class rdf:about="Relevance">
320         <rdfs:subClassOf rdf:resource="Weight"/>
321       </owl:Class>
322       <owl:Class rdf:about="Reliability">
323         <rdfs:subClassOf rdf:resource="Weight"/>
324       </owl:Class>
325       <owl:Class rdf:about="respiratory">
326         <rdfs:subClassOf rdf:resource="Condition"/>
327       </owl:Class>
328       <owl:Class rdf:about="Riddle">
329         <rdfs:subClassOf rdf:resource="Folklore"/>
330       </owl:Class>
```

```
    <owl:Class rdf:about="Ritual">
      <rdfs:subClassOf rdf:resource="Activity"/>
    </owl:Class>
    <owl:Class rdf:about="Sangoma">
      <rdfs:subClassOf rdf:resource="Person"/>
    </owl:Class>
    <owl:Class rdf:about="school">
      <rdfs:subClassOf rdf:resource="Institution"/>
    </owl:Class>
    <owl:Class rdf:about="Seller">
      <rdfs:subClassOf rdf:resource="Person"/>
    </owl:Class>
    <owl:ObjectProperty rdf:about="sells">
      <rdfs:domain rdf:resource="Manufacturer"/>
      <rdfs:range rdf:resource="Product"/>
      <rdfs:range rdf:resource="Service"/>
    </owl:ObjectProperty>
    <owl:Class rdf:about="sensory">
      <rdfs:subClassOf rdf:resource="Condition"/>
    </owl:Class>
    <owl:Class rdf:about="Service">
      <rdfs:subClassOf rdf:resource="Item"/>
    </owl:Class>
    <owl:Class rdf:about="shop">
      <rdfs:subClassOf rdf:resource="Institution"/>
    </owl:Class>
    <owl:ObjectProperty rdf:about="soldAt">
      <rdfs:domain rdf:resource="Product"/>
    </owl:ObjectProperty>
    <owl:ObjectProperty rdf:about="soldBy">
      <rdfs:domain rdf:resource="Product"/>
      <rdfs:range rdf:resource="Manufacturer"/>
      <rdfs:range rdf:resource="Owner"/>
      <rdfs:range rdf:resource="Seller"/>
    </owl:ObjectProperty>
    <owl:DatatypeProperty rdf:about="subject"/>
    <owl:Class rdf:about="surgical">
      <rdfs:subClassOf rdf:resource="Treatment"/>
    </owl:Class>
    <owl:Class rdf:about="Tag"/>
    <owl:ObjectProperty rdf:about="taggedBy">
      <rdfs:domain rdf:resource="Tagging"/>
      <rdfs:range rdf:resource="Person"/>
    </owl:ObjectProperty>
    <owl:ObjectProperty rdf:about="taggedItem">
      <rdfs:domain rdf:resource="Tagging"/>
      <rdfs:range rdf:resource="Item"/>
    </owl:ObjectProperty>
    <owl:Class rdf:about="Tagging"/>
    <owl:Class rdf:about="Text">
      <rdfs:subClassOf rdf:resource="Document"/>
    </owl:Class>
    <owl:Class rdf:about="tour">
      <rdfs:subClassOf rdf:resource="Service"/>
    </owl:Class>
```

```
386      <owl:Class rdf:about="Treatment">
387         <rdfs:subClassOf rdf:resource="Item"/>
388      </owl:Class>
389      <owl:Class rdf:about="Video">
390         <rdfs:subClassOf rdf:resource="Document"/>
391      </owl:Class>
392      <owl:Class rdf:about="village">
393         <rdfs:subClassOf rdf:resource="Geographical"/>
394      </owl:Class>
395      <owl:Class rdf:about="Weight"/>
396   </rdf:RDF>
```

Appendix B

KnowNet message tags

The KnowNet platform utilizes message tags to associate semantics to the content of the messages communicated between the agents, instead of using a defined vocabulary and ontology. These tags are utilized for specifying parameters within a conversation, and platform specific components encapsulated within a message.

B.1 Communication tags

The communication language tags within the platform facilitate the communication between agents in a standardized manner. The use of these tags in KnowNet is discussed in Section 6.7.1.

- <*piask:onto*> - This tag is used to specify the ontology associated with the message that is being communicated. For example, when the KBA agent sends a message to the KBONTA agent, it specifies the ontology associated with the request by including as the value of this parameter. Therefore in Figure 6.33, the request is associated with the ontology named *dwesa* (<*piask:onto*>*[dwesa]*).

- <*piask:subj*> - This tag specifies the subject referred to in the encapsulated message. In Figure 6.33, the subject in the *dwesa* ontology is *item* (<*piask:subj*>*[item]*).

- <*piask:pred*> - This is the tag for specifying the predicate associated with the message and the user request. For example, in a case of a user request for *"the name of the owner of the Resting Place BnB"*, this tag would carry the value hasOwner (<*piask:pred*>*[hasOwner]*) within the message passed to the relevant ontology.

- <*piask:func*> - For the cases where a specific function associated with an agent is required, this tag is utilized. In the communication between the MMIA agent and the SNA agent, this tag can be used with values *set* or *get* (<*piask:func*>*[set]* or <*piask:func*>*[get]*) to specify whether the SNA agent should update or return the information associated with a subject respectively. In Figure 6.33, the KBA agent is instructing the KBONTA agent to list (<*piask:func*>*[list]*) the triples in the associated ontology.

- <*piask:var*> - This is used for the specification of a variable that is communicated between the agents. For example, the session ID is passed between the access layer agents and the MMIA agent through this tag.

- <*piask:mime*> - In order to specify to the presentation agents the requested MIME type for content rendering, the request agent (typically access layer agents) would encapsulate the type within this tag. The actual value of this parameter is specified through any of the presentation tags (Section B.2).

- <*piask:result*> - This tag is used to specify the result of an operation that was undertaken by one agent on behalf of another. For example, the SNA agent would use this tag to set the response of an authentication request from the MMIA agent.

- <*piask:string*> - This tag is used to specify UI strings for the platform. The MPA agent then scans the received content for these tags in order to undertake the necessary linguistic translation of the strings.

B.2 Presentation tags

The following tags are utilized within KnowNet to specify to the Media Presentation Agent (MPA) the requested presentation format of the encapsulated content.

- <*piask:html*> - The content should be formatted in standard HTML. This is predominantly for web browser requests.

- <*piask:vxml*> - This specifies that content should be formatted as VXML, for the VXML browser integrated in Asterisks PBX.

- <*piask:festival*> - The content in this case is formatted for rendering by the Festival Text To Speech (TTS) engine. In essence, the presentation agent strips off all the mark-up language and outputs basic content strings.

- <*piask:im*> - The Instant Messaging (IM) format is similar to the <*piask:festival*> format, only with the usage of a few mark-up constructs that are appropriate for IM client (e.g. bold, underline).
- <*piask:email*> - This allows the content in a format appropriate for email clients, typically plain text rendering.
- <*piask:xml*> - In this format, the content is rendered as XML.

B.3 UI Components tags

Within the platform, the content is only formatted into specific mark-up language at the end of the request handling, by the presentation agent. Within the platform communication processes with the presentation agents, the following tags are used to specify the constructs that influence the UI rendering of the content.

- <*piask:br*> - This represents a break in the flow of the content. This tag is translated into a break (i.e. <*br*>) HMTL tag, a new line (i.e. *"\n"*) for IM, email and festival rendering
- <*piask:link*> - This represents a link to another content or function within the platform. This is translated into an anchor tag (i.e. <*a href="xxx" ... ></a*>) in HTML and processed as a submit block (i.e. <*submit next="xxx" ... /*>) in VXML. This tag is not rendered for IM clients.
- <*piask:outlink*> - This is for a link that is external to the platform. It is rendered following the same techniques as for the <*piask:link*> tag.
- <*piask:div*> - This is used for arranging content into logical groups. In HTML this translates to the <*div*> tag and in VXML it is translated into a <*block*> tag. For the plain text based rendering, (i.e. IM, festival, email) this is handled as a new content section/paragraph.
- <*piask:form*> - This represents an encapsulation of input content from the user. It is equivalent to the <form> tag in HTML and translates to a <*submit ... >* tag with the associated *namelist* parameters in VXML.
- <*piask:text*> - This tag is encapsulated within the <*piask:form*> tag to represent text input. It is translated to an HTML input tag (i.e. <*input type="text" name="xxx" ... >*) and a field tag in VXML (i.e. <*field name="xxx" ... >*).

- <*piask:story*> - Represents a longer occurrence of the <*piask:text*> tag, with another difference that it is handled as a <*textarea*> in HTML.

- <*piask:textupd*> - This provides the similar functionality to the <*piask:text*> tag, with the only difference that the field already contains a value to be updated or modified.

- <*piask:choice*> - This allows the user a selection of a choice among a number of options.

- <*piask:hidden*> - This is used within the <*piask:form*> tag to pass parameters back to the platform. In HTML, this is equivalent to the <*input type="hidden" ...* > tag and in VXML it is handled as an assigned variable that is passed within the namelist of the submit tag (i.e. <*assign name="xyz" expr="xxx" /*> <*submit namelist="xyz" next="yyy" ... /*>).

- <*piask:image*> - This is used to refer to an image. For HTML, the image is rendered (i.e. <*img src="xxx" ...*>) and for VXML and other plain text based clients, the textual description of the image is rendered.

- <*piask:password*> - This is handled similarly to the <*piask:text*> tag. The rendering of this control hides the content that the user enters using the HTML password input (i.e. <*input type="password" name="xxx" ...*>).

- <*piask:upload*> - This is used to upload files and content onto the platform. Within HTML, this is equivalent to the file input construct (i.e. <*input type="file" ...*>) and with VXML it is handled as a record tag (i.e. <*record*>)

Appendix C

KnowNet Interaction Contexts

The following are the interaction contexts that are defined within the MMIA agent. These contexts are to be differentiated from the HTTPA agent contexts, in a sense that HTTPA agent contexts are handled by the web server in a manner that allows for filtering of requests based on the specified context (e.g. in the URL *http://dwesa.piask.com/pws/*, the */pws* is the request context, that is associated with a WS application handler within the HTTPA agent). The MMIA agent contexts on the other hand represent the logical and functional organization of the processes that are handled within the MMIA agent in providing the business logic for the knowledge platform. These contexts are introduced in Section 6.3.1.4.

Context Label	Context Value	Context Description
CONTEXT_LOGIN	login.piask	This is the context for handling login requests.
CONTEXT_LOGOFF	logoff.piask	This handles the logoff requests.
CONTEXT_UPLOAD	upload.piask	Uploading of files and recorded audio onto the platform.
CONTEXT_DELETE	delete.piask	Handles the deletions on the platform.
CONTEXT_DOWNLOAD	download.piask	Facilitates the downloading of files and content from the platform.
CONTEXT_UPDATE	update.piask	Handles the updating of current knowledge on the platform.
CONTEXT_KNOW	know.piask	Handles all the requests for knowledge on the platform.
CONTEXT_DO	do.piask	Handles the requests on the platform to perform platform related operations (e.g. sending a voice message to a user, establishing a connection between VOIP extensions).
CONTEXT_GET	get.piask	Context for the interaction of getting information from the users (e.g. the platform requesting user authentication via the VOIP device).
CONTEXT_EMAIL	email.piask	Handles the email requests on the platform.
CONTEXT_PERSON	person.piask	A context handler for request associated with user profiles.
CONTEXT_REGISTER	register.piask	Provides the logic for handling new user registrations on the platform.
CONTEXT_VOTE	vote.piask	This is the context that supports the implementation of mechanisms to vote for different content, as far as reliability, relevance and accuracy are concerned.
CONTEXT_PERM	perm.piask	This context comes into play in setting access permissions for different content.
CONTEXT_KNOWREQ	knowreq.piask	Handles the requests to establish friend requests between platform users.
CONTEXT_SEARCH	search.piask	Handler for search requests on the platform.
CONTEXT_NEW	new.piask	This is the context for new content and new knowledge.
CONTEXT_FILE	file.piask	This is an internal context for accessing and handling platform specific file requests (e.g. UI background images, and CSS files).

Table C.1: Platform Interaction Contexts

Appendix D

Platform WS Ontology

```java
package com.dwesa.piask.ontology;

import jade.content.onto.BasicOntology;
import jade.content.onto.Ontology;
import jade.content.onto.OntologyException;
import jade.content.schema.AgentActionSchema;
import jade.content.schema.ConceptSchema;
import jade.content.schema.PrimitiveSchema;

public class PiaskOntology extends Ontology implements PiaskVocabulary {
	/**
	 *  by Mamello Thinyane for PIASK Platform v0.2
	 *  The ontology that exposes the minimal behaviours defined in the agents
	 */
	private static final long serialVersionUID = 1L;
	private static Ontology theInstance = new PiaskOntology();
	private static final String ONT_NAME = "PiaskOntology";

	public static Ontology getInstance() {
		return theInstance;
    }

	public PiaskOntology(){
		super(ONT_NAME, BasicOntology.getInstance());

		try{
			AgentActionSchema as = null;
			ConceptSchema cs = null;

			add(new ConceptSchema(ANSWER), Answer.class);
			add(new ConceptSchema(FRIENDS), Friends.class);

			/**
			 * adding the schema classes
			 */
			add(new AgentActionSchema(LOGIN), Login.class);
			add(new AgentActionSchema(GETFRIENDS), GetFriends.class);
			add(new AgentActionSchema(ADDFRIEND), AddFriend.class);
			add(new AgentActionSchema(REGISTER), Register.class);
			add(new AgentActionSchema(GETPROFILE), GetProfile.class);
			add(new AgentActionSchema(SEARCH), Search.class);
			add(new AgentActionSchema(GETCONTENT), GetContent.class);
			add(new AgentActionSchema(GETFILE), GetFile.class);
			add(new AgentActionSchema(SENDMESSAGE), SendMessage.class);
			add(new AgentActionSchema(CONNECT), Connect.class);


			/**
			 * these are all the concepts
			 */
			cs = (ConceptSchema)getSchema(ANSWER);
			cs.add(ANSWER_CONTENT, (PrimitiveSchema)getSchema(BasicOntology.ST...

			cs = (ConceptSchema)getSchema(FRIENDS);
			cs.add(FRIENDS_LIST, (PrimitiveSchema)getSchema(BasicOntology.STRING...
```

```
56
57
58                  /**
59                   * these are all the agent actions
60                   */
61                  as = (AgentActionSchema)getSchema(LOGIN);
62                  as.add(LOGIN_ID, (PrimitiveSchema)getSchema(BasicOntology.STRING));
63                  as.add(LOGIN_PASS, (PrimitiveSchema)getSchema(BasicOntology.STRING));
64                  as.add(LOGIN_USERNAME, (PrimitiveSchema)getSchema(BasicOntology.ST...
65                  as.setResult((ConceptSchema)getSchema(ANSWER));
66
67                  as = (AgentActionSchema)getSchema(GETFRIENDS);
68                  as.add(GETFRIENDS_ID, (PrimitiveSchema)getSchema(BasicOntology.STRIN...
69                  as.add(GETFRIENDS_USERNAME, (PrimitiveSchema)getSchema(BasicOntol...
70                  as.setResult((ConceptSchema)getSchema(FRIENDS));
71
72                  as = (AgentActionSchema)getSchema(ADDFRIEND);
73                  as.add(ADDFRIEND_ID, (PrimitiveSchema)getSchema(BasicOntology.STRIN...
74                  as.add(ADDFRIEND_FRIEND, (PrimitiveSchema)getSchema(BasicOntology.S...
75                  as.add(ADDFRIEND_USERNAME, (PrimitiveSchema)getSchema(BasicOntolo...
76                  as.setResult((ConceptSchema)getSchema(ANSWER));
77
78                  as = (AgentActionSchema)getSchema(REGISTER);
79                  as.add(REGISTER_NAME, (PrimitiveSchema)getSchema(BasicOntology.STRI...
80                  as.add(REGISTER_PASS, (PrimitiveSchema)getSchema(BasicOntology.STRI...
81                  as.add(REGISTER_USERNAME, (PrimitiveSchema)getSchema(BasicOntology...
82                  as.add(REGISTER_EMAIL, (PrimitiveSchema)getSchema(BasicOntology.STRI...
83                  as.add(REGISTER_PHONE, (PrimitiveSchema)getSchema(BasicOntology.STR...
84                  as.add(REGISTER_IM, (PrimitiveSchema)getSchema(BasicOntology.STRING));
85                  as.setResult((ConceptSchema)getSchema(ANSWER));
86
87                  as = (AgentActionSchema)getSchema(GETPROFILE);
88                  as.add(GETPROFILE_ID, (PrimitiveSchema)getSchema(BasicOntology.STRIN...
89                  as.add(GETPROFILE_USERNAME, (PrimitiveSchema)getSchema(BasicOntolo...
90
91                  as = (AgentActionSchema)getSchema(SEARCH);
92                  as.add(SEARCH_ID, (PrimitiveSchema)getSchema(BasicOntology.STRING));
93                  as.add(SEARCH_WORD, (PrimitiveSchema)getSchema(BasicOntology.STRIN...
94                  as.setResult((PrimitiveSchema)getSchema(BasicOntology.STRING));
95
96                  as = (AgentActionSchema)getSchema(GETCONTENT);
97                  as.add(GETCONTENT_ID, (PrimitiveSchema)getSchema(BasicOntology.STRI...
98                  as.add(GETCONTENT_ONTOLOGY, (PrimitiveSchema)getSchema(BasicOntolo...
99                  as.add(GETCONTENT_SUBJECT, (PrimitiveSchema)getSchema(BasicOntolog...
100                 as.add(GETCONTENT_PREDICATE, (PrimitiveSchema)getSchema(BasicOntol...
101
102                 as = (AgentActionSchema)getSchema(GETFILE);
103                 as.add(GETFILE_ID, (PrimitiveSchema)getSchema(BasicOntology.STRING));
104                 as.add(GETFILE_METHOD, (PrimitiveSchema)getSchema(BasicOntology.STR...
105
106                 as = (AgentActionSchema)getSchema(SENDMESSAGE);
107                 as.add(SENDMESSAGE_ID, (PrimitiveSchema)getSchema(BasicOntology.ST...
108                 as.add(SENDMESSAGE_RECEIVER, (PrimitiveSchema)getSchema(BasicOnto...
109                 as.add(SENDMESSAGE_CONTENT, (PrimitiveSchema)getSchema(BasicOnto...
110                 as.add(SENDMESSAGE_SENDER, (PrimitiveSchema)getSchema(BasicOntol...
```

```
111                 as.add(SENDMESSAGE_METHOD, (PrimitiveSchema)getSchema(BasicOntol...
112                 as.setResult((ConceptSchema)getSchema(ANSWER));
113
114                 as = (AgentActionSchema)getSchema(CONNECT);
115                 as.add(CONNECT_ID, (PrimitiveSchema)getSchema(BasicOntology.STRING));
116                 as.add(CONNECT_USER, (PrimitiveSchema)getSchema(BasicOntology.STRI...
117                 as.add(CONNECT_METHOD, (PrimitiveSchema)getSchema(BasicOntology.ST...
118                 as.setResult((ConceptSchema)getSchema(ANSWER));
119
120             }catch(OntologyException e){
121                 e.printStackTrace();
122             }
123         }
124
125     }
```

Appendix E

PWSA WSDL

The following is a WSDL of the services exposed for the purposes of implementing the J2ME mobile interface. The WSDL shows the basic services that are made available through the PIASK Web Services Agent (PWSA) as defined in the agent's ontology and the vocabulary (Section 6.4.6).

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="PIASK" targetNamespace="urn:PIASK" xmlns:impl="urn:PIASK" ...
 <wsdl:types>
  <xsd:schema targetNamespace="urn:PIASK" xmlns:impl="urn:PIASK" xmlns:xsd="http:/...
    <xsd:element name="getfriends">
        <xsd:complexType><xsd:sequence><xsd:element name="id" type="xsd:string"/>
        <xsd:element name="username" type="xsd:string"/></xsd:sequence></xsd:co...
    </xsd:element>
    <xsd:element name="getfriendsResponse">
        <xsd:complexType><xsd:sequence><xsd:element name="getfriendsReturn" typ...
    </xsd:element>
    <xsd:complexType name="FRIENDS">
        <xsd:sequence><xsd:element name="list" type="xsd:string"/></xsd:sequence>
    </xsd:complexType>
    <xsd:element name="getprofile">
        <xsd:complexType><xsd:sequence><xsd:element name="id" type="xsd:string"/...
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="getprofileResponse">
        <xsd:complexType><xsd:sequence/></xsd:complexType>
    </xsd:element>
    <xsd:element name="addfriend">
        <xsd:complexType><xsd:sequence><xsd:element name="id" type="xsd:string"/...
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="addfriendResponse"><xsd:complexType><xsd:sequence><xsd...
 </wsdl:types>
 <wsdl:message name="addfriendRequest">
  <wsdl:part name="parameters" element="impl:addfriend">
  </wsdl:part>
 </wsdl:message>
 <wsdl:message name="getprofileRequest">
  <wsdl:part name="parameters" element="impl:getprofile">
  </wsdl:part>
 </wsdl:message>
 <wsdl:message name="getprofileResponse">
  <wsdl:part name="parameters" element="impl:getprofileResponse">
  </wsdl:part>
 </wsdl:message>
 <wsdl:message name="getfileRequest">
  <wsdl:part name="parameters" element="impl:getfile">
  </wsdl:part>
 </wsdl:message>
 <wsdl:message name="addfriendResponse">
  <wsdl:part name="parameters" element="impl:addfriendResponse">
  </wsdl:part>
 </wsdl:message>
 <wsdl:message name="searchRequest">
  <wsdl:part name="parameters" element="impl:search">
  </wsdl:part>
 </wsdl:message>
 <wsdl:message name="getcontentRequest">
  <wsdl:part name="parameters" element="impl:getcontent">
  </wsdl:part>
 </wsdl:message>
```

```
    <wsdl:message name="loginResponse">
      <wsdl:part name="parameters" element="impl:loginResponse">
      </wsdl:part>
    </wsdl:message>
    <wsdl:message name="searchResponse">
      <wsdl:part name="parameters" element="impl:searchResponse">
      </wsdl:part>
    </wsdl:message>
    <wsdl:message name="connectRequest">
      <wsdl:part name="parameters" element="impl:connect">
      </wsdl:part>
    </wsdl:message>
    <wsdl:message name="connectResponse">
      <wsdl:part name="parameters" element="impl:connectResponse">
      </wsdl:part>
    </wsdl:message>
    <wsdl:message name="sendmessageResponse">
      <wsdl:part name="parameters" element="impl:sendmessageResponse">
      </wsdl:part>
    </wsdl:message>
    <wsdl:message name="registerResponse">
      <wsdl:part name="parameters" element="impl:registerResponse">
      </wsdl:part>
    </wsdl:message>
    <wsdl:message name="getfileResponse">
      <wsdl:part name="parameters" element="impl:getfileResponse">
      </wsdl:part>
    </wsdl:message>
    <wsdl:message name="getfriendsResponse">
      <wsdl:part name="parameters" element="impl:getfriendsResponse">
      </wsdl:part>
    </wsdl:message>
    <wsdl:message name="registerRequest">
      <wsdl:part name="parameters" element="impl:register">
      </wsdl:part>
    </wsdl:message>
    <wsdl:message name="getfriendsRequest">
      <wsdl:part name="parameters" element="impl:getfriends">
      </wsdl:part>
    </wsdl:message>
    <wsdl:message name="loginRequest">
      <wsdl:part name="parameters" element="impl:login">
      </wsdl:part>
    </wsdl:message>
    <wsdl:message name="getcontentResponse">
      <wsdl:part name="parameters" element="impl:getcontentResponse">
      </wsdl:part>
    </wsdl:message>
    <wsdl:message name="sendmessageRequest">
      <wsdl:part name="parameters" element="impl:sendmessage">
      </wsdl:part>
    </wsdl:message>
    <wsdl:portType name="PIASKPort">
      <wsdl:operation name="getfriends">
        <wsdl:input message="impl:getfriendsRequest">
```

```
111     </wsdl:input>
112      <wsdl:output message="impl:getfriendsResponse">
113     </wsdl:output>
114     </wsdl:operation>
115     <wsdl:operation name="getprofile">
116      <wsdl:input message="impl:getprofileRequest">
117     </wsdl:input>
118      <wsdl:output message="impl:getprofileResponse">
119     </wsdl:output>
120     </wsdl:operation>
121     <wsdl:operation name="addfriend">
122      <wsdl:input message="impl:addfriendRequest">
123     </wsdl:input>
124      <wsdl:output message="impl:addfriendResponse">
125     </wsdl:output>
126     </wsdl:operation>
127     <wsdl:operation name="login">
128      <wsdl:input message="impl:loginRequest">
129     </wsdl:input>
130      <wsdl:output message="impl:loginResponse">
131     </wsdl:output>
132     </wsdl:operation>
133     <wsdl:operation name="getcontent">
134      <wsdl:input message="impl:getcontentRequest">
135     </wsdl:input>
136      <wsdl:output message="impl:getcontentResponse">
137     </wsdl:output>
138     </wsdl:operation>
139     <wsdl:operation name="register">
140      <wsdl:input message="impl:registerRequest">
141     </wsdl:input>
142      <wsdl:output message="impl:registerResponse">
143     </wsdl:output>
144     </wsdl:operation>
145     <wsdl:operation name="getfile">
146      <wsdl:input message="impl:getfileRequest">
147     </wsdl:input>
148      <wsdl:output message="impl:getfileResponse">
149     </wsdl:output>
150     </wsdl:operation>
151     <wsdl:operation name="connect">
152      <wsdl:input message="impl:connectRequest">
153     </wsdl:input>
154      <wsdl:output message="impl:connectResponse">
155     </wsdl:output>
156     </wsdl:operation>
157     <wsdl:operation name="sendmessage">
158      <wsdl:input message="impl:sendmessageRequest">
159     </wsdl:input>
160      <wsdl:output message="impl:sendmessageResponse">
161     </wsdl:output>
162     </wsdl:operation>
163     <wsdl:operation name="search">
164      <wsdl:input message="impl:searchRequest">
165     </wsdl:input>
```

```
166       <wsdl:output message="impl:searchResponse">
167       </wsdl:output>
168       </wsdl:operation>
169     </wsdl:portType>
170     <wsdl:binding name="PIASKBinding" type="impl:PIASKPort">
171       <wsdlsoap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http...
172       <wsdl:operation name="getfriends">
173         <wsdlsoap:operation soapAction="urn:PIASKAction"/>
174         <wsdl:input>
175           <wsdlsoap:body use="literal"/>
176         </wsdl:input>
177         <wsdl:output>
178           <wsdlsoap:body use="literal"/>
179         </wsdl:output>
180       </wsdl:operation>
181       <wsdl:operation name="getprofile">
182         <wsdlsoap:operation soapAction="urn:PIASKAction"/>
183         <wsdl:input>
184           <wsdlsoap:body use="literal"/>
185         </wsdl:input>
186         <wsdl:output>
187           <wsdlsoap:body use="literal"/>
188         </wsdl:output>
189       </wsdl:operation>
190       <wsdl:operation name="addfriend">
191         <wsdlsoap:operation soapAction="urn:PIASKAction"/>
192         <wsdl:input>
193           <wsdlsoap:body use="literal"/>
194         </wsdl:input>
195         <wsdl:output>
196           <wsdlsoap:body use="literal"/>
197         </wsdl:output>
198       </wsdl:operation>
199       <wsdl:operation name="login">
200         <wsdlsoap:operation soapAction="urn:PIASKAction"/>
201         <wsdl:input>
202           <wsdlsoap:body use="literal"/>
203         </wsdl:input>
204         <wsdl:output>
205           <wsdlsoap:body use="literal"/>
206         </wsdl:output>
207       </wsdl:operation>
208       <wsdl:operation name="getcontent">
209         <wsdlsoap:operation soapAction="urn:PIASKAction"/>
210         <wsdl:input>
211           <wsdlsoap:body use="literal"/>
212         </wsdl:input>
213         <wsdl:output>
214           <wsdlsoap:body use="literal"/>
215         </wsdl:output>
216       </wsdl:operation>
217       <wsdl:operation name="register">
218         <wsdlsoap:operation soapAction="urn:PIASKAction"/>
219         <wsdl:input>
220           <wsdlsoap:body use="literal"/>
```

```
      </wsdl:input>
      <wsdl:output>
        <wsdlsoap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="getfile">
      <wsdlsoap:operation soapAction="urn:PIASKAction"/>
      <wsdl:input>
        <wsdlsoap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <wsdlsoap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="connect">
      <wsdlsoap:operation soapAction="urn:PIASKAction"/>
      <wsdl:input>
        <wsdlsoap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <wsdlsoap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="sendmessage">
      <wsdlsoap:operation soapAction="urn:PIASKAction"/>
      <wsdl:input>
        <wsdlsoap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <wsdlsoap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="search">
      <wsdlsoap:operation soapAction="urn:PIASKAction"/>
      <wsdl:input>
        <wsdlsoap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <wsdlsoap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:service name="PIASKService">
    <wsdl:port name="PIASKPort" binding="impl:PIASKBinding">
      <wsdlsoap:address location="http://www.dwesa.com/pws/ws/PIASK"/>
    </wsdl:port>
  </wsdl:service>
</wsdl:definitions>
```

Appendix F

Platform Agents details

The following are the class diagrams of the different agents that have been implemented in the KnowNet platform. The details of the functionality provided by the agents are discussed in Section 6.

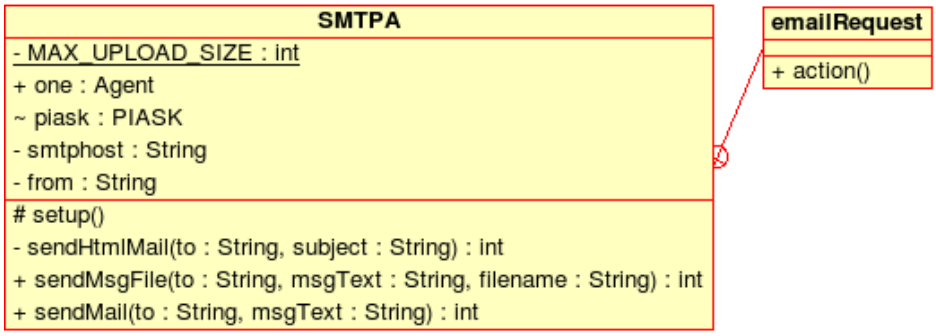


Figure F.1: SMTPA agent class diagram

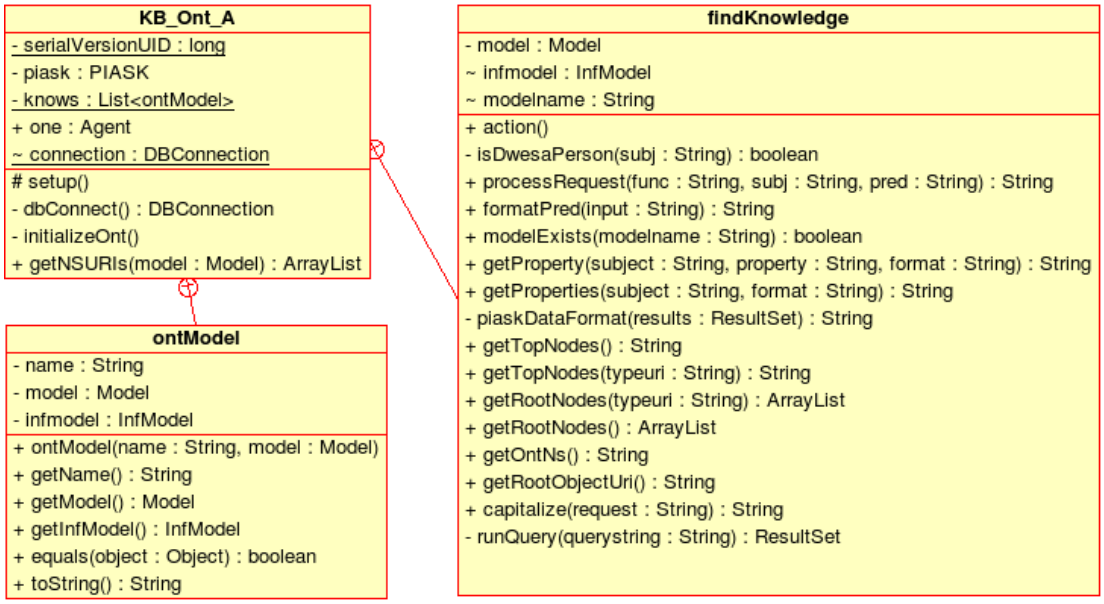


Figure F.2: KBONTA agent class diagram

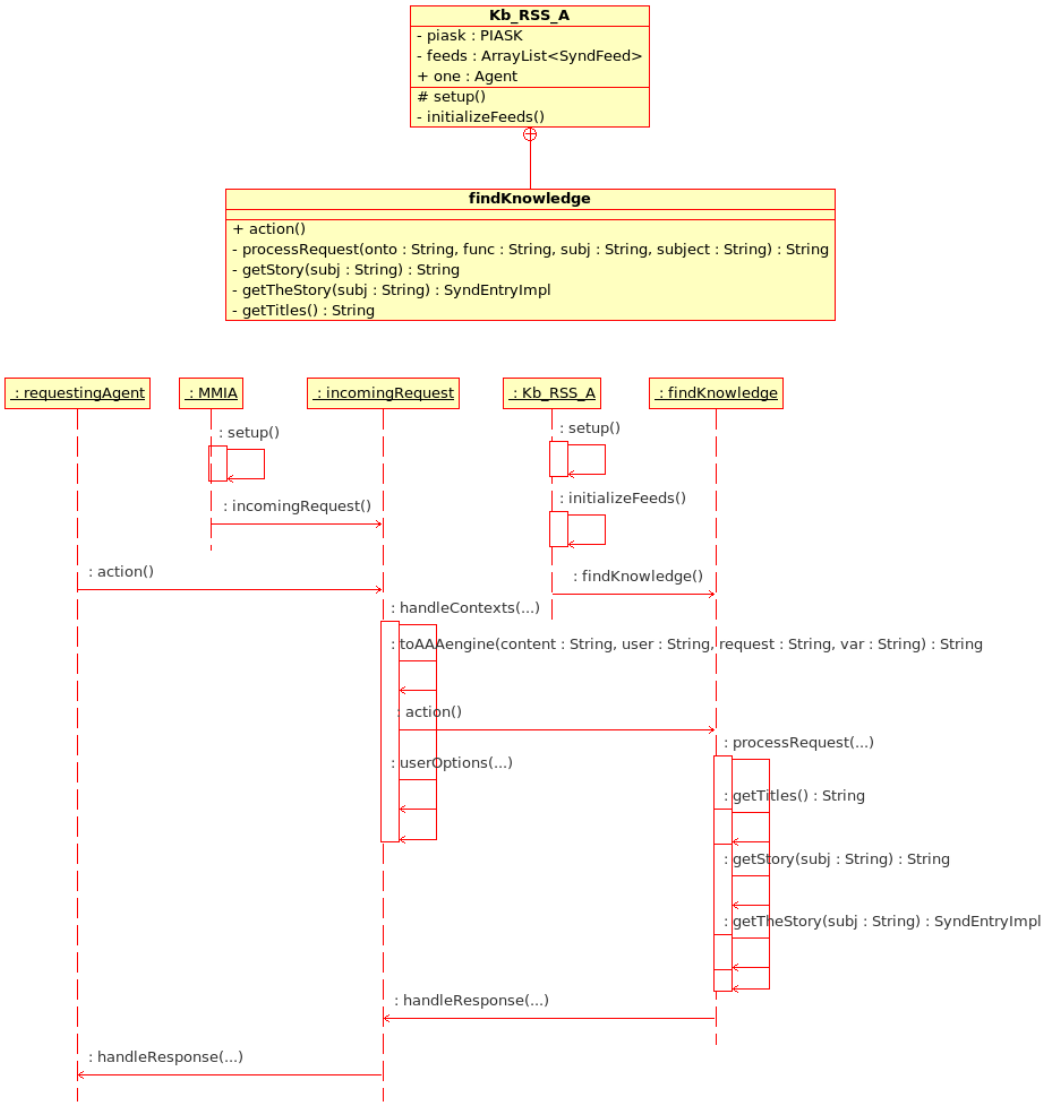


Figure F.3: RSS Agent class diagram

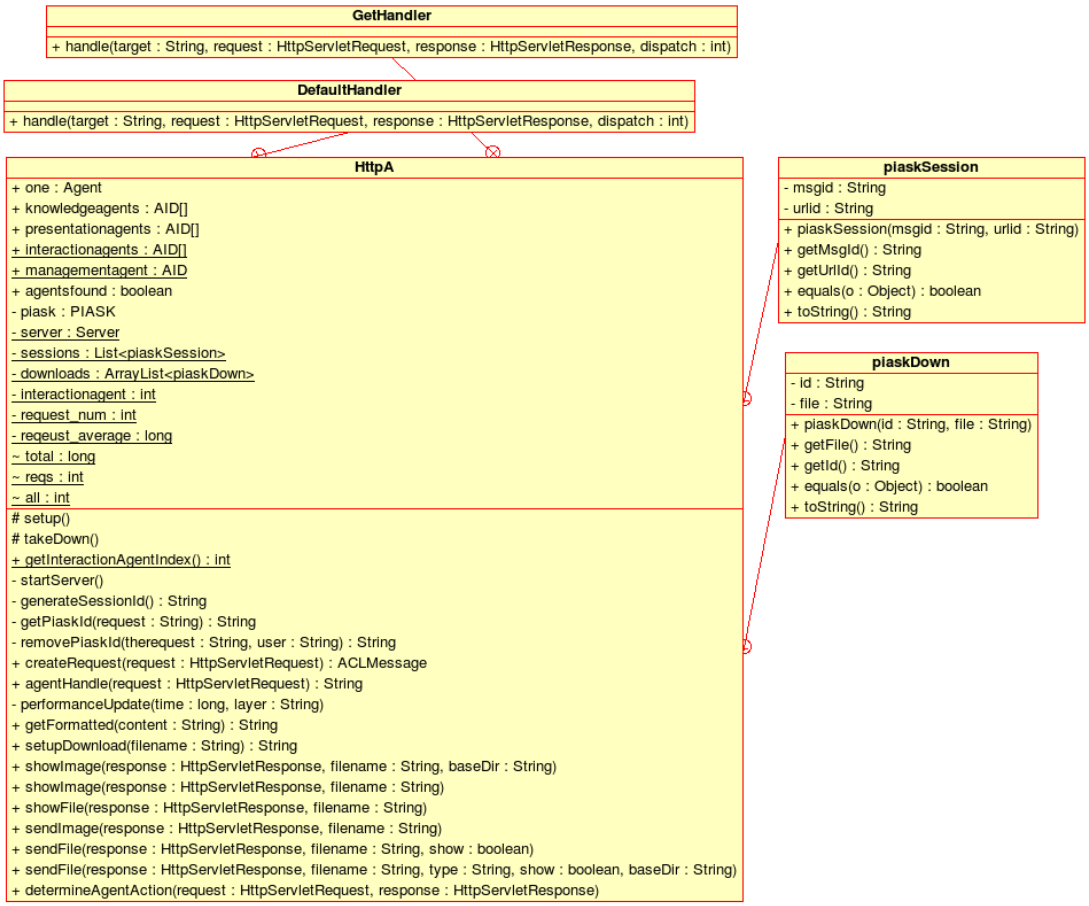


Figure F.4: HTTPA agent class diagram

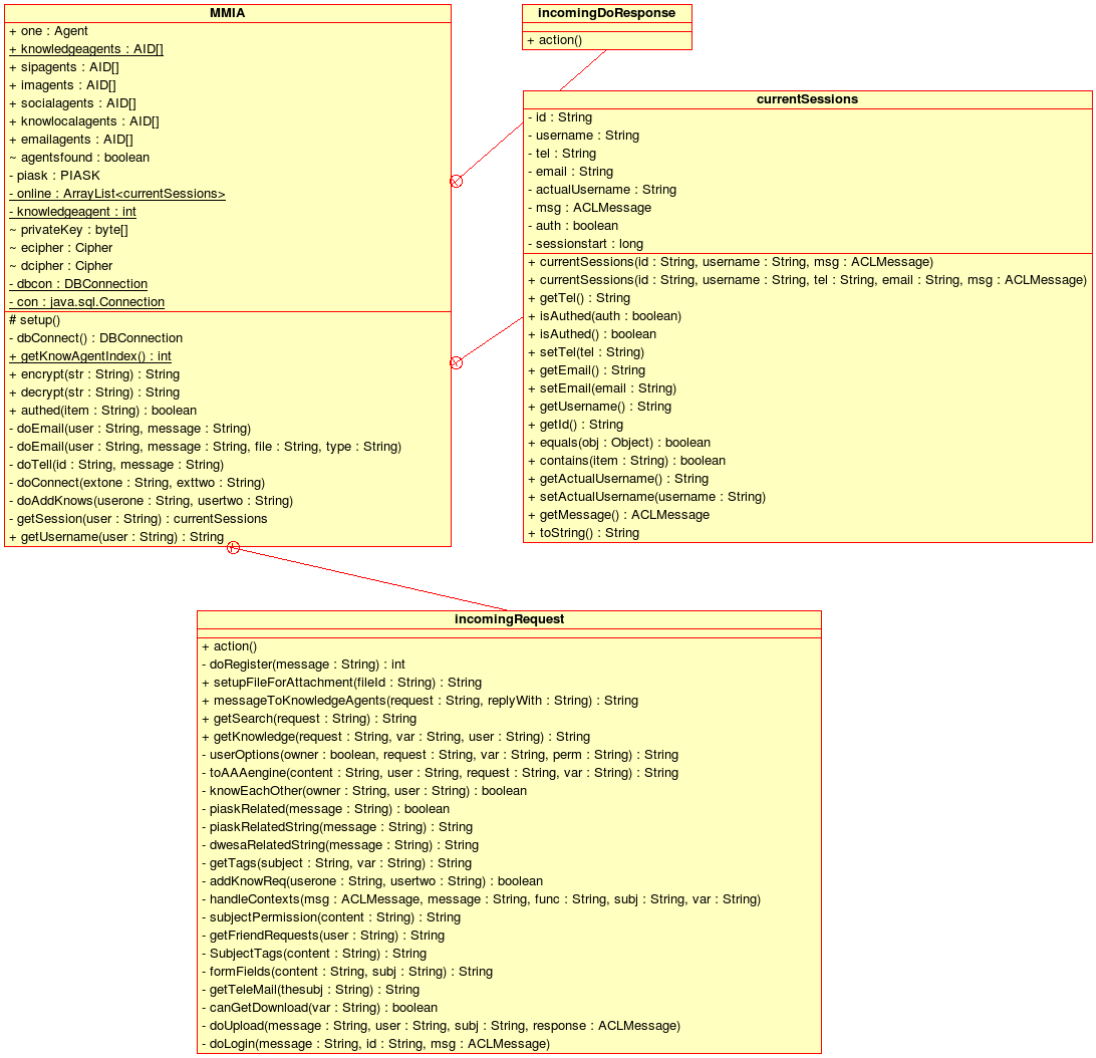


Figure F.5: MMIA agent class diagram

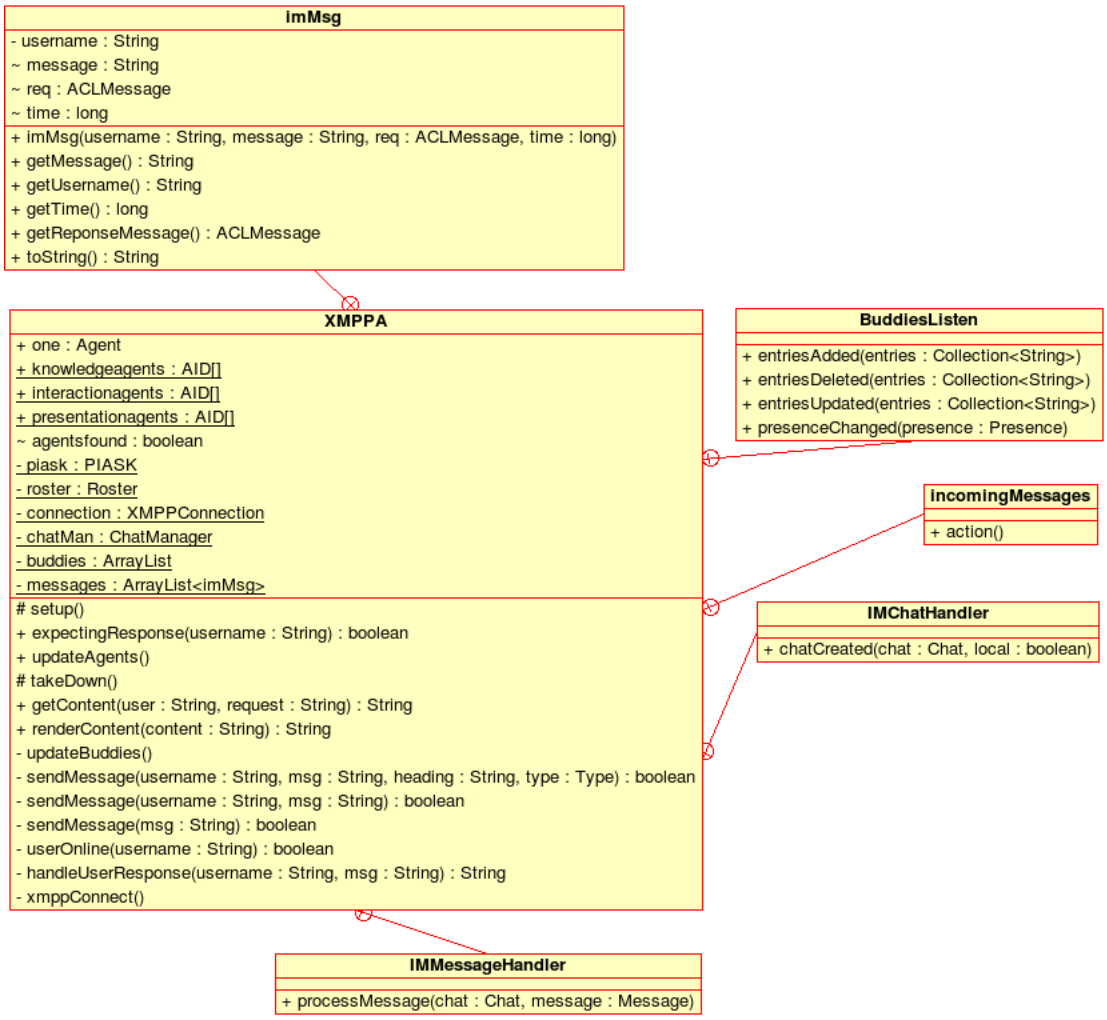


Figure F.6: XMPPA agent class diagram

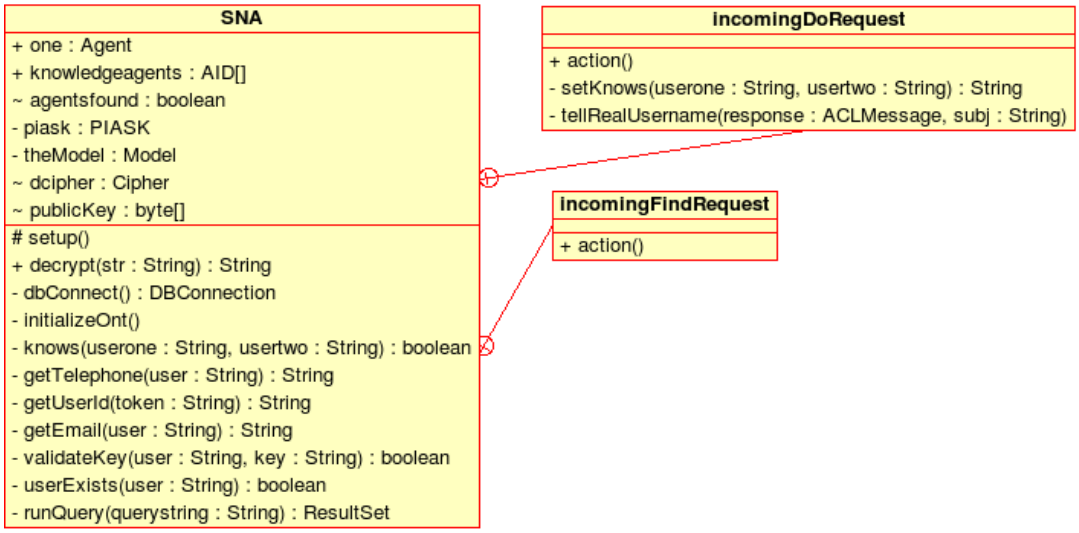


Figure F.7: SNA agent class diagram

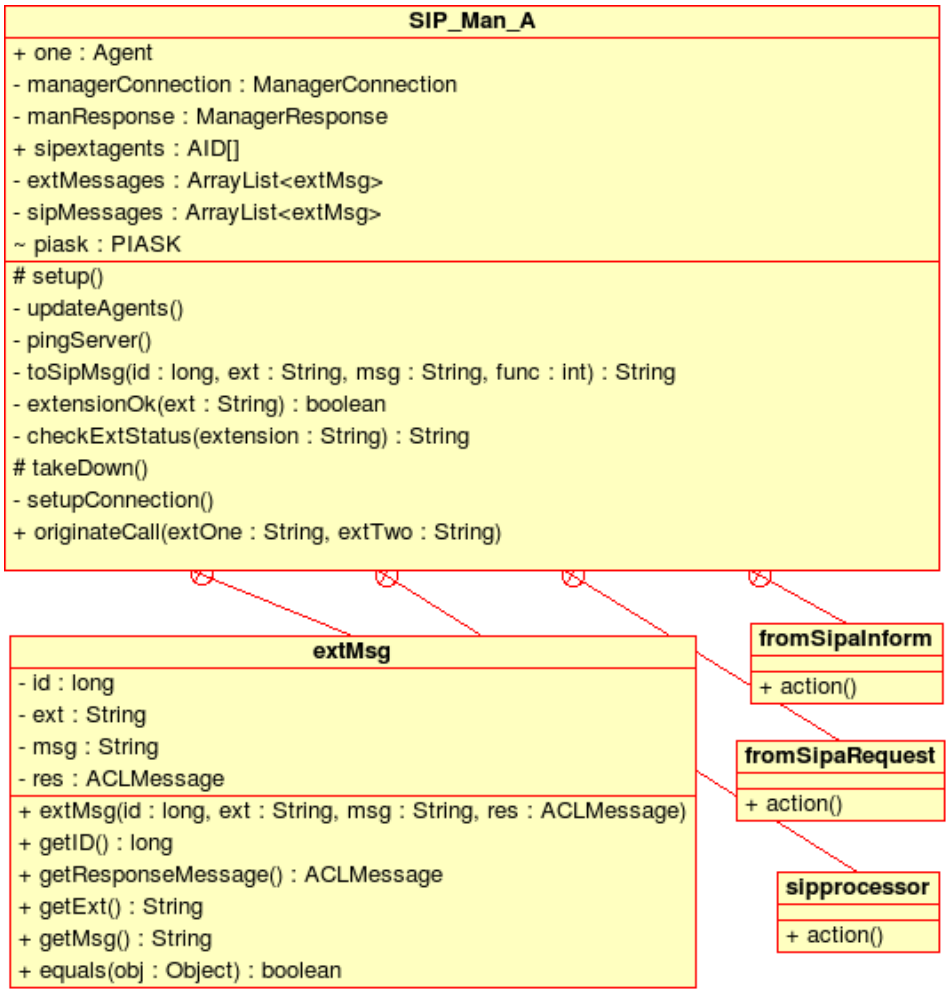


Figure F.8: SIPMA agent class diagram

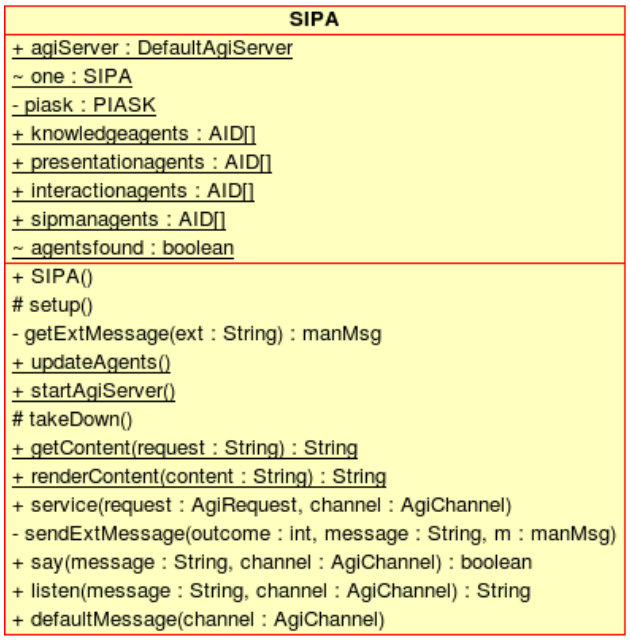


Figure F.9: SIPA agent class diagram

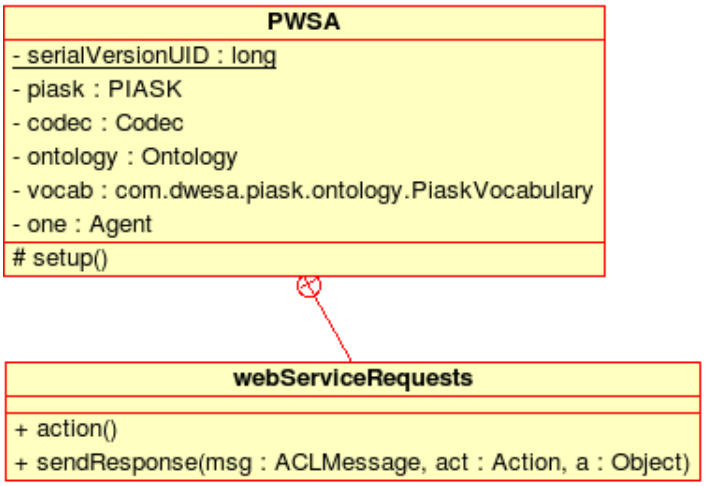


Figure F.10: PWSA agent class diagram

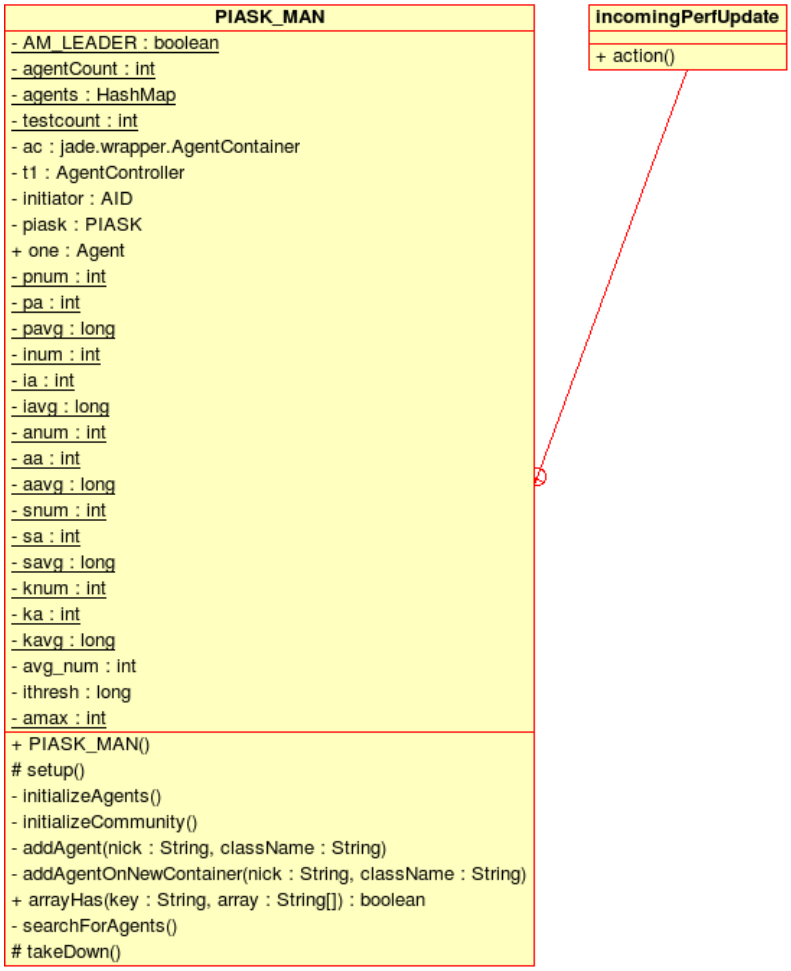


Figure F.11: PMANA agent class diagram

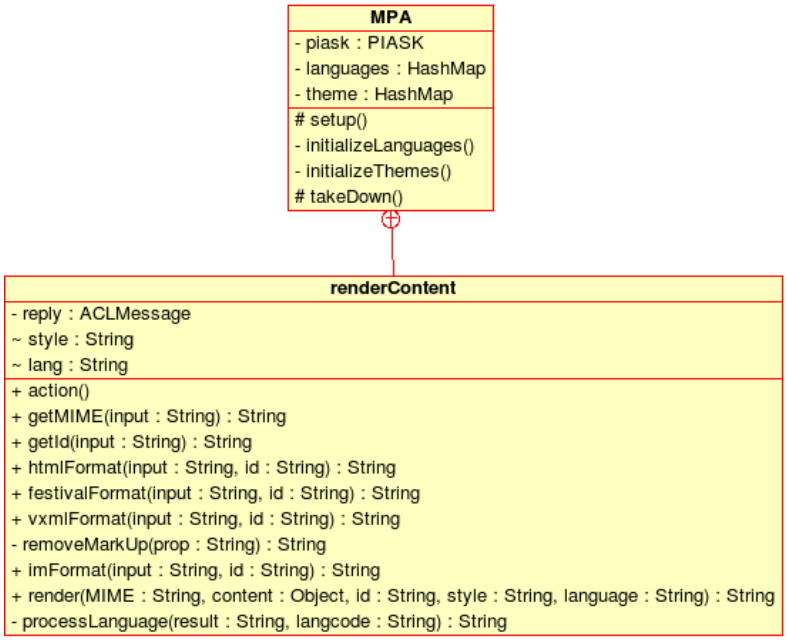


Figure F.12: MPA agent class diagram

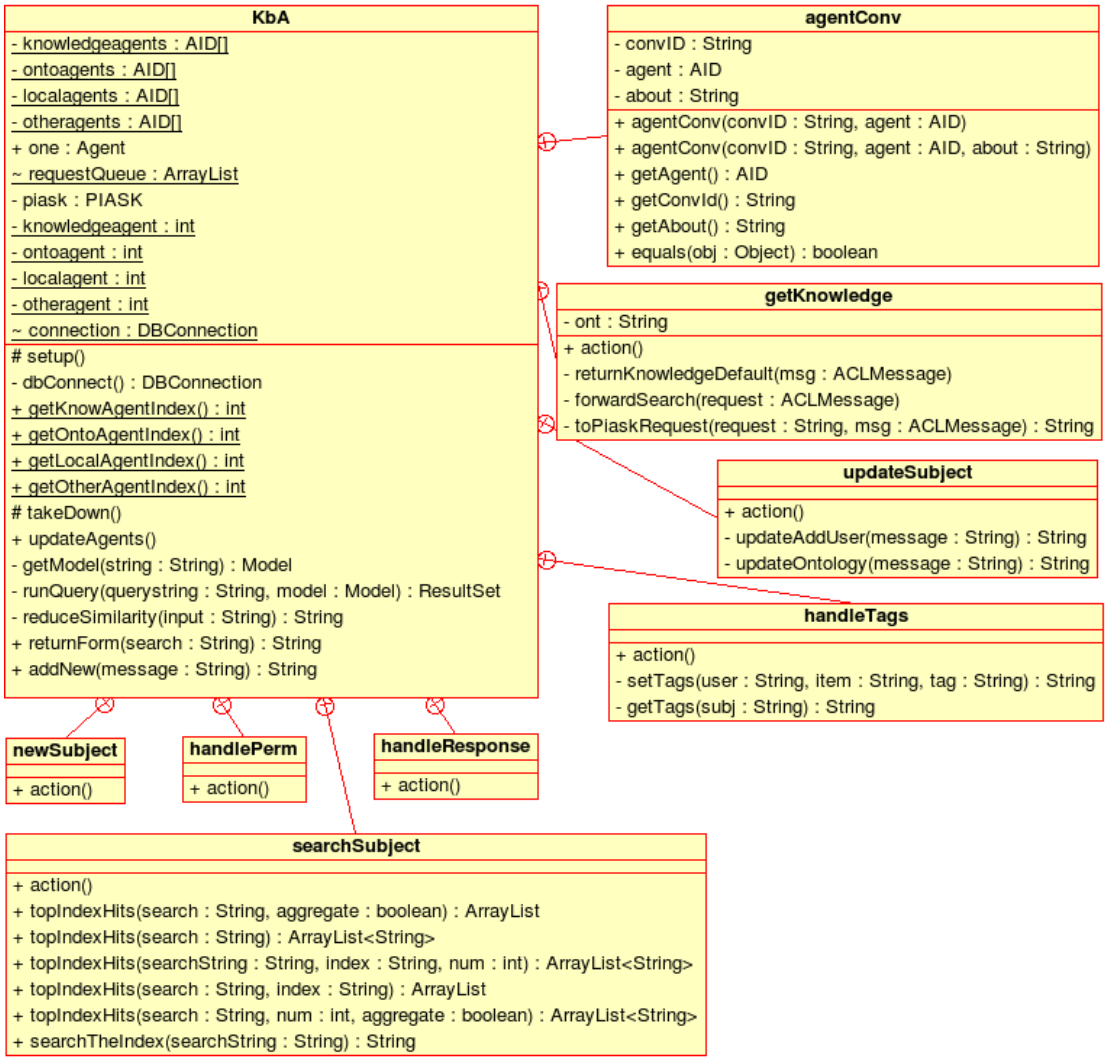


Figure F.13: KBA agent class diagram

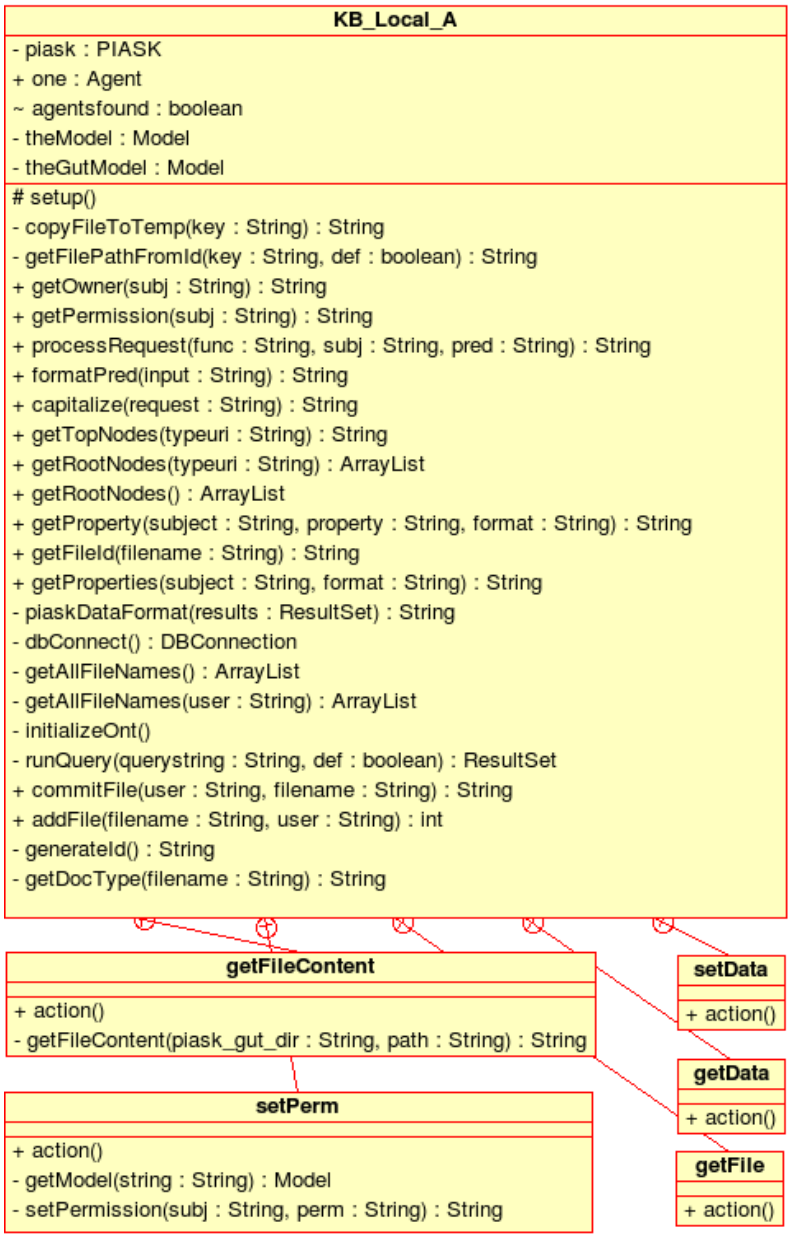


Figure F.14: KBLA agent class diagram

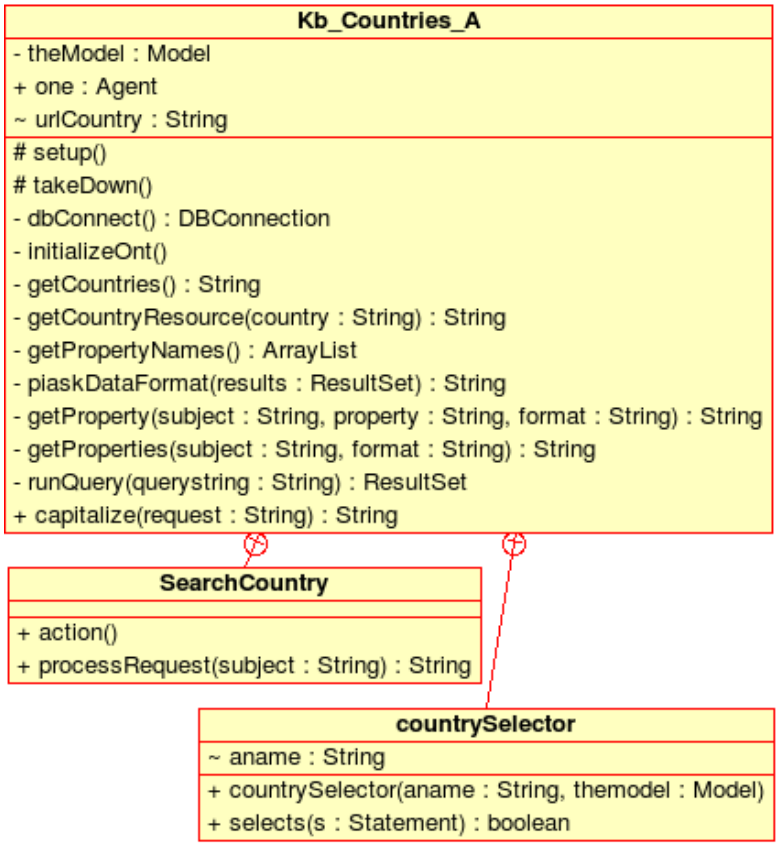


Figure F.15: KBCA agent class diagram

Appendix G

KnowNet Error recovery

The following KnowNet logs highlight the process of recovery from a Main Container crash of the platform. In this scenario, a replicated back-up Main Container is setup. When the Main Container is terminated on the platform, the backup container takes the role of leadership on the platform, and the peripheral containers associated with the new Main Container. This scenario illustrates the fault-tolerance and resilience of the PIASK architecture as initially discussed in Section 7.3.

```
[A] >>>>>>>>>>>>>>>>>>>> The Main Container <<<<<<<<<<<<<<<<<<<...


Agent container Main-Container@thepentagon is ready.
--------------------------------------------
2008/10/01 09:35:42 jade.core.PlatformManagerImpl localAddReplica
INFO: Adding replica <rmi://thebox:1099/> to the platform
2008/10/01 09:35:42 jade.core.PlatformManagerImpl localAddNode
INFO: Adding node <Main-Container-1> to the platform
2008/10/01 09:35:42 jade.core.messaging.MessagingService clearCachedSlice
INFO: Clearing cache
2008/10/01 09:35:42 jade.core.messaging.MessagingService clearCachedSlice
INFO: Clearing cache
2008/10/01 09:35:42 jade.core.replication.MainReplicationService$ServiceComponent ad...
INFO: Main container ring re-arranged: label = 0 monitored label = 1
2008/10/01 09:35:50 jade.core.PlatformManagerImpl localAddNode
INFO: Adding node <pentagon> to the platform
2008/10/01 09:35:50 jade.core.messaging.MessagingService clearCachedSlice
INFO: Clearing cache
2008/10/01 09:35:50 jade.core.PlatformManagerImpl$1 nodeAdded
INFO: --- Node <pentagon> ALIVE ---
2008/10/01 09:35:56 jade.core.PlatformManagerImpl localAddNode
INFO: Adding node <thebox> to the platform
2008/10/01 09:35:56 jade.core.messaging.MessagingService clearCachedSlice
INFO: Clearing cache
2008/10/01 09:35:56 jade.core.PlatformManagerImpl$1 nodeAdded
INFO: --- Node <thebox> ALIVE ---
2008/10/01 09:37:31 jade.core.nodeMonitoring.BlockingNodeFailureMonitor run
INFO: PING from node Main-Container-1 returned [GO ON]
2008/10/01 09:37:31 jade.core.Runtime endContainer
WARNING: Time-critical threads still active:
java.lang.ThreadGroup[name=JADE time-critical threads,maxpri=10]
2008/10/01 09:37:31 jade.core.Runtime$1 run
INFO: JADE is closing down now.



[B] >>>>>>>>>>>>>>>>>>> The Replicated Main Container <<<<<<<<<<<<<<<<<...


Agent container Main-Container-1@thebox is ready.
--------------------------------------------
Oct 1, 2008 9:33:39 AM jade.core.PlatformManagerImpl localAddNode
INFO: Adding node <pentagon> to the platform
Oct 1, 2008 9:33:45 AM jade.core.PlatformManagerImpl localAddNode
INFO: Adding node <thebox> to the platform
Oct 1, 2008 9:35:20 AM jade.core.nodeMonitoring.BlockingNodeFailureMonitor run
INFO: PING from node Main-Container returned [EXIT]
Oct 1, 2008 9:35:20 AM jade.core.PlatformManagerImpl localRemoveReplica
INFO: Removing replica <rmi://thepentagon.ict.ru.ac.za:1099/> from the platform
Oct 1, 2008 9:35:20 AM jade.core.PlatformManagerImpl removeTerminatedNode
INFO: --- Node <pentagon> TERMINATED ---
Oct 1, 2008 9:35:20 AM jade.core.PlatformManagerImpl localRemoveNode
INFO: Removing node <pentagon> from the platform
Oct 1, 2008 9:35:20 AM jade.core.PlatformManagerImpl adopt
```

```
INFO: Node <thebox> adopted
Oct 1, 2008 9:35:20 AM jade.core.PlatformManagerImpl$1 nodeAdded
INFO: --- Node <thebox> ALIVE ---
Oct 1, 2008 9:35:20 AM jade.core.PlatformManagerImpl localRemoveNode
INFO: Removing node <Main-Container> from the platform
Oct 1, 2008 9:35:20 AM jade.core.replication.MainReplicationService$ServiceComponent ...
INFO: Main container ring re-arranged: label = 0 monitored label = 0
Oct 1, 2008 9:35:20 AM jade.core.replication.MainReplicationService$ServiceComponent ...
INFO: -- I'm the new leader ---
Oct 1, 2008 9:35:20 AM jade.core.PlatformManagerImpl adopt
WARNING: NO descriptor found for node <pentagon> requesting adoption. Ignore...


[C] >>>>>>>>>>>>>>>>>>>> The Pentagon Peripheral Container <<<<<<<<<<...

INFO: Service jade.core.management.AgentManagement initialized
2008/10/01 09:35:50 jade.core.BaseService init
INFO: Service jade.core.messaging.Messaging initialized
2008/10/01 09:35:50 jade.core.BaseService init
INFO: Service jade.core.replication.AddressNotification initialized
2008/10/01 09:35:50 jade.core.BaseService init
INFO: Service jade.core.mobility.AgentMobility initialized
2008/10/01 09:35:50 jade.core.BaseService init
INFO: Service jade.core.event.Notification initialized
2008/10/01 09:35:50 jade.core.messaging.MessagingService clearCachedSlice
INFO: Clearing cache
2008/10/01 09:35:50 jade.core.ServiceManagerImpl addAddress
INFO: Adding PlatformManager address rmi://thebox:1099/
2008/10/01 09:35:50 jade.core.ServiceManagerImpl addAddress
INFO: Adding PlatformManager address rmi://thepentagon.ict.ru.ac.za:1099/
2008/10/01 09:35:50 jade.core.AgentContainerImpl joinPlatform
INFO: ----------------------------------------
Agent container pentagon@thepentagon is ready.
------------------------------------------------
2008-10-01 09:35:50.845::INFO:  Logging to STDERR via org.mortbay.log.StdErrLog
2008-10-01 09:35:50.886::INFO:  jetty-6.1.0
2008-10-01 09:35:50.966::INFO:  Started SocketConnector @ 0.0.0.0:8080
 INFO [sipma] (ManagerConnectionImpl.java:678) - Connecting to thebox.ict.ru.ac.za:5038
 INFO [sipa] (DefaultAgiServer.java:250) - Thread pool started.
 INFO [sipa] (DefaultAgiServer.java:262) - Listening on *:4573.
 INFO [Asterisk-Java ManagerConnection-0-Reader-0] (ManagerConnectionImpl.java:1230) -...
 INFO [sipma] (ManagerConnectionImpl.java:579) - Successfully logged in
 INFO [sipma] (ManagerConnectionImpl.java:584) - Determined Asterisk version: Asterisk 1.4
- - - - No Customisation or Mime Types available.
- - - - - - 2008/10/01 09:37:31 jade.core.messaging.MessagingService clearCachedSlice
INFO: Clearing cache
2008/10/01 09:37:31 jade.core.ServiceManagerImpl reconnect
INFO: Reconnecting to PlatformManager at address rmi://thebox:1099/
2008/10/01 09:37:31 jade.core.messaging.MessagingService clearCachedSlice
INFO: Clearing cache
2008/10/01 09:37:31 jade.core.ServiceManagerImpl reconnect
INFO: Reconnection OK

[D] >>>>>>>>>>>>>>>>>>>> The Box Peripheral Container <<<<<<<<<<<<...

```

```
111 INFO: Service jade.core.event.Notification initialized
112 Oct 1, 2008 9:33:45 AM jade.core.messaging.MessagingService clearCachedSlice
113 INFO: Clearing cache
114 Oct 1, 2008 9:33:45 AM jade.core.ServiceManagerImpl addAddress
115 INFO: Adding PlatformManager address rmi://thebox:1099/
116 Oct 1, 2008 9:33:45 AM jade.core.ServiceManagerImpl addAddress
117 INFO: Adding PlatformManager address rmi://thepentagon.ict.ru.ac.za:1099/
118 Oct 1, 2008 9:35:20 AM jade.core.ServiceManagerImpl platformManagerDead
119 INFO: PlatformManager at rmi://thepentagon.ict.ru.ac.za:1099/ no longer valid!
120 Oct 1, 2008 9:35:20 AM jade.core.ServiceManagerImpl addAddress
121 INFO: Adding PlatformManager address rmi://thebox:1099/
122 Oct 1, 2008 9:35:20 AM jade.core.ServiceManagerImpl removeAddress
123 INFO: Removing PlatformManager address rmi://thepentagon.ict.ru.ac.za:1099/
124 Oct 1, 2008 9:35:20 AM jade.core.ServiceManagerImpl reconnect
125 INFO: Reconnecting to PlatformManager at address rmi://thebox:1099/
126 Oct 1, 2008 9:35:20 AM jade.core.messaging.MessagingService clearCachedSlice
127 INFO: Clearing cache
128 Oct 1, 2008 9:35:20 AM jade.core.ServiceManagerImpl reconnect
129 INFO: Reconnection OK
130
```

Appendix H

Performance Profiling

Further observations that have been made as far as the performance of the PIASK platform under increasing usage loads is concerned are shown in the following graphs:

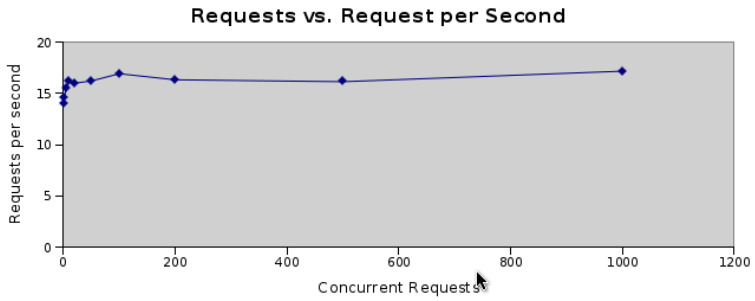


Figure H.1: KnowNet request per second

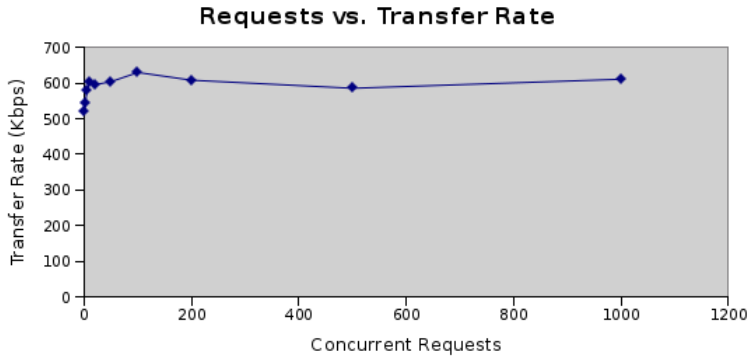


Figure H.2: KnowNet transfer rate

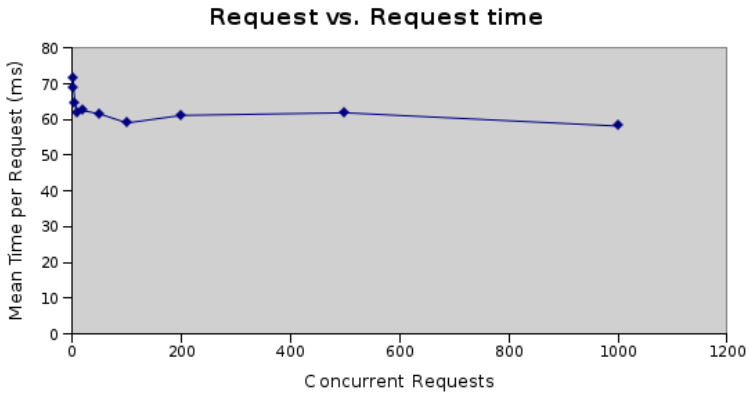


Figure H.3: KnowNet request time

Appendix I

Usability Framework

The following is a usability framework developed by Folmer *et al* for the mapping of different usability properties to usability the four usability attributes of Satisfaction, Learnability, Efficiency and Reliability [213]. This framework highlights architecture sensitive usability patterns and how they contribute to the realization of the usability properties. This framework has been utilized in the usability analysis of the PIASK architecture (Section 8.4).

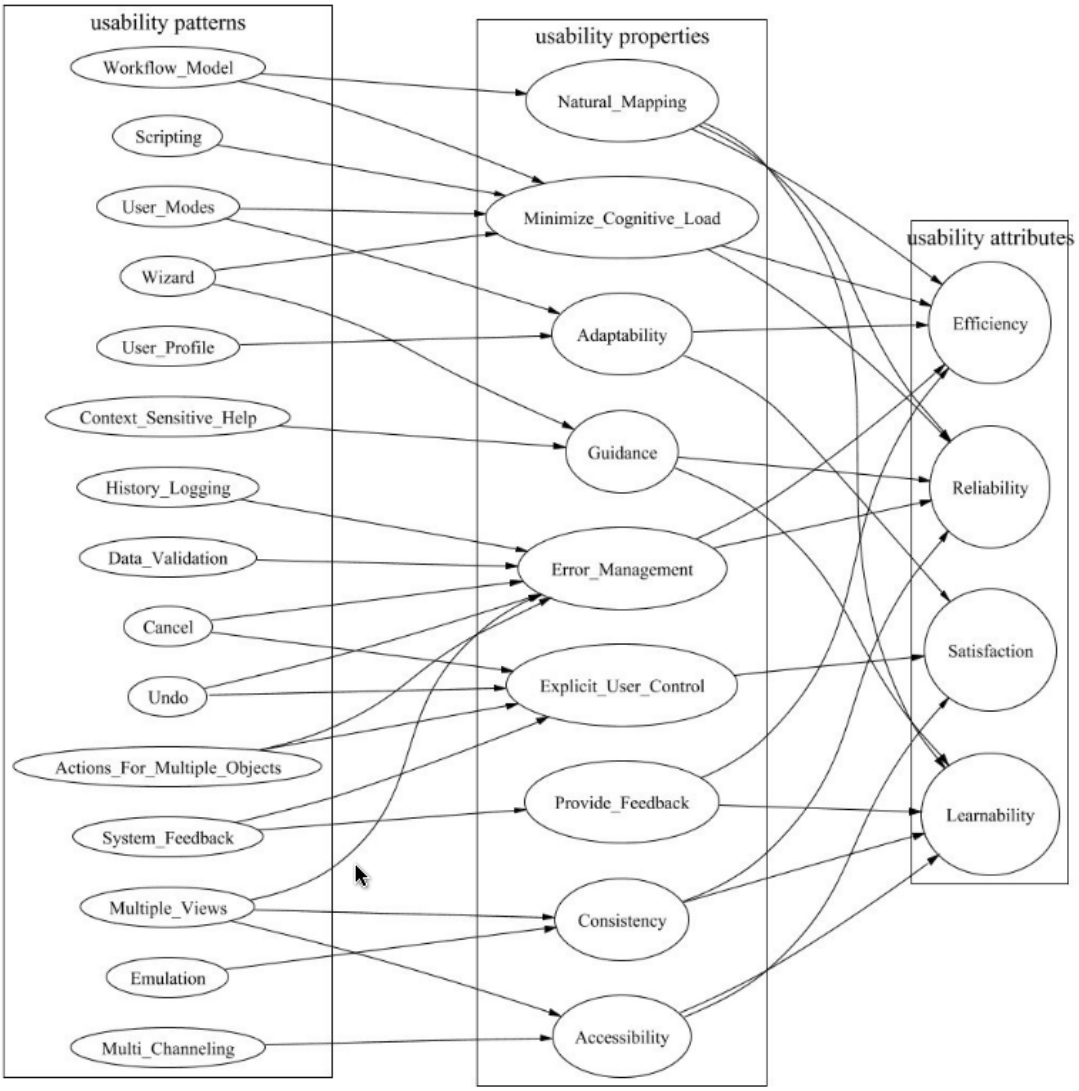


Figure I.1: Folmer *et al*. Usability Framework[213]

Appendix J

Published Papers

- THINYANE, M; TERZOLI, A & CLAYTON, P. (2008) *Old Wine, New Wine Skins: Exploring an ICT Intervention Towards the Protection of Indigenous Knowledge and Folklore*, SAFOS Conference, Lesotho, 23rd - 25th September 2008.

- THINYANE, M; TERZOLI, A & CLAYTON, P. (2008) *Transitions Towards a Knowledge Society: Aspectual Pre-evaluation of a Culture-sensitive Implementation Framework*, IFIP – Learning to Live in the Knowledge Society, Springer Boston, July 2008, Vol. 281, pp 271-278, ISBN: 978-0-387-09728-2

- THINYANE, M; DALVIT, L; TERZOLI, A & CLAYTON, P. (2008) *The Internet in Rural Communities: Unrestricted and Contextualized*, ICT Africa, Addis Abba - Ethiopia, Nepad Council, 2008.

- DALVIT,L; KASCHULA, R; MAPI, T; TERZOLI, A & THINYANE, M. (2007). *Integrating ICT-based Indigenous Knowledge in the teaching of isiXhosa to Rhodes University pharmacy students*. Kenton Conference, Phumula, KwaZuluNatal, 25 – 28 October 2007.

- THINYANE, M; DALVIT, L; MAPI, T; SLAY, H; TERZOLI A. & CLAYTON P. (2007) *An ontology-based, Multi-modal Platform for the Inclusion of Marginalised Rural Communities into the Knowledge Society*, ACM International Conference Series, ACM New York, 2007, Vol. 226, pp. 143-15, ISBN: 978-1-59593-775-9

- THINYANE, M; TERZOLI A. & CLAYTON P. (2007) *Exploring a Novel Service Deployment Framework, PIASK, Through the Design and Implementation of an E-Commerce*

Function for a Rural Community. Southern African Telecommunications Networks and Applications Conference, Mauritius, 9th - 13th September 2007.

- SLAY, H; THINYANE, M;& TERZOLI A. (2007) *MobileCom: A SIM-based Application to Support Second Economy Entrepreneurship*, Southern African Telecommunications Networks and Applications Conference, Mauritius, 9th-13th September 2007.

- TARWIREYI, P; THINYANE, M. & TERZOLI A. (2007) *Implementation of an Internet access cost management system for disadvantaged communities* . Southern African Telecommunications Networks and Applications Conference, Mauritius, 9th-13th September 2007.

- MOYO, T; THINYANE, M; WRIGHT, M; IRWIN, B; TERZOLI A. & CLAYTON P. (2007) *Bridging the gap for Next Generation Services: Presence Services on Legacy Devices* . Southern African Telecommunications Networks and Applications Conference, Mauritius, 9th-13th September 2007.

- DALVIT, L; THINYANE, M; TERZOLI, A & MUYINGI, H. (2007) *The Deployment of an e-commerce Platform and Related Projects in a Rural Area in South Africa*, SREC-07, ACM International Journal of Computing and ICT Research, June 2007, Vol 1 No 1, pp. 9-17, ISSN: 1818-1139

- THINYANE, M; DALVIT, L; TERZOLI, A & CLAYTON, P. (2007) *Towards a Model of an Ontology Based, Multi-Modal and Multimedia Knowledge Portal for Marginalized Rural Communities*. Proceedings of IEEE Information Communication Technologies International Symposium, Fez - Morocco, 3rd - 5th April 2007.

- THINYANE, M; SLAY, H; TERZOLI A. & CLAYTON P. (2006) *A preliminary investigation into the implementation of ICTs in marginalized communities*. Southern African Telecommunications Networks and Applications Conference, Spier - Western Cape, South Africa, September 2006.

- THINYANE, M; DALVIT, L; TERZOLI, A; ISABIRIYE, N. (2006) *A case study on the teaching of computer literacy in a marginalized community*. Comparative Education Society of Europe Conference, Granada, Spain, 3-6 July 2006.

J.1 Papers in submission

- THINYANE, M; TERZOLI A. & CLAYTON P. (2009) *A JADE MAS Platform for eServices Provision in a Community Development Context*, IEEE/ACM International Conference on Information and Communication Technologies and Development, Doha - Qatar, 17th - 19th April 2009.

- THINYANE, M; TERZOLI A. & CLAYTON P. (2009) *Preserving the Integrity of Folklore on Knowledge-based service Platforms*, Knowledge Management Africa 2009, Dakar - Senegal, 4th-7th May 2009.

Appendix K

Platform Poster

CONTEXT-SENSITIVE & ETHNO-CENTRIC KNOWLEDGE ORIENTED SERVICES PLATFORM

Mamello Thinyane, Alfredo Terzoli and Peter Clayton

Department of Computer Science, Rhodes University, Grahamstown - 6140

mthinyane@africanology.org - +27 46 603 8641

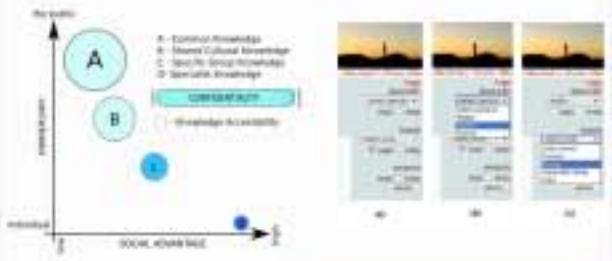
http://www.dwesa.org

Objectives

1. Design a knowledge management framework for marginalized communities
2. Explore knowledge systems (IKS) dynamics in marginalized communities
3. Develop a knowledge platform that handles heterogeneity and multiplicity of end-user devices, and usage modalities
4. Develop a knowledge-based eService deployment platform

PIASK and IK

Knowledge Matrix



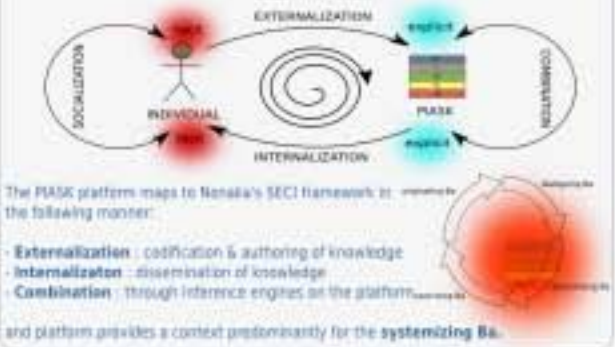
IK Life Cycle

The PIASK platform facilitates processing of indigenous Knowledge through:

- enabling the users to enforce the same permissions and levels of authorization on the underlying knowledge as in every day IKSs.
- allowing for a mapping of the different IK life cycle activities to the implemented PIASK specific processes.

The resultant effect is that the platform enables creation, processing, preservation and dissemination of IK, and a greater level of participation of the community in the global knowledge society .

PIASK and SECI



The PIASK platform maps to Nonaka's SECI framework in the following manner:

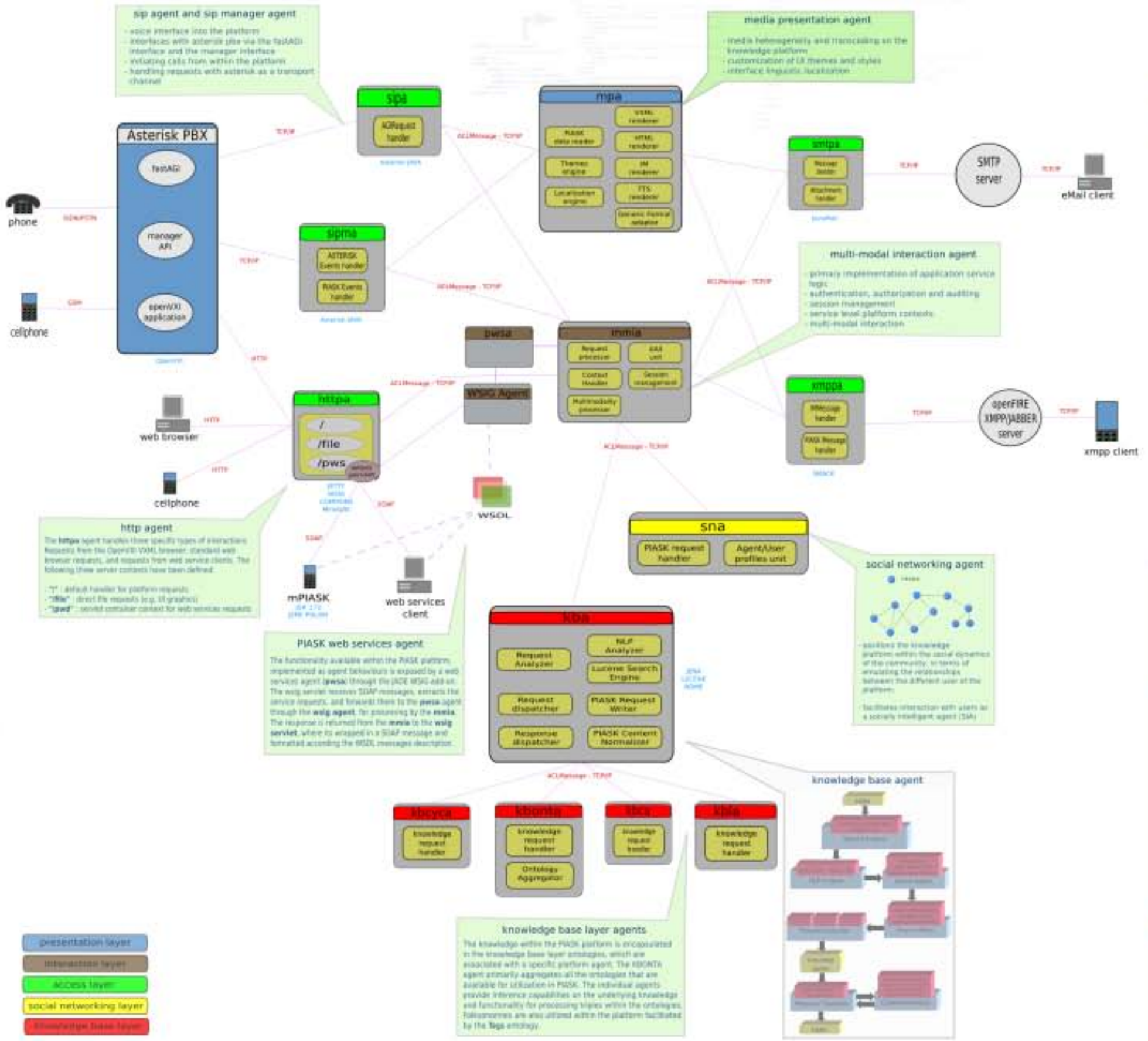
- **Externalization** : codification & authoring of knowledge
- **Internalizaton** : dissemination of knowledge
- **Combination** : through inference engines on the platform

and platform provides a context predominantly for the **systemizing Ba**.

PIASK architectural pattern

presentation layer	html, vxml, css, wml multi-media transcoding, ...
interaction layer	AAA, session management, multi-modality handling
access layer	http, sip, smtp, xmpp
social networking layer	feat, trust, user profiles
knowledge base layer	ontologies, folksonomies, okbc, rdf, owl, inference engines, dami-ol

Builds on the OOD principles of behavioural completeness to make a distinction between presentation elements of a system, and the underlying knowledge models. PIASK adopts a Component-Based Development (CBD) perspective by incorporating structural considerations for access heterogeneity and explicitly abstracting the social dynamics of utilizing knowledge services, through a distinct social networking layer



Service Deployment

The three alternatives to service deployment on the platform:

A - A web service client : that accesses the functionality exposed through the PIASK Web Services Agent (PWSA)
B - A platform agent : that participates in the PIASK Multi-Agent System (MAS) to provide added services logic
C - A knowledge layer ontology : extending the underlying data level ontologies in the platform

mPIASK

A J2ME interface into the PIASK platform utilizing JSR172

PIASK platform

The PIASK platform has been implemented as a MAS based on the JADE environment. The platform is architecturally distributed in a manner that maximizes service availability, robustness and flexibility.

Distributed, with Main-Container redundancy

End-to-end process handling on PIASK