Sudan University of Science and Technology
College of Graduate Studies

Declaration

I, the signing here-under, declare that I'm the sole author of the Ph.D. thesis entitled..... Enhancing Interlacking Institutional Worlds Integrity by using an Ontology Server ..... which is an original intellectual work. Willingly, I assign the copy-right of this work to the College of Graduate Studies (CGS), Sudan University of Science & Technology (SUST). Accordingly, SUST has all the rights to publish this work for scientific purposes.

Candidate's name: Abdelrahman Abbas Ibrahim Ahmed

Candidate's signature: ..... Date: 5/10/2015

إقرار

أنا الموقع أدناه أقر بأننى المؤلف الوحيد لرسالة الدكتوراه المعنونة تحسين تكاملية العوالم المؤسسية المتشابكة باستخدام خادم الأنطولوجيا

وهى منتج فكري أصيل . وباختياري أعطى حقوق طبع ونشر هذا العمل لكلية الدراسات العليا جامعه السودان للعلوم والتكنولوجيا ،عليه يحق للجامعه نشر هذا العمل للأغراض العلمية .

اسم الدارس : عبدالرحمن عباس ابراهيم أحمد

توقيع الدارس : ..... التاريخ : ٢٠١٥/١٠/٢٢

Sudan University of Science & Technology

College of Graduate Studies

بسم الله الرحمن الرحيم

جامعة السودان للعلوم والتكنولوجيا

كلية الدراسات العليا

كلية الدراسات العليا

Approval Page

Name of Candidate: Abdelrahman Abbas Ibrahim Ahmed

Thesis title: Enhancing InterLocking Institutional Worlds Integrity Using An Ontology Server

تحسين تكاملية العوالم المؤسسية المتشابكة باستخدام خادم الانطولوجيا

Approved by:

1. External Examiner

Name: Dr. Ahmed Kayed

Signature: Kayed Date: 12/8/2015

2. Internal Examiner

Name: Mohamed Elhafiz Mostafa Musa

Signature: Date: 12/8/2015

3. Supervisor

Name: Robert Colomb

Signature: Date: 12/8/2015

SUDAN UNIVERSITY OF SCIENCE AND TECHNOLOGY
COLLEGE OF GRADUATE STUDIES

ENHANCING INTERLOCKING INSTITUTIONAL WORLDS INTEGRITY BY USING AN ONTOLOGY SERVER

تحسين تكاملية العوالم المؤسسية المتشابكة باستخدام خادم الانطولوجيا

A thesis submitted in fulfillment of the Requirements
for degree of Doctor of Philosophy in Computer Science

By

ABDELRAHMAN ABBAS IBRAHIM AHMED

Supervisor: Prof. Dr. Robert Michael Colomb
Co-Supervisor Dr. Abdelgaffar Hamid Ahmed

August 2015

بسم الله الرحمن الرحيم

قال الله تعالى:

كَمَا أَرْسَلْنَا فِيكُمْ رَسُولاً مِّنكُمْ يَتْلُو عَلَيْكُمْ آيَاتِنَا وَيُزَكِّيكُمْ وَيُعَلِّمُكُمُ الْكِتَابَ وَالْحِكْمَةَ وَيُعَلِّمُكُم مَّا لَمْ تَكُونُواْ تَعْلَمُونَ{151} فَاذْكُرُونِي أَذْكُرْكُمْ وَاشْكُرُواْ لِي وَلاَ تَكْفُرُونِ{152}

سورة البقرة

DEDICATION

To the saintly soul of my ever love, my mother, the woman who devote her live to foster and encourage me to ascend staircase of successfulness

To the greatest man in my life, my father

To my brothers, sisters who shared my existence

To my advisors and sponsors

ACKNOWLEDGEMENT

Finally, I am very pleased to write this page. Principally, Praise to Allah (almighty), who has provided whatever talent I might have, and has given the great gifts, which have helped me to achieve this work. I would like to express my sincere gratitude to all those who gave me the opportunity to complete this thesis. First, I am deeply indebted to my supervisor Professor Robert M. Colomb, from him I received a great balance of freedom, monitoring, and guidance in gaining a challenging yet enjoyable learning experience of doing research. Second, I wish to express my gratitude to Assistant Professor Abdelgaffar Hamid whose help, critical suggestions and encouragement motivated me during my study.

Third, I wish to thank all academic and support staff from the faculty of computer science and information technology, assisting me in academic, administrative, technical, and social matters, specifically, Professor Dr. Ezzadeen Mohammed Othman, Associate professor Mohammed Elhafiz Mustafa. I wish to thank all academic and support staff from the Blue Nile University.

Last, but not least, I would like to give my special thanks to my family for their support and encouragements to do this research.

Abdelrahman Abbas

ABSTRACT

Ontology Engineering is a new development methodology for building ontologies. Ontologies are widely used to overcome the problem of interoperability between integrated information systems. Ontology in general is an agreed understanding of a certain domain. It can enable semantic interoperability where autonomous and distributed applications can meaningfully communicate to exchange data and interoperate independently of their internal technologies. Ontologies contain perdurant entities (entities that happen in time) as well as endurants (entities that exist in time). First, the study investigates the definition of ontology as a *specification* of a *conceptualization* by showing that an ontology supporting interoperating information systems can be seen as the result of *interlocking institutional worlds (**IWs**)*. ***IWs*** are collections of interlocked organizations interact together and exchange information to achieve partake tasks. Second, it specifies a software system, an *ontology server* needed to support domain ontology of IWs at run-time and design-time for ontology engineering.

The main goal of this thesis is twofold. First (general sense), it presents a *specification* for ontology engineering to guide ontology designers towards building ontology supporting interoperation of information system using a standard modeling approach. Second (specific sense), it presents a *specification* of an ontology server to serve as a mechanism for binding information systems together in specific domain and support them with agreed semantics.

The principal contribution of this work is the technical solution for ontology of perdurants representation. Based on speech act theory, and institutional fact, we present a technique for representing ontology of perdurants with a mechanism for managing instances of them. The proposed the technical solution (***POUP profile***) and the *ontology server frameworks* was applied in some examples of ***IWs*** and the results show that the combination of ***POUP*** with the ***ontology server*** provides a promise tool for enhancing semantic interoperability between ***IWs*** participants and hence, it mitigates semantic heterogeneity in ***IWs***.

المستخلص

هندسة الانطولوجيا منهجية حديثة لتطوير و بناء الانطولوجيا. تستخدم الانطولوجيا على نطاق واسع لحل مشكلة التوافق البيني بين أنظمة المعلومات المتكاملة. و هي بشكل عام عبارة عن مفاهيم متفق عليها مسبقاً في مجال معين ، و تمكّن الانظمة المستقلة من تبادل المعلومات دلالياً - بشكل مفهوم. تحتوى الانطولوجيا على عناصر أحداث (تحدث في زمن معين) بالاضافة الى عناصر كينونات (موجودة في وقت محدد). تناولت الدراسة تعريف الانطولوجيا باعتبارها مواصفات للمفاهيم و يمكن أن ننظر للأنطولوجيا التي تدعم أنظمة المعلومات المترابطة و مجموعة هذه الانظمة كعوالم مؤسسية متشابكة (IWS)، و هي مجموعات من المؤسسات المتشابكة تتفاعل معا وتبادل المعلومات لتحقيق مهام مشتركة.

الهدف الرئيس لهذه الرسالة مكون من شقين: الأول تقدم الدراسة توصيف معياري في مجال هندسة الانطولوجيا لقيادة مهندس الانطولوجيا نحو بناء انطولوجيا تدعم التبادل البيني بين نظم المعلومات باستخدام منهجية نمذجة معيارية. الثاني تقدم الدراسة توصيف معياري لخادم الانطولوجيا و الذي يعمل كتقنية لربط انظمة المعلومات معا في مجال معين مع تذويده بمعاني و مفاهيم متفق عليها.

الاسهام الرئيس لهذه الدراسة هو الحل التقني لتمثيل انطولوجيا الاحداث و الذي يعتمد على نظرية (الافعال اللغوية للكلام) و الحقائق المؤسسية. يتمثل هذا الحل في تطوير بروفايل يمثل امتداد للغة النمذجة المعيارية (UML) لنمذجة انطولوجيا مع آلية لادارة مكونات انطولوجيا الاحداث. وبعد تطبيق هذه التقنيات على بعض الحالات من الانظمة المؤسسية المتشابكة (IWs) اظهرت النتائج أن الجمع بين البروفايل (POUP) مع خادم الأنطولوجيا يوفر أداة واعدة لتحسين العمل المشترك الدلالي بين المشاركين في العوالم المؤسسية المتشابكة وبالتالي، فإنه يخفف من مشكلة عدم التجانس الدلالي في الانظمة المؤسسية المتشابكة.

TABLE OF CONTENTS

Contents

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

TERM	ABBREVIATION
API	Application Programming Interface
BBW	Bunge–Wand–Weber
CASE	Computer-Aided Software Engineering
DEMO	Dynamic Essential Modeling of Organizations
DL	Descriptive Logic
DOLCE	Descriptive Ontology for Linguistic and Cognitive Engineering
DSML	Domain specific modeling language
EDI	Electronic Data Interchange
EMF	Eclipse Modeling Framework
GFO	General Formal Ontology
HFIWs	Halal Food interlocking institutional worlds
HFO	Halal Food Ontology
HFOS	Halal Food Ontology Server
IS	Information System
IWs	Interlocking Institutional worlds
MBE	Model Based Engineering
MDE	Model Driven Engineering
MDA	Model Driven Architecture
MDD	Model Driven Development
MDSE	Model Driven System Engineering
MOF	Metamodel Object Facility
OCL	Object Constraint Language
ODM	Ontology Definition Metamodel
OE	Ontology Engineering
OMG	Object Management Group
OS	Ontology Server
OWL	Web Ontology Languages
OWL-S	Web Ontology markup Languages for services
POUP	Perdurant Ontology UML Profile
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema

SE	Software Engineering
SCM	Supply Chain Management
SWS	Semantic Web Services
UML	Unified Modeling Language
WS	Web Services
XMI	XML Metadata Interchange
XML	Extensible Markup Language
XSD	XML Schema Definition

CHAPTER 1

INTRODUCTION

1.1. Introduction

Conceptual modeling is concerned with identifying and describing the basic concepts of a domain depending on modeling languages, which is based on essential meta-concepts. Ontological modeling is concerned with identifying relevant entities of a domain with the help of ontology specification languages that based on small set of basic, domain independent ontological categories (Horkoff and Maiden, 2015).

Ontology is widely used in the semantic web as a solution of information systems integration problem. In order for these integrated information system (which is called interlocking institutional world and abbreviated to ***IWs***) to interoperate; there must be an agreement on terms and words that could be used during the interoperation process, the ontology provides these interlocking institutional worlds with such vocabulary. Ontology is very complex information object; information systems are being used to manage complex information objects, in order to manage this complexity there must be an information system to provide such facility, this information system is the ***ontology server*** (Colomb, 2007).

When integrated information system interoperate they exchanges messages which may perform an action or send commands for querying information, these acts are called ***speech acts***, often speech acts make changes in the ***IWs*** and creates new facts. According to Searle and Colomb (Colomb, 2007, Searle, 1995) there are two types of facts, brute facts and institutional facts. A brute fact is about something in the physical world that is independent of human society, while an institutional fact is dependent on human society, for example a coin is a piece of metal in a round shape; this is a brute fact, but in human society it's something used in selling and buying goods. Speech acts are something said that affects and changes the world, when you placing an order you submit a speech acts. An institutional fact is a record of a speech act having been made, for example, a drive certificate is a record of the

speech act of authorizing to someone to drive a motor vehicle on public roads after passing a series of driving tests.

Institutions could be seen as information systems that create a collection of instances of institutional facts, which are created by speech acts. These institutions are called institutional world for example in Halal food supply chain farmers, Butcheries, wholesalers and retailers are institutional worlds. When two or more institutional worlds share their systems of institutional facts, then they called interlocking institutional worlds.

Ontology-Based ***IWs***, that exchanging institutional fact and perform speech acts depend on both endurant and perdurant ontologies. Endurant ontology describes entities in a hierarchy and relationships between them while perdurant represent events and actions that could be taken to perform speech act during interoperation. For instance in Amazon supply Chain buying an element is a speech act, which involves number of processes will be performed by different institutions, such as paying through American Express, Diners Club, Discover, or Visa Check Cards. Each process of payment through one of these institutions is an instance of paying speech act.

This work contributes to domain of ontology modeling, in one hand it proposes a UML profile for representing perdurant ontology, on other the hand, it proposes a framework for ontology server, which will be constructed to help in enhancing interlocking institutional world's integrity. The ontology server helps in recording and retrieving information of transactions done during interoperation between IWs' institutions. Both the profile and ontology server framework will be examined via Halal Food IWs case study.

1.2. Motivations

Currently, information systems are everywhere, every business or organization is supported with an information system in every aspect of our live as Robert Colomb stated in (Colomb, 2007): "*"…companies in the finance, insurance and real estate industry group are very little more than information systems. Mines and farms use information systems to keep track of production and assets. Manufacturers, wholesalers and retailers use information systems to manage their production, sales and employees.*

Construction firms use information systems to bid for and manage projects. Transportation firms use them to schedule services. The health sector is served by thousands of systems assisting in the operation of various departments in hospitals, doctors" surgeries, and the flow of payments through the system. Universities use information systems to keep track of students, courses, libraries and staff. Governments use them to record births, deaths and marriages, and in the provision of all sorts of services…"

Therefore, there are millions of information systems, in one domain, for example in education, there are thousands of them, although they might serve the same objectives, they use different notations and technologies. Some of them are web-based while others are desktop-based; some of them are platform independent while some of them are depend on specific platform. Nowadays, most organization tend to integrate with each other in order do business together, some organizations need to outsource some services from other organizations, therefore their information systems should integrate in order to exchange information.

Ontologies are widely used in the semantic web and Artificial Intelligence to support syntactic and semantic interoperation, machines and information systems can exchange information and understand each other semantically. Ontology is concern with shared specification of conceptualization as Gruber stated in (Gruber, 1993); all domain concepts involve objects, their properties and the relationship between them should be specified with standardized representation mechanism.

Ontologies are a complex information object it consists of thousands categories of thing and relationships. Ontology server is an information system intended to manage an ontology during it is live-cycle in design, commit and run time. The ontology server is originally designed for different purposes in supporting a lifecycle of ontology-based applications.

1.3. Problem Statements

In the real world, there are number of integrated information systems that do business together to provide their end customers with high quality products. In order to achieve this task; they must interoperate together and exchange raw materials, goods as well as information, one example of these interlocking

institutional worlds is supply chains, which consist of a number of information systems. For instance Halal food supply chain consist of a number of institutions (i.e. firms and organizations) each of which has an information system, these institutions work together to provide their customers with healthy and high quality food, but each institution keeps its own information as private data, and hence it is difficult for them to interoperate effectively.

The problem is that, each participant (institution) in the ***IWs*** needs to interoperate with other participants; they could not understand each other unless they agree on consensus vocabulary. Consensus vocabulary defines shared terms and set of standardized operations used in the ***IWs***. For example, in halal food supply chain there should be definitions for all halal ingredients and products, and procedures for producing halal products and issuing halal certificates. Procedures are standardized series of events all participants should follow to achieve specific process. Furthermore, they could be used in auditing, tracking, and tracing activities. We can outline problem statements in the following questions:

1. ***How can we represent various transactions and events in the IWs? Moreover, how can we record and make available instances of these transactions in the IWs, which might be generally private to interoperating institutions?***

The word **record** point out the process of saving instance of every action that will take place within the IWs during the interoperation, as in ***database transaction log*** all events are stored in the database log for tracking complete or incomplete transactions in order rollback in case of incompleteness. In our case, we will use these recorded transactions (actions) to make sure that all processes of interlocking institutional worlds are done according to the framing rules that governs them and guide its interoperation.

2. ***What sort of governance mechanisms do we need to hold cooperating institutions in IWs together to facilitate interoperation?***

Ontology is always situated outside interoperating information systems. Because it is independent of all participating information systems and hence there should be an independent specification to hold both, ontology and ontology data, this

specification should enable all participants to commit to that ontology, beside allowing them to update and editing ontology's data. In other words, this specification governs all operation related to editing, versioning, committing to, updating, querying and inference ontology's data.

3. ***How will do such mechanisms scale to cover a large amount of IWs'.***

The specification of independent mechanism discussed in previous paragraph should be scalable to involve all current participants and potential participant predicted to commit to it, and all participants in the same domain. In this study, all mentioned research question would be addressed in details.

1.4. Research Objectives

With regarding to the research questions stated in the previous section, research objectives subsequently transformed to the following:

Table 1-1: Objectives of the study

Objective	Statement
Obj1	To establish a perdurant ontology profile capable of representing transactions (action and processes) needed during interoperation
Obj2	To specify a mechanism to govern all operations related to ontology of both endurants and perdurants and hold cooperating organizations together in IWs.
Obj3	To generalize the mechanism stated in bj2 to cover most of IWs' current participants and potential.

1.5. Significance of the Study

This section presents a brief description of the various significances of the study given in three categories; interlocking institutional worlds' interoperability, ontology specification and ontology server development.

1.5.1.To IWS Interoperability

Institutional worlds i.e. Enterprises and firms, need to co-operate and exchange information, traditional mechanisms such as Electronic Data Interchange (**EDI**)

which does not provide enough semantics, and only bind pair-wise enterprises. This study will contribute to the knowledge and understanding of the subject of interlocking institutional worlds' interoperability, by improving semantic interoperability and mitigate semantic heterogeneity and this is very important for today's business.

1.5.2.Ontology Specification

Ontology design and usage is not yet full matured for the semantic web and information system integration, specifically designing of the ontology for processes (***Perdurant ontology)***. The focus of this study is mainly on perdurant ontologies, which is very important to represent events and processes (***speech acts)***.

1.5.3. Ontology Server Development

This study is very important to the area of ontology server development, because specifications of ontology server's functionality on different lifecycle phases are not fully specified. Furthermore, the literature shows that no sufficient researches were done in this field and more studies should be adopted to cover this area. The study specifies some functionality of the server in run time such as manipulating ontology data and metadata, query for information, and inferencing new data. Moreover, it provides a mechanism for managing consensus vocabulary across institutions in specific domain. An authorized body needed to facilitate the semantic interoperability and hold all IWs' participants together.

1.6. Scope of the Study

The subject materials in this thesis should be seen from the conceptual modeling perspective so it is independent of any technical implementation point of view. Our research context is defined in the sense of ontology supporting the interoperation of information systems understood through the notion of IWs, in general, and the development of conceptual models (domain ontology) for an ontology server supporting IWs at commit-time, and run-time in particular. There are many kinds of ontology such as ontology of perdurant, task ontologies, application ontologies, upper-level ontologies (formal ontology) and domain ontologies. Some concepts

related to conceptual and ontology modeling not in our scope such as ontology learning and modularization, ontology merging and alignment.

1.7. Contributions

This study contributes to area of ontology specification and modeling, and ontology server development. It extends Descriptive Ontology for Linguistics and Cognitive Engineering (DOLCE) upper ontology and establishing new concepts from ancient philosophy and theory of speech act from some linguistic philosophy. The new concepts will help in distinguishing between items in the real world and in information systems. The study also presents an UML profile for modeling perdurant ontology capable of representing instances of perdurants. The profile conforms to DOLCE upper ontology. The study also presents a framework for ontology server, to facilitate and enhancing the semantic interoperability between IWs' participants.

1.8. Organization of the Thesis

To achieve objectives in section (1.4) we should follow a scientific approach in thesis design. Figure (1.1) below describes how the thesis carrying out these objectives, tailored from (Guizzardi, 2005). The structure of the thesis composed of three main parts preceded by ***introduction*** and concluded with ***conclusions***.

Chapter 1, ***the introduction***, illustrates the study main building blocks; it discusses research questions, objectives, scope, motivations, contributions and thesis structure.

The first part, ***the literature review***, contains three chapters provide a comprehensive theoretical background and review of the literature. Chapter 2 discusses ontology and ontology server literature. It gives a wide discussion about ontology definitions in philosophy and information technology and ontology related concepts. Chapter 3 reviews the IWs concepts; the review includes IWs definition, speech act theory and IWs integrity. Chapter 4 presents current model based engineering concepts and modeling tools.

The second part, the ***proposed approach***, composes of two chapters. Chapter 5 discusses the proposed UML profile for perdurant ontology (**POUP)** based on speech act theory. Chapter 6 shows the framework of an ontology server for managing ontologies.

The third part, ***Case studies,*** has only one chapter***, Halal food IWs***. Chapter 7 is for evaluating proposed approaches suggested in chapter 5 and 6. The chapter defines the case participants, roles, static and behavioral views, designing Halal ontology of endurants and perdurants. It also discusses the results.

Finally, the thesis ends up with Chapter 8, which shows the ***conclusions and future work.***

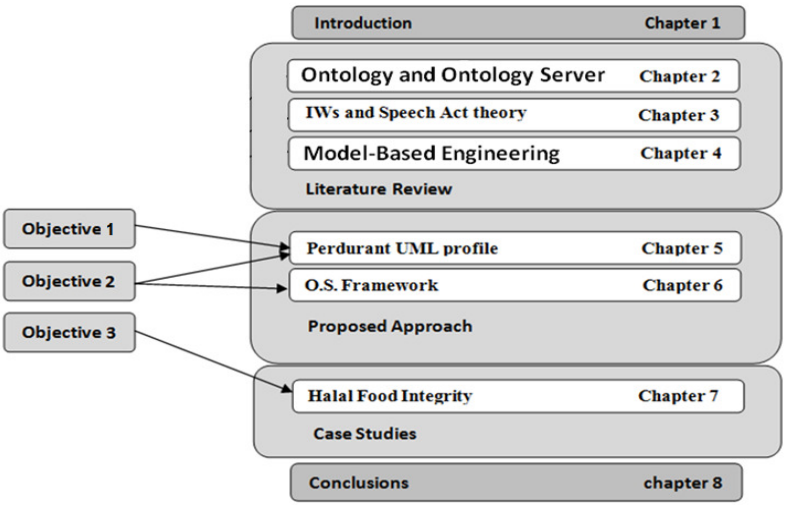

Figure 1-1: Organization of the thesis

1.9. Summary

In this chapter we have provided a quick overview of the thesis, we have shown the research questions, objectives, motivations, importance of the study, the scope of the study, and the organization of the thesis , as well as main contributions and thesis structure.

CHAPTER 2

ONTOLOY AND ONTOLOGY SERVER

2.1. Introduction

In this chapter, some background information regarding ontology and ontology server will be provided. Understanding the notion of ontology in general, and the usage of ontology for supporting the interoperation of information systems, in particular, may help us to understand a major theme and context described in the remainder of this thesis. Furthermore, it will provide an explanation of relevant terminology including definitions of ontologies and conceptual models, the notion of a conceptualization and a specification as well as interoperation of information systems. Moreover, it will briefly describe several traditional representation systems (modeling languages) for conceptual modeling, in general, and ontology modeling in particular. In the sense of a language evaluation framework, we discuss how suitable these systems are to model phenomena according to a given real-world conceptualization.

2.2. Philosophical Background

In philosophy, ontology is the most fundamental branch of metaphysics. It is a mature discipline, which has been systematically developed in western philosophy at least since Aristotle. Ontology as a branch of philosophy is the science of what is, of the kinds and structures of objects, properties, events, processes and relations in every area of reality (Smith, 2003). ***Ontology*** is often used by philosophers as a synonym of ***metaphysics*** (a label meaning literally: '*what comes after the Physics*'), a term used by early students of Aristotle to refer to what Aristotle himself called ***first philosophy*** (Viinikkala, 2005). Sometimes 'ontology' is used in a broader sense, to refer to the study of what *might* exist; 'metaphysics' is then used for the study of which of the various alternative possible ontologies is in fact true of reality (Smith, 2003). The term ontology or ***ontologia*** was coined in 1613 (Ingarden,

1964), independently, by two philosophers, Rudolf Göckel ***Goclenius***, in his Lexicon philosophicum and Jacob Lorhard ***Lorhardus***, in his Theatrum philosophicum. Its first occurrence in English as recorded by the OED appears in Bailey's dictionary of 1721, which defines ontology as '*an Account of being in the Abstract'*.

Ontology seeks to provide a definitive and exhaustive classification of entities in all spheres of being. The classification should be definitive in the sense that it can serve as an answer to such questions as: What classes of entities are needed for a complete description and explanation of all the goings-on in the universe? Or: What classes of entities are needed to give an account of what makes true all truths? It should be exhaustive in the sense that all types of entities should be included in the classification, including also the types of relations by which entities are tied together to form larger wholes.

Different schools of philosophy offer different approaches to the provision of such classifications. One large division is that between what we might call substantialists and fluxists, which is to say between those who conceive ontology as a substance- or thing- (or continuant-) based discipline and those who favor an ontology centered on events or processes (or occurrents). Another large division is between what we might call adequatists and reductionists. Adequatists seek taxonomy of the entities in reality at all levels of aggregation, from the microphysical to the cosmological, and including the middle world (the *mesocosmos*) of human-scale entities in between. Reductionists see reality in terms of someone privileged level of existents; they seek to establish the 'ultimate furniture of the universe' by decomposing reality into its simplest constituents, or they seek to 'reduce' in some other way the apparent variety of types of entities existing in reality.

Aristotle defines ontology as 'the science of being' (Abugessaisa and Sivertun, 2004). This definition can be reformulated as 'the science of being with regards to the aspect of being'. Ontology as a branch of philosophy is the science of what is, of the kinds and structures of the objects, properties and relations in every area of reality. In simple terms, it seeks the classification of entities. Ontology is descriptive, which means focused on the classification of existing entities.

2.3. Ontology Definitions

In the context of computer and information sciences, the ontology defines a set of representational primitives with which to model a domain of knowledge or discourse. The representational primitives are typically classes (or sets), attributes (or properties), and relationships (or relations among class members). The definitions of the representational primitives include information about their meaning and constraints on their logically consistent application. In the context of database systems, ontology can be viewed as a level of abstraction of data models, analogous to hierarchical and relational models, but intended for modeling knowledge about individuals, their attributes, and their relationships to other individuals. Ontologies are typically specified in languages that allow abstraction away from data structures and implementation strategies; in practice, the languages of ontologies are closer in expressive power to first-order logic than languages used to model databases. For this reason, ontologies are said to be at the "semantic" level, whereas database schema are models of data at the "logical" or "physical" level. Due to their independence from lower level data models, ontologies are used for integrating heterogeneous databases, enabling interoperability among disparate systems, and specifying interfaces to independent, knowledge-based services. In the technology stack of the Semantic Web standards, ontologies are called out as an explicit layer. There are now standard languages and a variety of commercial and open source tools for creating and working with ontologies.

There are many definitions of the concept of ontology in AI and in computing in general. The most widely cited one is:

"*Ontology is a specification of a conceptualization*".(Gruber, 1993)

Actually, this definition is the most concise one, and requires some further clarification. It depends on two main concepts "***Specification***" and "***Conceptualization***" the two main concepts will be discussed respectively.

Conceptualization means an abstract, simplified view of the world. If the knowledge base of an intelligent system is to represent the world for some purpose, then it must be committed to some conceptualization, explicitly or implicitly. That

is, every body of formally represented knowledge is based on a conceptualization. Every conceptualization is based on the concepts, objects, and other entities that are assumed to exist in an area of interest, and the relationships that exist among them. This also clarifies the meaning of the term ***world***—in practice, ***world*** actually refers to some phenomenon in the world, or to some topic (or topics), or to some subject area.

The second main concept in the above definition—***specification***—means a formal and declarative representation. In the data structure representing the ontology, the type of concepts used and the constraints on their use are stated declaratively, explicitly, and using a formal language. The formal representation implies that ontology should be *machine-readable.* However, ontology is not "active;" it cannot be run as a program. It represents declaratively some knowledge to be used by programs.

"Ontology . . . can be seen as the study of the organization and the nature of the world independently of the form of our knowledge about it." **(Guarino, 1995)**

Guarino augments the above definition with the notion of a *formal ontology,* the theory of a priori distinctions between the entities of the world (physical objects, events, regions, quantities of matter,), as well as between the meta-level categories used to model the world (concepts, properties, qualities, states, roles, parts, …). Fundamental roles are played in formal ontology by the theory of part–whole relations and topology (the theory of the connection relation).

Ontology is a set of knowledge terms, including the vocabulary, the semantic interconnections, and some simple rules of inference and logic for some particular topic. **(Hendler, 2001)**

The important parts in Handler's definition are the *semantic interconnections,* and *inference and logic.* The former says that ontology specifies the meaning of relations between the concepts used. In addition, it may be interpreted as a suggestion that ontologies themselves are interconnected as well; for example, the ontologies of "hand" and "arm" may be built to be logically, semantically, and formally interconnected. The latter part means that ontologies enable some forms of

reasoning. For example, the ontology of “musician” may include instruments and a specification of how to play them, as well as albums and how to record them.

Swartout and Tate offer an informal and metaphorical but extremely useful definition for understanding the essentials of ontology:

"Ontology is the basic structure or armature around which a knowledge base can be built." **(Swartout and Tate, 1999)**

As an armature in concrete, ontology should provide a firm and stable knowledge skeleton to which all other knowledge should stick. Another important issue here is the distinction between ontological knowledge and all other types of knowledge. Ontology represents the fundamental knowledge about a topic of interest; it is possible for much of the other knowledge about the same topic to grow around the ontology, referring to it, but representing a whole in itself.

Kalfoglou stresses yet another important issue related to ontologies:

"An ontology is an explicit representation of a shared understanding of the important concepts in some domain of interest." **(Kalfoglou, 2001)**

The word shared here indicates that ontology captures some consensual knowledge. It is not supposed to represent the subjective knowledge of some individual, but the knowledge accepted by a group or a community. All individual knowledge is subjective; ontology implements an explicit cognitive structure that helps to present objectivity as an agreement about subjectivity. Hence, ontology conveys a shared understanding of a domain that is agreed among a number of individuals or agents. Such an agreement facilitates accurate and effective communication of meaning. This, in turn, opens up the possibility for knowledge sharing and reuse, which enables semantic interoperability between intelligent agents and applications.

2.4. Ontology Importance

Ontology defines a common vocabulary for researchers who need to share information in a domain. It includes machine-interpretable definitions of basic

concepts in the domain and relations among them. Why ontology is important? Why would someone want to develop an ontology? There are definitely some reasons: ontology can facilitate sharing a common understanding of the structure of information among people or software agents, enables reuse of domain knowledge, makes domain assumptions explicit, separates domain knowledge from the operational knowledge, finally it helps to analyze domain knowledge. We can summarize the benefits of ontology as follows:

- It provides a common and shared understanding definition about certain key concepts in the domain.
- It offers the terms one can use when creating RDF documents in the domain.
- It provides a way to reuse domain knowledge.
- It makes the domain assumptions explicit.
- Together with ontology description languages (such as RDFS and OWL), it provides a way to encode knowledge and semantics such that machines can understand.
- It makes automatic large-scale machine processing possible.

2.5. Types of Ontologies

In the literature (Fensel, 2003, Genesereth, 1998, Heflin and Hendler, 2000, Studer et al., 1998) we generally find three common layers of knowledge. Based on their levels of generality, these three layers correspond to three different types of ontologies, namely:

- ***Upper level Ontology*** (Generic or top-level ontologies), which capture general, domain independent knowledge (e.g. space and time). Examples are ***WordNet*** (Miller and Fellbaum, 1998) and CYC (Lenat, 1995) . Generic ontologies are shared by large numbers of people across different domains.
- ***Domain ontologies*** capture the knowledge in a specific domain. An example is NSPSC, which is a product classification scheme for vendors. Domain ontologies are shared by stakeholder in a domain.
- ***Application ontologies*** capture the knowledge necessary for a specific application. An example could be an ontology representing the structure of a particular Web site. Arguably, application ontologies are not really ontologies, because they are not really shared.

2.6. Upper Level Ontologies

Upper level ontologies are used to facilitate the semantic integration of domain ontologies and guide the development of new ontologies. For this purpose, they contain general categories that are applicable across multiple domains. Upper level ontologies usually provide rich definitions and axioms for their categories. Different upper level ontologies provide different distinctions based on the kinds of entities they include, their theories of space, and time as well as the relation of individuals to space and time.

2.7. Some Examples of Upper Ontology

A number of formal upper ontologies have been proposed. We will present here some of them, the Bunge-Wand-Weber (BWW) system, the DOLCE system developed by the OntoClean project and **Basic Formal Ontology (BFO)**. These ontologies are different, but are compatible with each other. Each emphasizes different aspects of form.

Bunge-Wand-Weber (BWW) formal upper ontology was developed around 1990s by ***Wand*** (from Canada) and ***Weber*** (from Australia). They developed it on from a more philosophical ontology developed around 1977 by ***Mario Bunge*** (originally from Argentina) for many years. BWW follows some of Bunge's original ideas, but not all.

DOLCE Upper ontology: Descriptive Ontology for linguistics and Cognitive engineering (DOLCE) is one of famous upper ontologies. It is the first module of the WonderWeb foundational ontologies library. (Masolo et al., 2003a) As implied by its acronym, DOLCE has a clear cognitive bias, in that it aims at capturing the ontological categories underlying natural language and human common sense.

Basic Formal Ontology (BFO): developed by Barry Smith(Smith and Grenon, 2002) , is a foundational or upper-level ontology used in information science for the description of entities at the highest level of generality.

2.8. Ontology In Computer and Information Science

Ontologies have been applied in a multitude of areas in computer science. The first noticeable growth of interest in the subject in mid 1990.s was motivated by the need to create principled representations of domain knowledge in the knowledge sharing and reuse community in AI. Which motivated the creation of forums such as the conference series FOIS (Formal Ontology and Information Systems). Ontologies are widely used in the following fields: artificial intelligence, the Semantic Web, systems engineering, software engineering, biomedical informatics, library science, enterprise bookmarking, and information architecture they all create ontologies to limit complexity and to organize information. The ontology can then be applied to problem solving.

2.9. Ontology and The Semantic Web

A key feature of ontologies is that, through formal, real-world semantics and consensual terminologies, they interweave human and machine understanding (Fensel, 2003). This important property of ontologies facilitates the sharing and reuse of ontologies among humans, as well as among machines. A major reason for the recent increasing interest in ontologies is the development of the Semantic Web (Berners-Lee et al., 2001) , which can be seen as knowledge management on a global scale. Tim Berners-Lee, inventor of the current World Wide Web and director of the World Wide Web Consortium (W3C), envisions the Semantic Web as the next generation of the current Web. This next generation will expand upon the prowess of the current Web by adding machine-readable information and automated services.

Fensel argue that "*The explicit representation of the semantics underlying data, programs, pages, and other Web resources will enable a knowledge-based Web that provides a qualitatively new level of service*" (Fensel, 2003). Ontologies provide such an explicit representation of semantics. The combination of ontologies with the Web has the potential to overcome many of the problems in knowledge sharing and reuse and in information integration.

Ontologies interweave human and computer understanding of symbols. These symbols, also called terms and relations, can be interpreted by both humans and machines. The meaning for a human is represented by the term itself, which is

usually a word in natural language, and by the semantic relationships between terms. An example of such a human-understandable relationship is a super-concept, sub-concept relationship, often referred to by the term ***is-a***. Such a relationship denotes the fact that one concept (the super-concept) is more general than another is (the sub-concept). For instance, the concept **Person** is more general than **Student** is. Figure 2.1 shows an example "is-a" hierarchy (or taxonomy), where the more general concepts are located above the more specialized concepts.

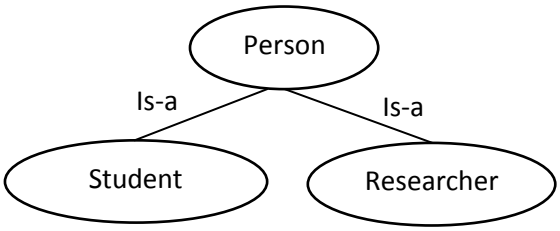

Figure 2-1: is-a taxonomy

Ontologies are considered one of the pillars of the Semantic Web, although they do not have a universally accepted definition. Figure (2.2) shows the position of the ontology within the semantic web cake. It is obvious that the ontology is built on top of RDF(S) which itself based on XML. The semantic web layer cake or semantic web architecture stack was presented by Tim Berners-Lee director of the World Wide Web Consortium (Kifer et al., 2005).

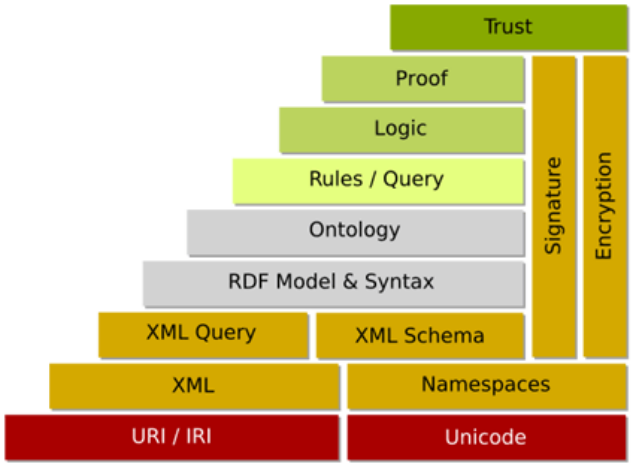

Figure 2-2: Semantic Web Cake

2.10. Ontology Engineering

Ontology engineering (OE) is a subfield of knowledge engineering that studies the methods and methodologies for building ontologies. It studies the ontology development process, the ontology life cycle, the methods and methodologies for building ontologies, and the tool suites and languages that support them.

OE is concerned with making representational choices that capture the relevant distinctions of a domain at the highest level of abstraction while still being as clear as possible about the meanings of terms. As in other forms of data modeling, there is knowledge and skill required. The heritage of computational ontology in philosophical ontology is a rich body of theory about how to make ontological distinctions in a systematic and coherent manner. For example, many of the insights of "formal ontology" motivated by understanding "the real world" can be applied when building computational ontologies for worlds of data (Gruber, 1995).When ontologies are encoded in standard formalisms; it is also possible to reuse large, previously designed ontologies motivated by systematic accounts of human knowledge or language (Guizzardi, 2005).

In general, ontology is the study or concern about what kinds of things exist - what entities there are in the universe. It derives from the Greek onto (being) and logia (written or spoken discourse). It is a branch of metaphysics , the study of first principles or the essence of things.

In information technology, ontology is the working model of entities and interactions in some particular domain of knowledge or practices, such as electronic commerce or the activity of planning. In artificial intelligence (AI), an ontology is, according to Tom Gruber, "*the specification of conceptualizations, used to help programs and humans share knowledge*." In this usage, an ontology is a set of concepts - such as things, events, and relations - that are specified in some way (such as specific natural language) in order to create an agreed-upon vocabulary for exchanging information

2.11. Ontology Representation Languages:

An ontology language is a formal language used to encode the ontology. There are a enormous of such languages for ontologies, in the following sections we will

address some of them, but before that we will address ontology languages requirements.

2.11.1.Requirements for Ontology Languages

Ontology languages allow users to write explicit, formal conceptualizations of domains models. There are five main requirements ontology language should provide: it should be *well-defined syntax*, *well-defined semantics*, *efficient reasoning support*, *sufficient expressive power*, and finally, *convenient of expression*.

2.11.2. Resource Description Framework (RDF)

The Resource Description Framework (RDF) (Klyne and Carroll, 2006), is the first language developed especially for the Semantic Web. RDF was developed as a language for adding machine-readable metadata to existing data on the Web. RDF uses XML for its serialization in order to realize the layering depicted in the Semantic Web language layer cake (Figure 2.2). RDF Schema (Brickley and Guha, 2000) extends RDF with some basic (frame-based) ontological modeling primitives. There are primitives such as classes, properties, and instances. Also, the instance-of, subclass-of, and sub-property-of relationships have been introduced, allowing structured class and property hierarchies. RDF has the subject–predicate–object triple, commonly written as P(S,O), as its basic data model. An object of a triple can, in turn, function as the subject of another triple, yielding a directed labeled graph, where resources (subjects and objects) correspond to nodes, and predicates correspond to edges. Furthermore, RDF allows a form of reification (a statement about a statement), which means that any RDF statement can be used as a subject in a triple. An example RDF graph is shown in Figure 2.3. The corresponding RDF/XML serialization is shown in Figure 2.4.

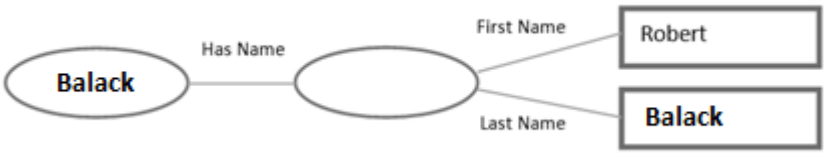

Figure 2-3: An example of RDF Graph

```
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:ex="http://example.org/#"
xml:base="http://example.org/">
<rdf:Description about="#Balack">
<ex:hasName rdf:parseType="Resource">
<ex:firstName>Robert</ex:firstName>
<ex:lastName>Balack</ex:lastName>
</ex:hasName>
</rdf:Description>
</rdf:RDF>
```

Figure 2-4: The corresponding RDF/XML serialization

2.11.3.RDF Schema (RDFS)

RDF Schema (RDFS) (Brickley and Guha, 2000) is a lightweight ontology language for defining vocabularies for RDF. Unlike XML Schema, which prescribes the order and combinations of tags (the structure) in an XML document, RDF Schema only provides information about the interpretation of the statements given in an RDF data model. RDF Schema does not say anything about the syntactical appearance of the RDF description. RDFS can in fact be seen as an extension of RDF with a vocabulary for defining classes, class hierarchies, properties (binary relations), property hierarchies, and property restrictions. RDFS classes and properties can be instantiated in RDF. For a more detailed comparison of XML Schema and RDF Schema we refer the reader to (Klein et al., 2003).

RDF(S) is not very expressive compared with many other ontology languages, as it allows only the representation of concepts, concept taxonomies, and binary relations. The expressive limitations of RDF(S) were a major motivation for developing languages that are more expressive for the Semantic Web. In the following section, we describe the ontology (OWL) which is layered on top of RDF(S).

2.11.4.Web Ontology Language (OWL)

The OWL Web Ontology Language is designed for use by applications that need to process the content of information instead of just presenting information to humans. OWL facilitates greater machine interpretability of Web content than that supported by XML, RDF, and RDF Schema (RDF-S) by providing additional vocabulary along with a formal semantics.

The Web Ontology Language OWL (McGuinness and Van Harmelen, 2004) is an expressive ontology language which extends RDFS. OWL has three increasingly expressive sublanguages: OWL Lite, OWL DL, and OWL Full.

· ***OWL Lite***, The least expressive of the OWL species. Compared with RDFS it adds local range restrictions, existential restrictions, simple cardinality restrictions, equality, and various types of properties (inverse, transitive, and symmetric).

· ***OWL DL,*** Compared with OWL Lite, OWL DL adds full support for (classical) negation, disjunction, cardinality restrictions, enumerations, and value restrictions. The element "DL" comes from the resemblance to an expressive description logic language.

· ***OWL Full,*** Whereas OWL Lite and OWL DL impose restrictions on the use of vocabulary and the use of RDF statements. OWL Full does not have such restrictions. Therefore, OWL Full allows both the specification of classes-as-instances and the use of language constructs in the language itself, which thereby modifies the language.

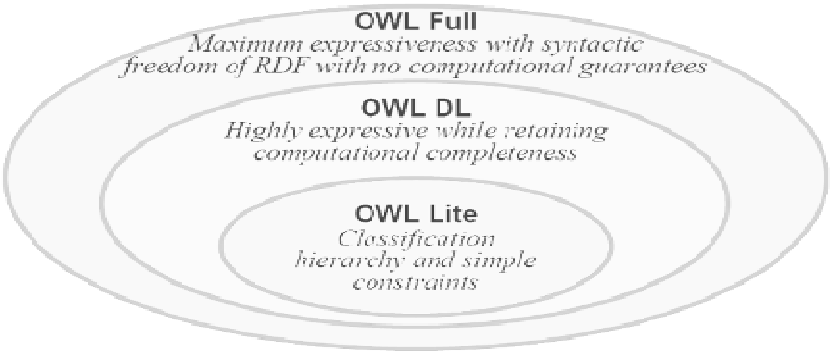

Figure 2-5: OWL Species

2.12. UML Profiles for Ontology Modeling

Since UML is the wide used specification for modeling in general, it would be better if we could use it in ontology modeling rather than OWL. UML does not have ontology concepts, but we can use its profile mechanism to adapt UML to represent ontologies. UML profile is a concept used for adapting the basic UML constructs to a specific purpose. Essentially, this means introducing new kinds of modeling elements by extending the basic ones, and adding the new elements to the modeler's repertoire of tools. In addition, free-form information can be attached to the new modeling elements. The Ontology UML Profile extends UML in a standard way to enable ontology modeling in the widely used UML modeling tools.

One of wide used profiles for modeling ontology is the Ontology Definition Metamodel (ODM), which is The OMG adopted specification (ODM, 2007) contains a formal specification of UML profile for RDFS and OWL. ***ODM*** enables the usage of Model Driven Architecture (MDA) standards in ontological engineering.

2.13. Ontology Engineering Tools

Ontology editors are applications designed to assist in the creation or manipulation of ontologies. They often express ontologies in one of many ontology languages. Some provide export to other ontology languages. In following sub sections, some of these editors are explained.

2.13.1. TopBraid Composer

TopBraid Composer (COMPOSER, 2007) is a visual modeling environment from industry experts for creating and managing domain models and ontologies in the Semantic Web standards: RDF, RDFS and OWL. Composer is an ontology editor and knowledge-base framework that provides visual editing support as well as interoperability with UML, XML Schema and databases. Topbraid Composer is based on the Eclipse platform and the Jena API (w3c, 2007). Composer seamlessly integrates logical and rule-based reasoning engines. It offers a convenient drag-and-

drop, form-based user interface with the ability to view and edit ontologies in a variety of serialization formats. Testing, consistency checking and debugging is supported by built-in OWL Inference engine, SPARQL query engine and Rules engine

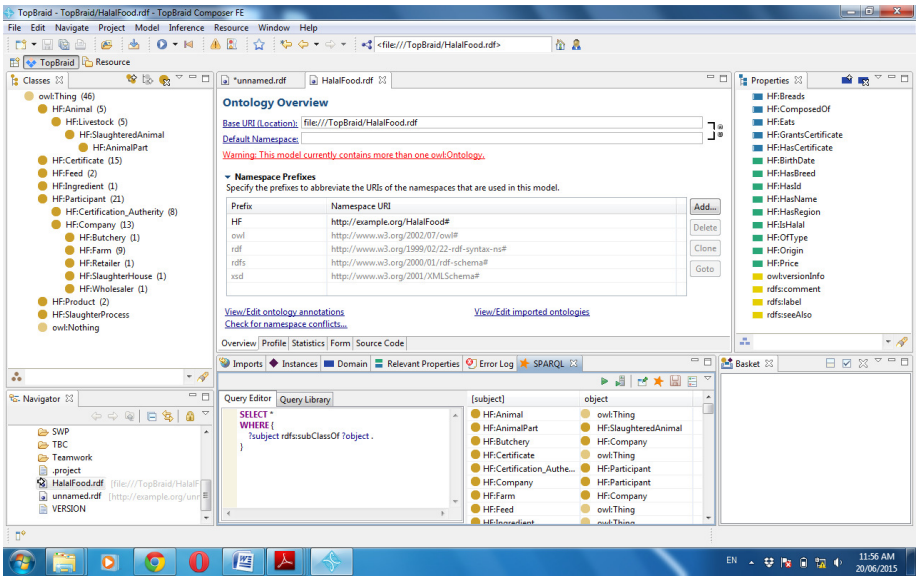

Figure 2-6: TopBraid composer Ontology development tool

2.13.2. Protégé

Protégé is a free, open-source platform that provides a growing user community with a suite of tools to construct domain models and knowledge-base applications with ontologies (Noy et al., 2001). It implements a rich set of knowledge-modeling structures and action that support the creation, visualization and manipulation of ontologies in various representation formats. It can be customized to provide domain-friendly support for creating knowledge models and entering data. Also, it can be extended by a plug-in architecture and Java-based application programming interface (API) for building knowledge-base tools and applications. Protégé allows the definition of classes, class hierarchy's variables, variable-value restrictions, and the relationships between classes and the properties of these relationships.

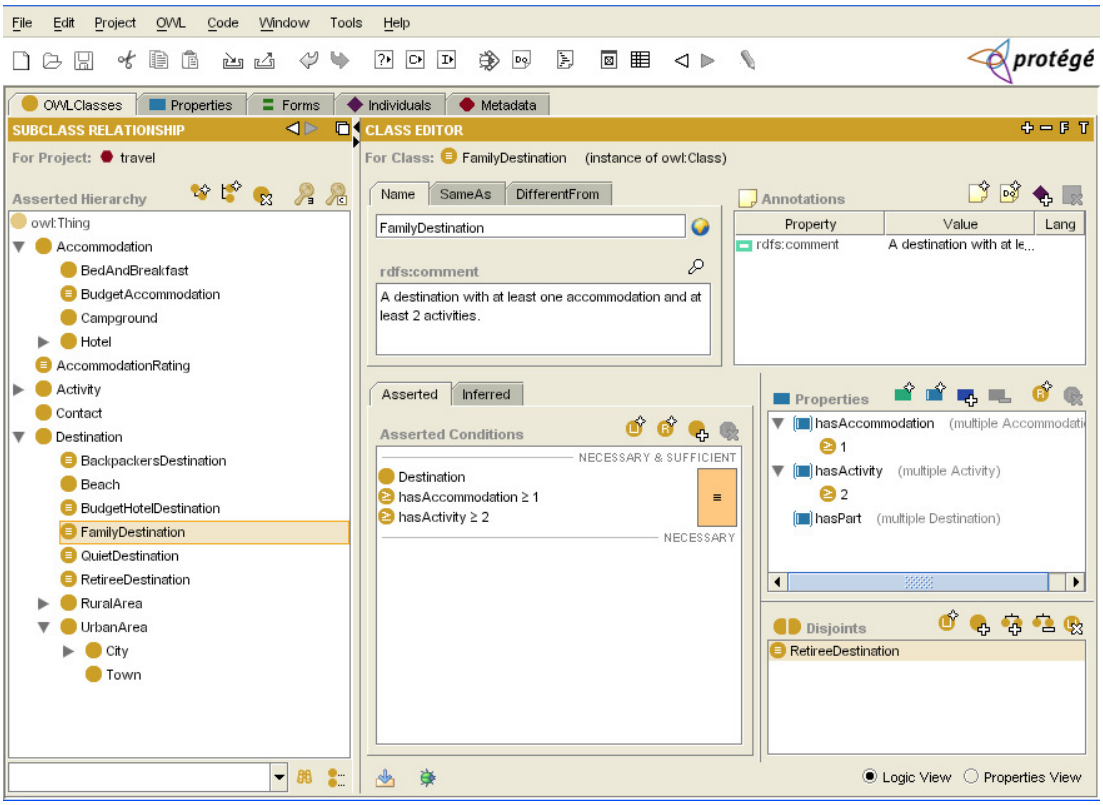

Figure 2-7: Protégé Ontology Editor

2.13.3. Other Tools

Altova Semantic Works (Altova, 2006) is a visual RDF and OWL editor that auto-generates RDF/XML or nTriples based on visual ontology design. No open source version available. ***Amine*** (Kabbaj et al., 2006) is a rather comprehensive, open source platform for the development of intelligent and multi-agent systems written in Java. As one of its components, it has an ontology GUI with text- and tree-based editing modes, with some graph visualization. The ***Apelon DTS*** (Distributed Terminology System) is an integrated set of open source components that provides comprehensive terminology services in distributed application environments. DTS supports national and international data standards, which are a necessary foundation for comparable and interoperable health information, as well as local vocabularies. Typical applications for DTS include clinical data entry, administrative review, problem-list and code-set management, guideline creation, decision support and information retrieval. Though not strictly an ontology management system, Apelon DTS has plug-ins that provide visualization of concept graphs and related functionality that make it close to a complete solution. ***DOME*** is a programmable

XML editor, which is being used in a knowledge extraction role to transform Web pages into RDF, and available as Eclipse plug-ins. DOME stands for DERI Ontology Management Environment. ***FlexViz*** is a Flex-based, Protégé-like client-side ontology creation, management and viewing tool; very impressive.

Finally, ***OntoStudio*** Ontoprise (Weiten, 2009) is based on IBM Eclipse framework. It is an Ontology Engineering Environment supporting the development and maintenance of ontologies by using graphical means. It is based on client/server architecture, where ontologies are managed in a central server and various clients can access and modify these ontologies. It supports multilingual development, and the knowledge model is related to frame-based languages. It supports collaborative development of ontologies. ***OntoStudio*** is built on top of a powerful internal ontology model. The tool allows the user to edit a hierarchy of concepts or classes. The internal representation data model can be exported to DAML+OIL, F-Logic, RDF(S), and OXML.

Most of these tools do not support representation for perdurant ontologies. they only adopt and implement OMG ontology specifications such as RDF(S), OWL (DL, Lite and Full) but these specifications do not support ontology of perdurant, we can exclude OWL-S (will be addressed in next sections), which has the ability to represent services as some sort of ontology perdurants.

2.14. Modeling Ontology of Perdurants:

Ontologies are actually data models, these data models are not just taxonomic hierarchies of entities (classes) and relationships (associations) among them, but also should involve actions (events and processes) which represents entities that happens in time. According to Claudio Masolo (Masolo et al., 2003a) DOLCE, Descriptive Ontology for Linguistic and Cognitive Engineering - upper ontology in its ontology data model it distinguished between two type of entities, An endurant which is an entity that exists in a timeless way, where all of its parts exist at the same time, and a perdurant is an entity that happens in time. The perdurant entity has sub-entities, event and stative, event represents accomplishment and achievement, while stative represents states and processes. Both event and stative not concretely specified although event has many attempts. Another prominent

upper ontology, BBW, Bunge–Wand–Weber (BWW) (Masolo et al., 2003b) system, also recognizes actions according to (Masolo et al., 2003b). A world is composed of things, things have properties, and the collection of property values at a point in time is the state of a thing. An event is a change of state of a thing. The history of a thing is a record of the events involving that thing. The two systems are different, but compatible. Clearly, a BWW event is a DOLCE perdurant. One conclusion Ahmed, M. Nazir et al (Ahmad et al., 2010, Segers et al., 2015) had draw from the combination is that every endurant comes into existence and goes out of existence via a perdurant. A second conclusion is that every perdurant must involve some endurants.

DOLCE divides perdurants into two kinds, events and statives. A DOLCE event is the same as a BWW event. It is an essential whole. All of its parts are necessary. A stative is not an essential whole. The BWW system does not explicitly recognize the concept of stative, although statives are covered by the system.

General Formal Ontology (GFO) also is an upper ontology for conceptual modeling has recognized the above actions using the term occurrence (Herre et al., 2006); GFO distinguishes between persistence through time and being wholly present at a time-boundary. This has produced two GFO categories instead of endurant alone: persistents and presentials. GFO persistent refers to the idea of persistence through time as attributed to DOLCE's endurant, although persistent are not considered in GFO as individuals but as universals . GFO presentials can be generally interpreted as DOLCE endurants, but without temporal extension. Intuitively, DOLCE notion of perdurant corresponds to GFO notion of occurrent. Moreover, it seems that the GFO notions of process, state and change can be interpreted in DOLCE as stative, state and event, respectively. Finally, the GFO categories that concern properties and their values correspond rather well to DOLCE qualities, qualia and quality spaces.

Robert M. Colomb and Mohammad N. Ahmad (Colomb and Ahmad, 2010),present a formal ontology for perdurants suitable for ***IWs*** in the general area of interoperating information systems. Their formal ontology is specialization of the perdurant elements of DOLCE and Bunge-Wand-Weber universal formal ontologies using an abstract material ontology based on the theory of speech acts

embedded in the DEMO method of information system design. In addition, their formal ontology is represented as a UML profile, enabling them to reuse vast structure of the UML. However, in DOLCE upper ontology perdurant entity has sub-element i.e. event and stative, they only discussed event and not touch Stative leaving it for future work, furthermore they discovered that the process is more complex than importing objects from one ontology in another.

2.15. Languages Supports Perdurant Ontology

The literature shows that there are a few ontology languages supporting ontology of perdurants, in the following sub-sections we will Address OWL-S as an extension of OWL for representing services, and DEMO profile.

2.16. Web Ontology Language for Services OWL-S

OWL-S ontology (Martin et al., 2004), built on top of Web Ontology Language – OWL, for describing Semantic Web Services. OWL-S organizes a service description into four conceptual areas: the ***process model***, the ***profile***, the ***grounding***, and the ***service***, in our case study we will concentrate on process models.

OWL-S process model distinguishes between three types of processes: ***atomic***, ***simple*** and ***composite***. For a ***composite*** process, the process model shows how it breaks down into simpler component processes, and the flow of control and data between them. ***Atomic*** processes are essentially ``black boxes'' of functionality, and ***simple*** processes are abstract process descriptions that can relate to other composite or atomic processes.

2.17. Problems with OWL-S

During executing ***Halal food case study*** in Chapter 7, we noticed the following some problems prevent using of OWL-s as a specification for representing perdurant ontologies:

- OWL-s does not provide representation for perdurant instance, it provide a description for web services, their groundings, model and WSDL files.

- OWL-s represent services as black box without showing service tasks, for example, owl-s ***Process Model*** does not show ***Process*** details – speech acts and task status, and it treats it as black box specifying its input and output and pre-conditions.
- OWL-S takes a service point of view to describe service activities, so we argue it works better for a workflow of grate services.
- Owl-s oriented to Services in service-oriented architecture rather than representing perdurant ontology, thus it is not suitable for modeling perdurant ontology.

2.18. DEMO profile

DEMO profile, was presented by Mohammad Nazir and Robert Colomb 2010 (Colomb and Ahmad, 2010). **DEMO** profile depends on Dietz's theory of DEMO (Dietz, 1999) and the theory of speech act which was coined by John Searle (Searle, 1995). Figure 2.8 depicts the profile's stereotypes and UML ***meta-classes***.

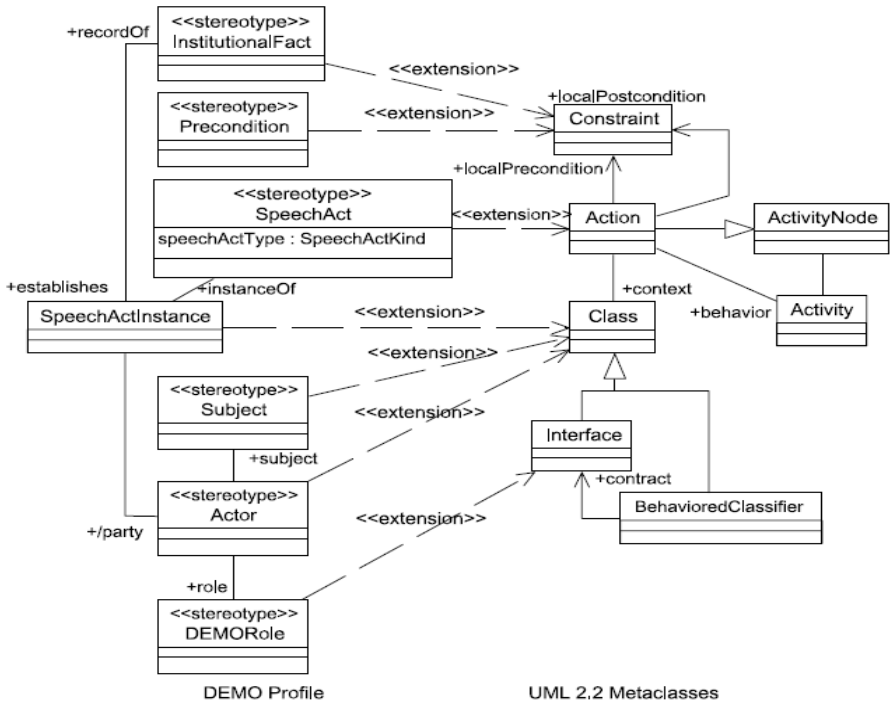

Figure 2-8: DEMO profile (Ahmad et al., 2010)

DEMO profile was well formed, it has the capability of representing perdurant ontologies, and we can have instances of speech act via the ***speech act instance's***

specification**.** In interoperability, semantics are key point for IWs' participant to understand each other. We argue that DEMO does not provide enough semantics for IWs. For instance a speech act should be more expressive to describe specifically its intention, some speech acts are only provide information, some are making some changes, while others needs response from affected participant. Moreover, the speech act has authority own it and has the responsibility to perform.

2.19. Ontology Server

Since the ontology is a complex information object, it is consisting of thousands of entities and hundreds of relations between them, organized in taxonomy. It needs an information system to manage its complexity. An information system intended to manage ontology is called an ***ontology server***. There are many tens of millions of information systems in use around the world doing everything from helping a business manage its billing to keeping track of the design and parts inventory of large aircraft. An information system intended to manage ontology is called an ontology server. Information systems are generally built around a database core, since the complex information object, being managed needs to be stored and desired parts of it retrieved. The database core of an ontology server is called ***an ontology repository***. Ontology repository is containing all ontology entities and relations as well as process and events.

There are a number of research projects. We will refer to that at Stanford University (Ontolingua , Protégé), (Farquhar et al., 1997) Free University of Brussels (Dogma) and the University of Karlsruhe (KAON). Another example for ontology server is MOS, which was used to discover web services (Ibrahim and Salman, 2015) .Because the field is immature there, is no body of routinely used comprehensive tools based on well-established principles. This Chapter therefore is necessarily somewhat speculative in nature. However, ontology servers are closely related to Computer-Aided Software Engineering (CASE) tools, which are a relatively mature technology. CASE tools are generally used to support the design of a system. Database systems such as Oracle or DB2 are supported by tools which assist in data modeling, often using some variant of the Entity-Relationship system (Colomb, 2007). They will assist in the construction of SQL Data Description Language (DDL) statements based on the conceptual model. They also assist in the

construction of SQL Data Manipulation Language (DML) statements using forms, a Query-By Example (QBE) interface, or perhaps a natural language query interface. Unlike CASE tools, an ontology server is used both during design phases and during the execution of the ontology-based applications (Colomb, 2007).

- At design time, the server assists the ontology engineers in the design and construction of the ontology.
- At commit time, a player wishing to join an exchange needs to commit to the ontology, integrating part of their local conceptual model with at least part of the ontology. The ontology server can assist with this task.
- At run time, an ontology server can perform tasks like mediating the exchange of messages.

Table (2.1) shows some functionalities and requirement of the ontology server at the three stages adopted from (Colomb, 2007)

Table 2-1: Ontology Server Requirements

Lifecycle Stage	Requirements
At design-time	An ontology server should provide tools such as editing tools to enable ontology engineers to enter, modify, and browse a developing ontology., certify an ontology, Manage Imported Ontology Modules, Abstract Data Types and Meta-properties, Version Control, Publishing,
At commit-time	A player wishing to join an exchange needs to commit to the ontology, integrating part of their local conceptual model with at least part of the ontology. It provides Browsing Services, Find Relevant Fragments of the Ontology, Subscription Services, and Multiple Natural Languages.
At run-time	An ontology server can perform tasks like Maintain Directories of Players, Roles and Objects. Validate Messages, Broker services, and Archive Services

2.20. Structure of Ontology Server

An ontology server is itself an information system. It is built around a database containing the ontology together with other information needed for it to perform its services. It will therefore need a conceptual model from which its database schemas

will be designed. Since its major content will be the ontology itself, it will make sense for the ontology server's conceptual model to be expressed in one of the systems already used to model the abstract syntax of an ontology representation language.

The repository can be implemented on a number of platforms, including relational database, object-oriented database or an RDF triple store. The repository is a database, but its design is conditioned not so much by size as with conventional information systems, but by complexity. The database will need not only to execute queries like SQL, but will need to be able to navigate classification hierarchies. It will also need a reasoning capability to support some of the server's functions. This can include graph-processing capabilities for navigating the ontology, the description logics capability of testing whether a subclass definition predicate is consistent (satisfiability) and whether one subclass definition predicate subsumes another, and perhaps even a full predicate calculus theorem proved. Finally, the ontology server will need to be able to translate to and from transport representations like XML in order to be able to send and receive complex structures (Colomb, 2007).

2.21. Summary

This chapter has discussed ontology definitions from different perspectives, and has given a sufficient background and history of the ontology in philosophy and computer science. Furthermore, it discussed ontology engineering, ontology representation languages, like OWL, RDF, RDFS, and a like, and some ontology development tools such as protégé, and Topbraid composer. Ontology server and its functionalities in design, commit, and rum time beside perdurant ontology are illustrated too.

CHAPTER 3

INTERLOCKING INSTITUTIONAL WOLRLDS

3.1. Introduction

In order to exchange information, organizations need to interoperate. In interoperation, messages would be passed forward and backward between interoperating systems, these messages have different purposes and conveying different meanings such as querying, requesting submitting and so forth. With a simple message a collection of actions might be performed. In the context of the study, these messages will be named as speech act. This chapter focuses mainly on illustrating the basic concepts of the **speech acts theory,** which will be used intensively in the few following chapters to establish the proposed POUP profile? Furthermore, this chapter highlights institutional facts, IWs, and IWs' Integrity.

3.2. Speech Act Definitions:

Robert Colomb defined the speech act as" *A speech act is something that is said which changes how the world is"*. This definition shows that a speech act has an effect on the surrounding worlds although it is only an utterance of words.

"*almost any speech act is really the performance of several acts at once, distinguished by different aspects of the speaker's intention: there is the act of saying something, what one does in saying it, such as requesting or promising, and how one is trying to affect one's audience*" stated (Bach and Harnish, 1979) defining the speech act.

Dictionary.com defined the term speech act as " *Any of the acts that may be performed by a speaker in making an utterance as stating, asking, requesting, advising, warning, or persuading, considered in term of the content of the message, the intention of the speaker, and the effect on the listener*" (House, 2015).

A speech act is an utterance that serves a function in communication. One performs speech acts when he offers an apology, greeting, request, complaint, invitation, compliment, or refusal. A speech act might contain just one word, as in "*Sorry*!" to perform an apology, or several words or sentences: "*I'm sorry I forgot your birthday. I just let it slip my mind*." Speech acts include **real-life interactions** and require not only knowledge of the language but also appropriate use of that language within a given culture.

3.3. Historical Background

Although some of the basic concepts of Speech Act Theory can be found in earlier philosophers, J. L. Austin and John Searle are credited with its full development. Speech Act Theory is concerned not just with the literal meaning of a sentence but with what kinds of acts derive from it. Some of these include ordering, promising, requesting, informing, and apologizing. An example of a speech act analysis might involve a minister saying at the end of a marriage ceremony: "I now pronounce you man and wife." The sentence, according to Austin and Searle, has three functions: locutionary, illocutionary, and per-locutionary. The locutionary function is saying the actual words, the illocutionary does something (it legally recognizes the couple's relationship), and the per-locutionary expresses the psychological consequences of what is said (in this case, a higher level of commitment and intimacy). Where most philosophers of language had examined the denotative meaning of words and the logic of propositions, Speech Act Theorists focus on connotations and the instrumentality of language, often to be inferred from tone, context, etc. For example, "You're taking the garbage out?" can be merely a question, an order, or a reprimand for past negligence. Whether a speech act succeeds depends on whether the listener understands the speakers' intended meaning. Speech acts can include non-verbal as well as verbal communication: slapping someone on the back can be an act of aggression or of congratulation.

To illustrate that Speech Act Theory actually has relevance beyond academia, I encourage attendees to join me in this exercise. In 2008, after winning the Iowa Caucus, Barack Obama gave what became known as the "Moment Speech". In it he drew attention to the historical significance of his victory and what it can mean to Americans if he becomes President. On the blog docuharma.com, a contributor

whose username is Lithium Cola (I have no idea how this name was chosen) uses Speech Act Theory to explain the per-locutionary power, and covert political strategy, behind the speech.

3.4. Parts of speech acts

The pragmatic dimension of human communication has been studied and conceptualized within speech act theory (Austin and Urmson, 1962) (Searle, 1969) Habermas, 1984). A 'language action' perspective on information systems has been articulated by several scholars, taking their main inspiration from speech act theory (e.g., Winograd & Flores, 1986; Goldkuhl & Lyytinen, 1982; Dietz, 1994). The fundamental speech act thesis is that communication is to be seen as one kind of action.

John Searle (1969) distinguishes between four different sub-acts of a speech act: ***Utterance a****ct*, ***Propositional act***, ***Illocutionary act*** and ***Per-locutionary act***. The *utterance act* is to be seen as the production of a sequence of words that together form a comprehensible wholeness of an utterance. We understand this mainly as equivalent to the syntactic level. The *propositional act* means referring and predicating, i.e., representing a world talked about in an utterance. This corresponds to the semantic level. The *illocutionary act* is what we are doing by speaking, for example, stating, commanding, promising or declaring. The *per-locutionary act* is the intentional 'causing' of effects in listeners. The relationships between these two last sub-acts and the semiotic aspects are not straightforward. To distinguish between utterance, proposition, illocution and per-locution seems to be important. We think, however, it is misleading to describe these as different sub-acts performed within a speech act.

3.5. Types of Speech Acts

Searle (Searle, 1985) enumerated five possible categories of speech acts. ***Assertive,*** speech acts that commit a speaker to the truth of the expressed proposition, e.g. reciting a creed. ***Directive is*** speech acts that cause the hearer to take a particular action, e.g. requests, commands and advice. ***Commissives*** is speech acts that commit a speaker to some future action, e.g. promises and oaths. ***Expressives***,

speech acts that express the speaker's attitudes and emotions towards the proposition, e.g. congratulations, excuses and thanks ***Declaratives***, which brings about a correspondence between the propositional content of the statement and reality. Figure (4.1) shows speech acts types; the two bottom types are stronger they can change they can direct the hearer to do something. While the three are provide hearer with information

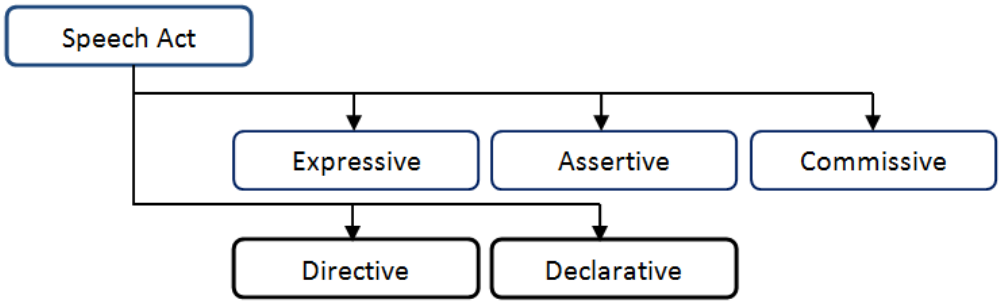

Figure 3-1: Speech Act and its Types

3.6. Usage Of Speech Acts In Technology

Speech acts are used widely in computer science and information technology. It has applications in *Distributed Computing*, *Distributed Artificial* Intelligence, *Natural Language Processing*, and *Electronic Data Interchange protocols.*

In multi-agent systems sometimes, use speech act labels to express the intent of an agent when it sends a message to another agent. For example the intent ***inform*** in the message ***inform*** (content) may be interpreted as a request that the receiving agent adds the item ***content*** to its knowledge base. This is in contrast to the message ***query*** (content) which may be interpreted depending on the semantics employed as a request to see if the item content is currently in the receiving agents' knowledge base.

3.7. Brute Facts and Institutional Facts

John Searle in his famous work "***the construction of social reality***" (Searle, 1995) stated that there are portions of the real world, objective facts in the world, that are only facts by human agreement, fact like money, property, government, and marriage, he called them institutional facts. A brute fact is about something in the

physical world that is independent of human society such things as rivers, trees, and mountains. Searle uses a formula "(**brute fact**) X counts as (**institutional fact**) Y in ***context*** C" to organize the relationship. In our example, the brute fact X is the truck dumping the compost in the driveway. The institutional fact Y is the delivery of an order. The context C in this case is your previously having placed an order for that amount of compost.

3.8. Institutional Worlds

In general, institution is an organization, establishment, foundation, society or the like, devoted to the promotion of a particular cause or program, especially one dedicated to education, public service, or culture(Dectionary.com, 2015).

For our purposes, we look to the **institutions** as human related organizations. They represent a system of ***speech acts*** and records of ***institutional fact***. we can say that institutions are Information systems that creates a collection of instances of ***institutional fact***s which are created by ***speech acts***, these institutions are called institutional world for example in Halal food supply chain farmers, Butcheries, wholesalers and retailers are institutional worlds.

3.9. Interlocking institutional worlds

When two or more institutional worlds share their systems of institutional facts, then they called **interlocking institutional worlds** and abbreviated **IWs**. One example of IWs is Halal Food Supply chains, which consist of a number of institutional worlds; these institutions share their system of speech. Another example is the Olympics; a large number of institutions are interlocked.

3.10. Interlocking institutional worlds' integrity

Integrity in interlocking institutional insures that all operations in the interlocked system are consistence and performed according to framing rules. In the following sub sections, we will show some approaches for IWs integrity, some are ontology-based while some are non-ontology based.

3.11.Ontology-based Approaches

In (Ye et al., 2008) Yan Ye et al present an approach to developing ontologies of supply chain management (Onto-SCM) as a common semantic model of the SCM domain, their model constructed in a modular way in order to enhance reusability and maintainability. This study focused mainly in providing supply chains with semantics via SCM ontology. This might help in interoperation between supply network partners and help in exchanging information. However, not clearly discussed the problem of how to record, store, and retrieve instances of transactions, its only help in recording institutional facts ignoring recording the speech acts that generate these institutional facts. The study does not extend to cover ***IWs*** to be more common model.

Ali Ahmad and others in (Ahmad et al., 2004) have presented a methodology for constructing a general-purpose ontology for supply chain management along with the resulting ontology. Their general-purpose supply chain management ontology can then be extended into various application areas including supply chain specification, supply chain knowledge management systems, various supply chain models and applications. In addition, this study focusing mainly on endurant ontology for supply chain management, and ignoring the perdurant ontology, which from our point of view is the most important to all supply network partners, and of course, all interlocking institutional worlds in the same domain.

3.12.None Ontology-based Approaches:

Suhaiza Zailnai et al in (Zailani et al., 2010) and (Ibrahim et al., 2015) have discussed the conceptual architecture on Halal traceability and Halal tracking system for Halal food product in Malaysia. Halal food supply chain is an example of interlocking institutional worlds. Their conceptual architecture built around central database with an interface for suppliers and consumers this interface is called traceability system front-end, through the internet suppliers can access all partners' information specifically shared information. In this system there are many issues arise: the conceptual architecture does not built around ontology, this means that the interoperation between supply network partners and therefore the Interlocking institutional worlds lacks of semantics and will include semantic

heterogeneity, on addition, each supplier in the supply chain will keep its information as private data, and this will affect information sharing. Another thing is that all instances of speech acts will not obviously recorded.

3.12. Summary

In this chapter, a detailed description of speech act theory has been presented; the description contains speech act definitions, parts, types, and its usage in technology. Furthermore, it shows the concept of institutional facts, institutional worlds, and interlocking institutional worlds as well as some approaches used in performing integrity of interlocking institutional worlds and their weaknesses.

CHAPTER 4

MODEL-BASED ENGINEERING

4.1. Introduction

Since computer invention up to now, software researchers and developers have been eventuating *abstraction* to help them to program in term of their design intent rather than the underlying computing environment and shield them from the complexity of this environment. This abstraction involve both programming languages and operating systems (platforms) for instance assembly shielded developer from programming with machine language, this one level up in abstraction ascent, up levels include procedural languages such as Basic, Fortran, and C; Object oriented languages such as C++ and Java; visual languages like visual basic. In the other hand platform abstraction started with early operating systems such as OS/360 and Linux which shielded developer from complexity of programming directly to hardware.

Although these early languages and platforms raised the level of abstraction, they still had a distinct ***computing-oriented*** focus. In particular, they provided abstractions of the solution space rather than abstractions of the problem space that express designs in terms of concepts in application domains, such as telecom, aerospace, healthcare, insurance, and biology.

The objectives of this chapter is to discuss the paradigm of model based engineering, it is related approaches such as MDA, MDSE, Ontology modeling and so forth, furthermore it will discuss the basic concepts related to modeling and metamodeling.

4.2. Model Based Engineering

The idea of Model based engineering (MBE) or Model Driven Engineering (MDE) stems from software engineering, and more specifically, from the recent research in

software development. MBE evolved as a paradigm shift from the object-oriented technology, in which the main principle is that *everything is an object;* into the model engineering paradigm, based on the principle that *everything is a model (Bézivin, 2005)*. The object-oriented technology is about classes and objects, and the main relations are *instantiation* (an object is an instance of a class) and *inheritance* (a class inherits from another class). MBE is about models, but it is also about relations between a model and the system under study (which can be a software artifact or a real-world domain), metamodels, and model transformations. Similar to the object-oriented technology, MBE can be characterized by two main relations, namely, *representation* (a model represents a software artifact or real-world domain) and *conformance* (a model conforms to a metamodel). According to Jean-Marie Favre, MBE is a field of system engineering in which the process heavily relies on the use of models and model engineering (Favre, 2004b). In that context, model engineering is considered the disciplined and rationalized production of models (Favre, 2004a).

MBE is a promising approach which addresses the platform complexity, and helping in resolve problems stated in the previous section, model driven engineering technologies combine two main things (Schmidt, 2006):

- ***Domain specific modeling language DSML:*** whose type system formalizes the application structure, behavior, and requirements within particular domain such as online financial services, warehouse management. DSML are described by using metamodels, which defines the relationship among concepts in a domain, and precisely specify key semantics and constraints.
- ***Transformation engines and generators:*** Those analyze certain aspects of models and then synthesize various types of artifacts, such as source code, simulation inputs, XML deployment descriptions, or alternative model representations. The ability to synthesize artifacts from models helps ensure the consistency between application implementations and analysis information associated with functional and QoS requirements captured by models. This automated transformation process is often referred to as "correct-by-construction," as opposed to conventional handcrafted "construct-by-correction" software development processes that are tedious and error prone.

MBE is an umbrella term that subsumes several sub-disciplines (Perisic, 2014) Model-Driven Development (MDD) aka Model-Driven Software Engineering (MDSE), which focuses on software-intensive applications; Model-Based Systems Engineering (MBSE), which focuses on Systems Engineering applications; Business Process Modeling (BPM), which focuses on Business Analysis applications; and finally, Ontology Engineering (OE), which focuses on Knowledge Engineering applications. Figure (4-1) illustrates MBE umbrella and its sub-domains, tailored from (Perisic, 2014).

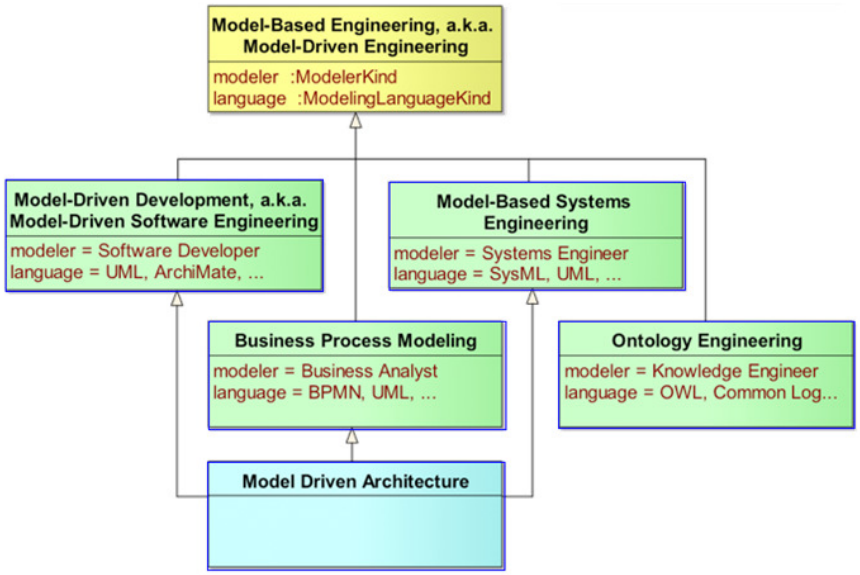

Figure 4-1: Model-Based Engineering and its Sub-domains

4.3. MBE Basic terms

In the few following sub section we will define some basic terms regarding model-based engineering, all sub-categories under the MBE umbrella make use of them

4.3.1. Model

Models play a major role in Model-Based Engineering. The most general definition says that a model is a simplified view of reality (Selic, 2003); or, more formally, a model is a set of statements about a system under study (Seidewitz, 2003). Furthermore, one can say that a model of a system is a description or specification of that system and its environment for some certain purpose. A model is often

presented as a combination of drawings and text. The text may be in a modeling language or in a natural language (OMG, 2003). See Figure 4-2.

In fact, model is a clear set of formal elements that describes something being developed for a specific purpose and that can be analyzed using various methods (Mellor et al. 2003a). In addition to what is specified by the definition of a model, an engineering model must possess, to a sufficient degree, the following five key characteristics (Selic 2003):

- ***Abstraction:*** A model is always a reduced rendering of the system that it represents.
- ***Understandability:*** It is not sufficient just to abstract away detail; we must also present what remains in a form (e.g., a notation) that most directly appeals to our intuition.
- ***Accuracy:*** A model must provide a true-to-life representation of the modeled system's features of interest.
- ***Predictiveness:*** We should be able to use a model to correctly predict the modeled system's interesting but non-obvious properties, either through experimentation (such as by executing a model on a computer) or through some type of formal analysis.
- ***Inexpensiveness:*** A model must be significantly cheaper to construct and analyze than the modeled system.

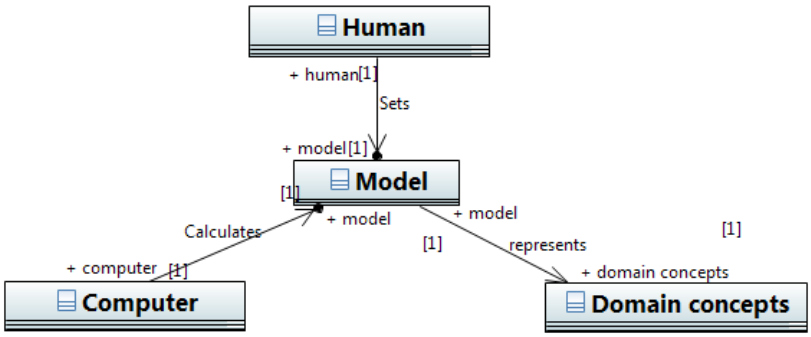

Figure 4-3: modeling domain concepts using models

4.3.2. Metamodel and Metamodeling

Metamodel model is a model of a model, and Metamodeling is the process of generating such Metamodels. **Metamodeling** or **meta-modeling** is the analysis, construction and development of the frames, rules, constraints, models and theories applicable and useful for modeling predefined class of problems. Whereas models describe a specific abstraction of reality, Metamodels define models. *Metamodels* are models of languages used to define models.

4.3.3. Conceptual model

A **conceptual model** is a model made of the composition of concepts, which are used to help people know, understand, or simulate a subject the model represents. Some models are physical objects; for example, a toy model, which may be assembled, and may be made to work like the object it represents.

4.3.4. Conceptual Modeling

John Mylopoulos (Mylopoulos, 1992) defines the discipline of conceptual modeling as" *the activity of formally describing some aspects of the physical and social world around us for purposes of understanding and communication. Conceptual modeling supports structuring and inferential facilities that are psychologically grounded. After all, the descriptions that arise from conceptual modeling activities are intended to be used by humans, not machines... The adequacy of a conceptual modeling notation rests on its contribution to the construction of models of reality that promote a common understanding of that reality among their human users..*"

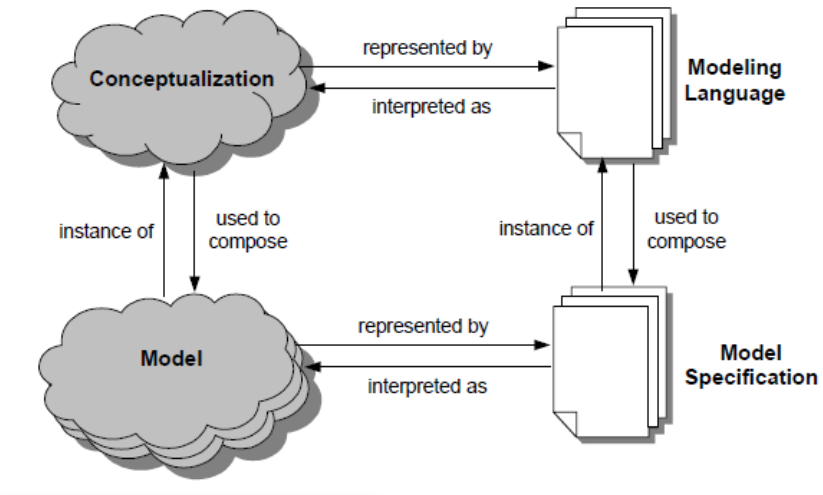

Figure 4-4: conceptualization tailored from (Ahmad, 2009)

4.3.5. Viewpoint

A viewpoint is an abstraction technique for focusing on a particular set of concerns within a system while suppressing all irrelevant detail. A viewpoint can be represented via one or more models.

4.3.6. Model Transformation

Model transformation is the process of converting one model to another within the same system. The transformation combines the platform independent model with additional information to produce a platform specific model.

4.4. Metamodeling Languages

In this section, we define two major Metamodeling languages. The first one is the Meta-Object Facility defined in the scope of the OMG's standards dedicated to the MDA. The second one is the unified modeling language UML, an OMG specification.

4.5. The Meta-Object Facility

The MOF (Omg, 2008)originated as an adaptation of the UML core, which had already gained popularity among software modelers, to the needs of the MDA. The MOF is, essentially, a minimal set of concepts, which can be used to define other

modeling languages. It is similar (but not identical) to the part of UML which is used in the modeling of structure. In the latest version (2.0), the concepts of the MOF and of the UML superstructure are derived from the concepts of the UML infrastructure.

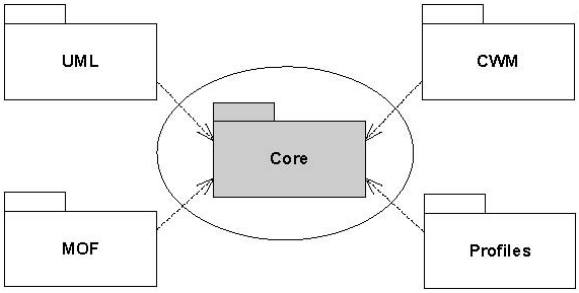

Figure 4-5: dependency of some Metamodels based on the UML core package

Essentially, there is an OMG standard called the UML infrastructure (OMG, 2007), which contains basic concepts that are intended to be used in other Metamodels. Figure 2.5 shows the dependency of some widely used Metamodels based on the UML core package.

4.6. Unified Modeling Language (UML)

The Unified Modeling Language (UML) is a language for specifying, visualizing, and documenting software systems, as well as for modeling business and other non-software systems. UML is a result of the best practice in engineering modeling and has been successfully proven in modeling many big and complex systems.

As we have already mentioned, the UML core is the same as that of the MOF. Accordingly, we shall not discuss its elements; a comprehensive overview can be found in the UML language specification(OMG, 2007). UML is often identified as a graphical notation, which was true for its initial versions. Recently, UML has been recognized more as a language independent of any graphical notation rather than as a graphical notation itself. However, UML is also very important as a language for the graphical representation of models of software systems. The point of view from which a problem under study is examined determines crucially which elements of that problem will be stressed in the final model. UML has features for emphasizing specific views of a model by using graphical representations of models, namely UML diagrams. In this way, we can abstract models; we may otherwise not be able

to analyze and solve complex systems. Evolving from UML 1.x versions, UML2 consists of 13 diagrams, categorized into three main categories: Structure Diagrams which involves class, composite structure, component, package, deployment and object diagrams; Behavioral diagrams which includes Activity Diagram, State Machine Diagram, Use Case Diagram; and Interaction Diagrams: Sequence Diagram, and Communication Diagram, and Timing Diagram. These diagrams provide developers with various perspectives on the system under study or development. A model, which captures a whole system and is graphically represented by diagrams just integrates all these perspectives into one common entity, comprising the union of all modeled details of that system.

4.7. UML Profiles

When we need to define a new language to model a system that either restricts the number of UML elements or adds some constraints or syntactic sugar to them while respecting the original semantics, we do not need to create a new language from scratch using the MOF. Instead, UML can easily be customized by using a set of extension mechanisms that UML itself provides. More precisely, the Profiles package included in UML 2.0 defines a set of UML artifacts that allows the specification of an MOF model to deal with the specific concepts and notation required in particular application domains (e.g., real-time, business process modeling, finance, etc.) or implementation technologies (such as .NET, J2EE, or CORBA). It should be noted that UML Profiles allow the customization of any MOF defined (not just UML defined) metamodel. Similarly, a UML Profile can also specify another UML Profile.

The UML profile is a concept used for adapting the basic UML constructs to a specific purpose. Essentially, this means introducing new kinds of modeling elements by extending the basic ones, and adding them to the modeler's repertoire of tools. In addition, free-form information can be attached to the new modeling elements. We can represent the following MOF model as UML profile.

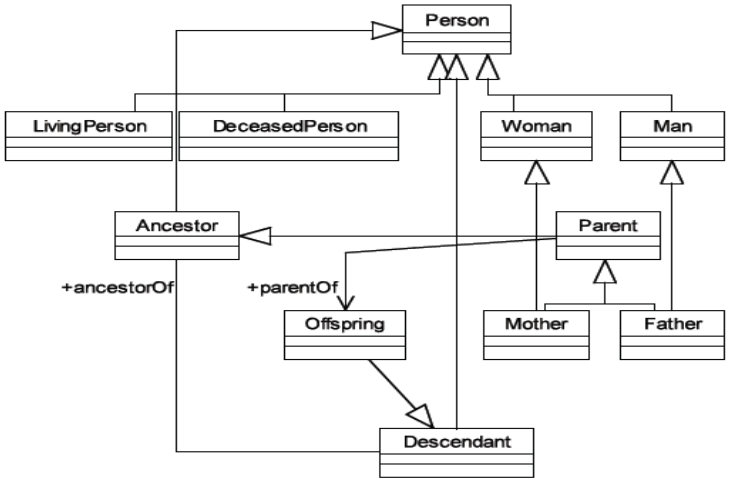

Figure 4-6 Metamodel example

The following figure 5.6 shows the UML profile for Metamodel above, defining three stereotypes Kind, Sub-Kind, Phase and Role respectively extending ***Class Metaclass***

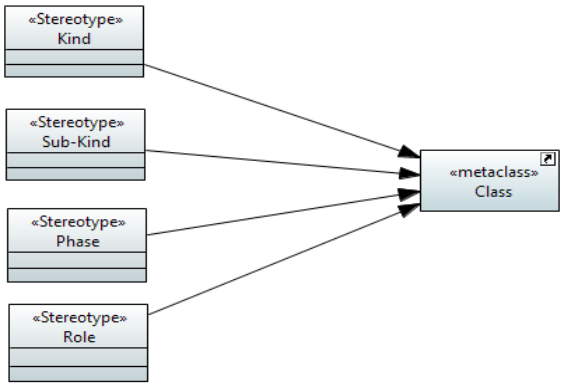

Figure 4-7: the profile

The following model is modeled using a new modeling language described by using UML profile in figure (4.6).

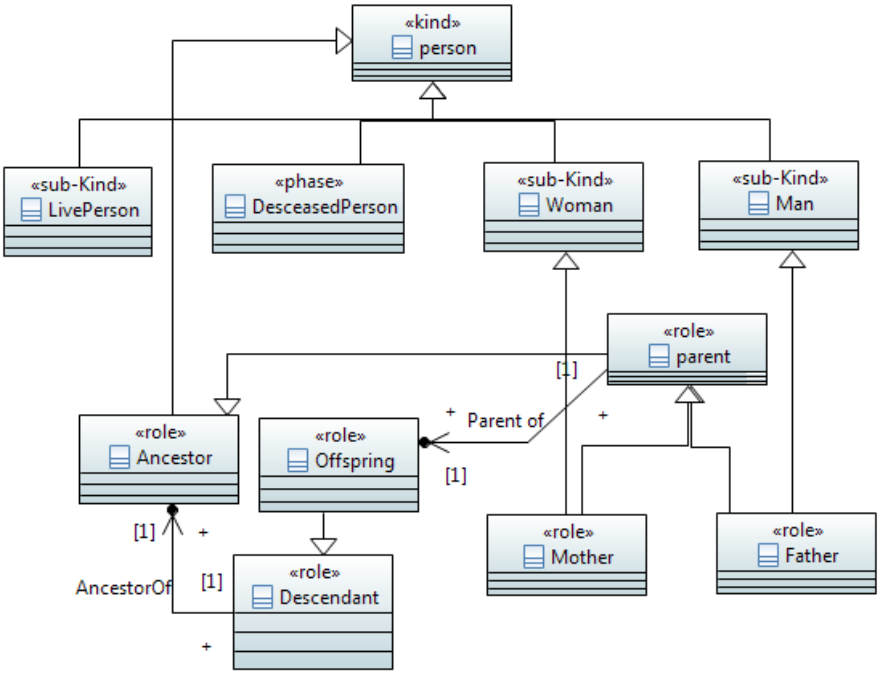

Figure 4-8 the above model using the profile defined above

4.8. Summary:

This chapter demonstrates the state-of-the-art of model based engineering and its sub-domains MDA, MDD, MBSE and ontology modeling, it also discussed modeling and Metamodeling concepts, and common Metamodeling specifications and languages. It gives sufficient theoretical background for MBE as equipment for our proposed approach, following two chapters completes the rest of the literature in this domain.

CHAPTER 5

PROPOSED PERDURANT ONTOLOGY UML PROFILE (POUP)

5.1. Introduction

In chapter Three, we have discussed ontologies and their types, upper level and domain ontologies, and why they are important in modeling interlocking institutional worlds. Upper ontologies describe what is fundamental in the totality of what exists. It defines the most general categories to which we need to refer in constructing a description of reality, and it tells us how these categories are related. Upper level ontologies are used to facilitate the semantic integration of *domain ontologies* and guide the development of new ontologies. For this purpose, they contain general categories that are applicable across multiple domains. Upper level ontologies usually provide rich definitions and axioms for their categories.

Nearly, most of developed ontologies in practice are represented in languages that are ontologically deficient from the point of view of the established upper ontologies; further research has shown that even these upper ontologies are ontologically deficient (Colomb and Ahmad, 2010). In particular, Guizzardi in (Guizzardi, 2005) present an upper ontology based on a combination of DOLCE and BWW which extends the concept of endurant in a number of ways, including the articulation of the concept of class into a system of kinds of classes including *kind*, *sub-kind*, *phase* and *role*. He argues that use of these more specific concepts avoids a number of anomalies in ontologies, particularly involving the identity of individuals.

A domain ontology (or domain-specific ontology) represents concepts, which belong to part of the world specific domain. In this chapter, we present a UML profile for modeling interlocking institutional worlds domain ontologies, this profile capable of modeling domain ontologies of perdurants entities involved in the interlocking institutional worlds. This profile developed to overcome the problem of

representing perdurant ontologies, since most of ontology development languages are not clearly addressed them. Most basic elements of the profile are conform to DOLCE upper ontology Metamodel after making some modifications, and it also inline with ***DEMO*** profile of (Ahmad et al., 2010).

5.2. Introducing *Gerem* and *Arad* Concepts

Ibn Hazm al-Andalusi in his tractates (Al-Andulisy, 1475) stated that: all things in this world are categorized into two categories of things: things that depend on themselves (based themselves) in its existence and can hold other things, we agree to name this category **Gerem** (essence). Things that does not depend on themselves on their existence and depend on other Gerems to hold it, we agree to name this type ***Arad*** (incidental)**.** To illustrate these concepts, we will take simple examples, a stone is Gerem, but its size, width; height, color and shape are Arads. A human is Gerem, but walking, running, sleeping, eating, and physical and mental characteristic like color, height, honest, gentle; and happy, sad and so forth are Arads.

He also argues that there are three types of Gerems: ***Thick***, ***Suave*** and ***Transparent*** Gerems, the first type prevents other Gerems from dwelt with it in the same place, and you cannot see through it such as stone, wall and a like. The second type you can see through it such as glass and water, and the last does not prevent other Gerems to dwell with it in the same place and you see through it such as air and fog.

Formally, ***Gerem*** is an Arabic term means essence or core, and refers to anything that can stand alone without depending on others to reside in or appear on. In contrast, ***Arad*** depends on others to appear, it always resides in Gerem and dwell in it, for example a white colored wall, the wall represents **Gerem** because it does not need other entity to reside in, it based himself, but the white color is Arad because it depends on the wall to appear, it is never be found alone. One Gerem might have one or more Arads appear with it, for example, a pen has color and shape, a dog has color, and can run, bark, bite and so forth figure (5-2) describe the relationship between Germen and Arad. Arad always resides in Gerem but not vice versa, and Gerem might abide in another germen as the reflexive association shows.

We have used the terms ***Gerem*** and ***Arad*** as they in Arabic pronunciation because we did not find appropriate analogous synonym in English language. There are some terms in English have close meaning to them but not completely describe them.

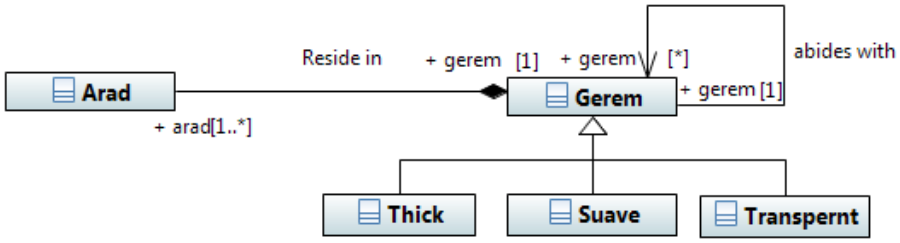

Figure 5-1: Arad and Gerem relationship

5.3. Why Do We Need Arad-Gerem Concepts

It clear that the existence of these new types will make clear distinctions between entities exists in the reality. For example, stone, water, Oxygen all are entities and according To DOLCE, they are Physical Endurants, without giving them more distinctions, but buy using the concept of Gerem we can say that stone is a thick Gerem, water is a Suave Gerem while Oxygen is transparent Gerem. Another example is how to distinguish between White color and the light, we can say that light is transparent Gerem because it could stand alone independent of others, physically it contains of particles and photons. While white color is Arad, because you will never find white color without something to appear with it. Furthermore, if the light disappeared suddenly, white color and all colors will all vanish. Because light compose of colors. We belief this will add value to the IWs interoperability problem.

5.4. DOLCE Analysis Based on Arad-Gerem Concepts

There are many features used in categorizing upper ontology concepts involve: essence and identity, parts and wholes, ***dependence***, composition and constitution, and prosperities and qualities. A special type of dependency is the ***residenes,*** which concern with the ability of some entities to reside inside other entities. Arad always resides in Gerems, some suave Gerems can reside in each other, an obvious example is the salt which has the ability to reside (melt) into the water, and the

daemon and ghost has the ability to reside inside the human body. In this section, we will analyze DOLCE upper ontology according to ***residenes feature,*** and categorizing its main concepts into Arad and Gerem categories.

5.5. DOLCE Upper ontology basic Categories:

Descriptive Ontology for linguistics and Cognitive engineering (DOLCE) is one of famous upper ontologies. One feature of DOLCE upper ontology is its bias to cognitive and linguistics. It is ontology of particulars; it has a number of concepts organized in hierarchy depicted in figure (5-1), many of these classes in the system are based on the part/whole relationship. Entity (Particular) in the top of the hierarchy, which indicate that everything in the system is an entity which they name it ***Particular,*** followed by four main types: Abstract, Quality, Perdurant and Endurant.

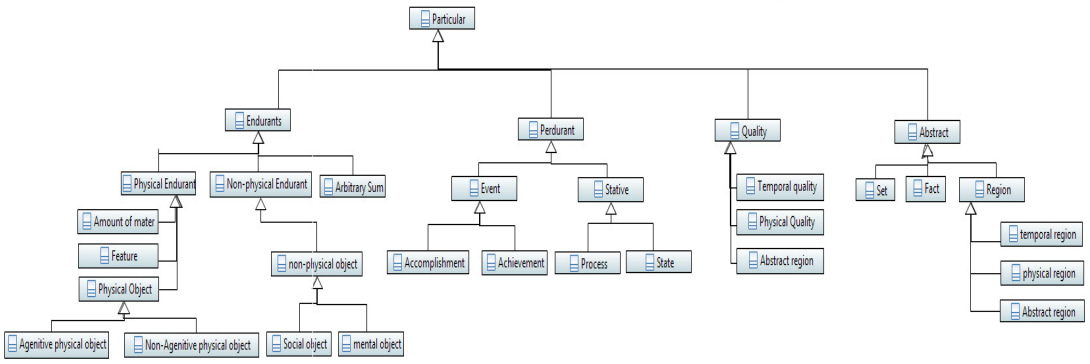

Figure 5-2: DOLCE Upper ontology Categories

Abstracts in rightmost of figure (5-1) are mathematical entities such as facts, sets, and regions. Endurants –entities which exists in time and it can have parts, but all of its parts as at any time are present at that time

In table (5-1), some examples of DOLCE leaf basic category are shown and categorized according to Gerem-Arad concepts. Perdurant, abstract and quality are categorized as **Arad** because they depend on other. While endurant is classified as **Gerem** because it depends on itself and not depend on others to endure.

DOLCE classifies ***features*** under the sub-category ***physical endurant;*** although it looks like Arad rather than Gerem, Example of feature ***a hole, a gulf, an opening, and a boundary***. Boundary and hole or scratch are depend on some Gerems existence, you will never find a scratch of nothing, or boundary alone without existence of something to bind it, a hole is dug always in something, the ground or a wall. From our point of view ***feature*** are not Gerem, furthermore it is not endurant at all, because it is depends in its existence on other Gerem, it exist only if it is Gerem (endurant) exist.

Table 5-1: Categorizing DOLCE According to Arad/Gerem concepts

"Leaf" Basic Category	**Dolce Cat**	**Main-type**	**Examples**
Accomplishment	*Perdurant*	*Arad*	*a conference*, *an ascent*, *a performance*
Achievement	*Perdurant*	*Arad*	*reaching the summit of K2*, *a departure*, *a death*
Process	*Perdurant*	*Arad*	*running*, *writing*
State	*Perdurant*	*Arad*	*being sitting*, *being open*, *being happy*, *being red*
Agentive Physical Object	*Endurant*	*Gerem*	*a human person* (*as opposed to legal person*)
Amount of Matter	*Endurant*	*Gerem*	*some air*, *some gold*, *some cement*
Arbitrary Sum	*Endurant*	*Gerem*	*my left foot and my car*
Feature	***Endurant***	***Arad***	***a hole, a gulf, an opening, a boundary***
Mental Object	*Endurant*	*Gerem*	*a percept*, *a sense datum*
Non-agentive Physical Object	*Endurant*	*Gerem*	*a hammer*, *a house*, *a computer*, *a human body*
Non-agentive Social Object	*Endurant*	*Gerem*	*a law*, *an economic system*, *a currency*, *an asset*
Social Agent	*Endurant*	*Gerem*	*a* (*legal*) *person*, *a contractant*
Society	*Endurant*	*Gerem*	*Fiat*, *Apple*, *the Bank of Italy*
Abstract Quality	*Abstract*	*Arad*	*the value of an asset*
Abstract Region	*Abstract*	*Arad*	*the conventional value of 1 Euro*
Temporal Region	*Abstract*	*Arad*	*the time axis*, *22 june 2002*, *one second*
Physical Quality	*Quality*	*Arad*	*the weight of a pen*, *the color of an apple*
Physical Region	*Quality*	*Arad*	*the physical space*, *an area in the color spectrum*, *80Kg*
Temporal Quality	*Quality*	*Arad*	*the duration of World War I*, *the starting time of the 2000 Olympics*

Figure (5-3) depicts DOLCE upper ontology main concepts classifies into two main categories' Arad and Gerem.

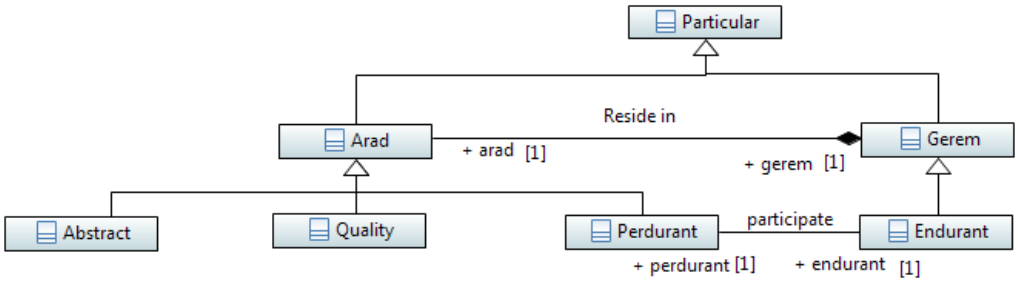

Figure 5-3: Ontology main concepts with Arad and Gerem categories

5.6. Speech Act As Perdurants

A speech act in linguistics and the philosophy of language is an utterance that has performative function in language and communication. According to Kent Bach, "*almost any speech act is really the performance of several acts at once, distinguished by different aspects of the speaker's intention: there is the act of saying something, what one does in saying it, such as requesting or promising, and how one is trying to affect one's audience*." The term was introduced by J.L. Austin (Austin, 1975) and further developed by J.R. Searle(Searle, 1995), Robert M. Colomb (Colomb, 2007) distinguishes between two types of speech acts, ***informative*** and ***performative*** speech acts, refer to chapter two for further information.

Speech acts are not recognized by the DOLCE upper ontology, it only introduced two main types of perdurants ***events*** and ***stative,*** but speech acts are classified as subclass of events, each speech act is an event but not every event is speech act. An **event** is an occurrence, which is an essential whole, has two subtypes ***Achievement*** an event that is atomic and ***Accomplishment*** an event that is not atomic. The second type, **Stative,** which is not an essential whole and has two types ***State***: all of whose temporal parts are of the same type as the Stative itself, and ***Process***: not all of whose temporal parts are of the same type.

Neither of DOLCE's event nor Stative represent exactly the speech act, because the speech act is something ***said*** which changes how the world is. The term “said” is used in a very general sense (in information system context might be sending a query, message, command etc). Examples of speech acts: buying, selling, being inaugurated President of the USA, getting married, getting divorced, being given a

name, winning or losing a contest, being hired or fired, enrolling in a university program, being awarded grades, earning a degree, graduating. Some of these have temporal parts. In particular, earning a degree has parts including enrolling, being awarded grades and graduating, none of which are earning a degree.

All perdurants in above paragraph are speech acts, a shared thing between these events is that, there is some said (written) to achieve the event, Barak Obama became a president after inauguration taking the presidential "oath of office". A man and a woman do not counter as marriage couple (in i.e. Islamic countries) unless the authorized person (*Maazon*) utters the *marriage contract formula.* A student will not get an academic degree unless the *academic senate* grants it. In Islamic ***Sharia,*** a piece of meat will not consider as *Halal* unless uttering ***basmala*** and ***takbir*** when slaughtering its origin animal.

Searle (Searle, 1985) enumerated five possible categories of speech acts. ***Assertive,*** speech acts that commit a speaker to the truth of the expressed proposition, e.g. reciting a creed. ***Directive is*** speech acts that cause the hearer to take a particular action, e.g. requests, commands and advice. ***Commissives*** is speech acts that commit a speaker to some future action, e.g. promises and oaths. ***Expressives***, speech acts that express the speaker's attitudes and emotions towards the proposition, e.g. congratulations, excuses and thanks ***Declaratives***, which brings about a correspondence between the propositional content of the statement and reality. Figure (5-2) shows speech acts type and categorizations.

Table 5-2: Speech Acts Types

If the speech act is :		Example
Directives	The *authority* wants the *target participant* to do something.	asking, ordering, requesting, inviting, advising, begging , *A teacher ask his students to do the homework today*
Commissives	commit a speaker to some future action	promises and oaths
Expressives	express the speaker's attitudes and emotions towards the proposition	congratulations, excuses and thanks
Assertive	They commit the speaker to something being the case.	Suggesting, putting forward, swearing, boasting, and concluding.
Declaratives	The speech act results in a change in the real world situation. They change the state of the world in an immediate way.	*A boss to his employee:"you are fired", disconnect a node* from the network

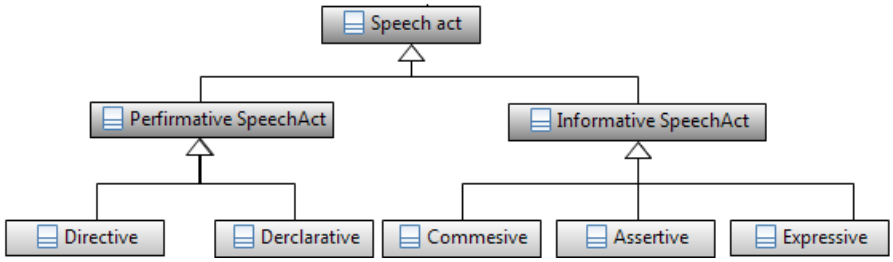

Figure 5-4: Speech Act and its sub-types

5.7. Fitting Speech Acts In DOLCE

Since speech act performs events and action it could be countered as Dolce perdurant, therefore the perdurant hierarchy could be extended to involve speech acts, the upper ontology does not show details, all details will be shown in lower level of abstraction particularly in domain ontologies. The figure below shows the extension of dolce ontology upper ontology by adding speech act concept and its subtypes.

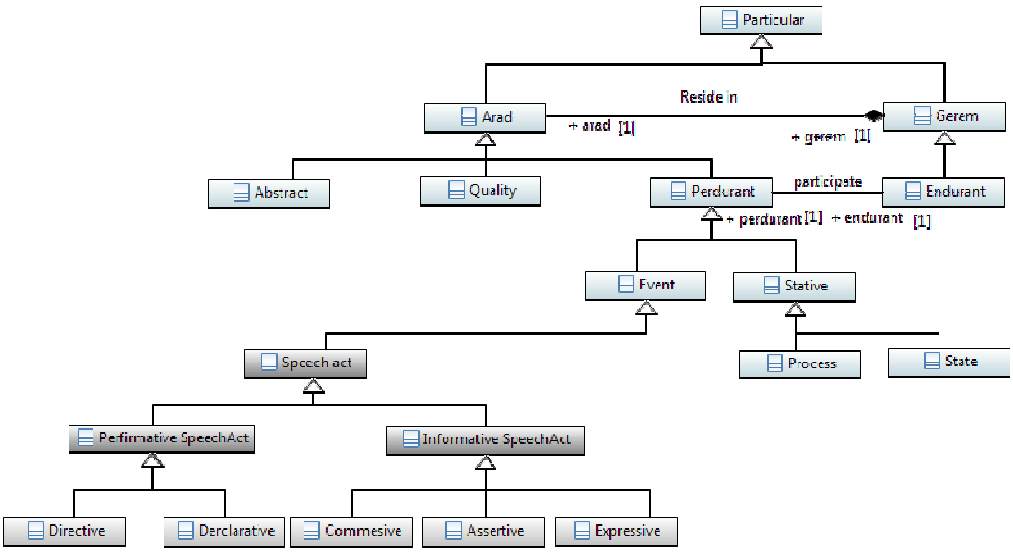

Figure 5-5: Extension of DOLCE Upper Ontology

5.8. Institutional Fact as Endurants:

John Searle in his famous work "***the construction of social reality***" (Searle, 1995) argued that there are portions of the real world, objective facts in the world, that are

only facts by human agreement, fact like money, property, government, and marriage, he called them institutional facts. An institutional fact is a special kind of endurants, it is an element of social reality as distinct from physical reality, and they are created, destroyed, and modified by instances of ***speech acts***, speech acts themselves are invoked by an authority. Diagram in figure (5-6) shows the relationship between speech act and the institutional facts.

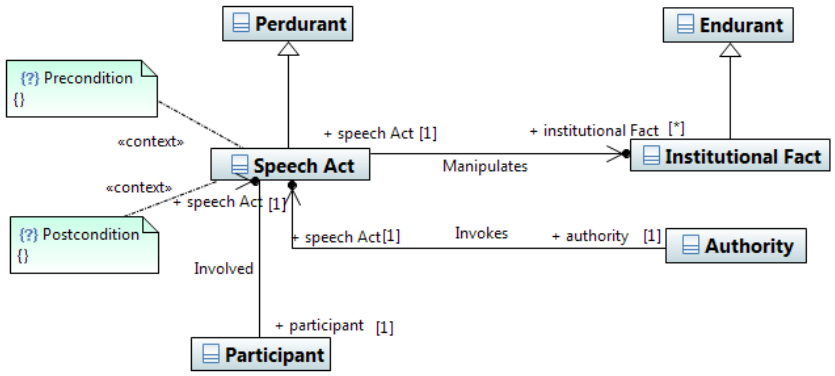

Figure 5-6: Relationship between Speech Act and the Institutional Facts.

Searle distinguishes between ***brute*** physical facts and mental facts. Brute physical facts include such things as rivers, trees, and mountains similar to ***physical objects*** of physical endurants in DOLCE upper ontology. Mental facts include such things as perceptions, feelings, and judgments the same as mental objects of non-physical endurants of DOLCE. Mental facts are ultimately caused by physical facts, because mental facts depend for their existence on physiological processes of consciousness. The physiological processes that produce consciousness enable conscious individuals to recognize physical and mental facts. Thus, mental facts are based on physical facts, and both physical and mental facts are required for the construction of social reality as Searle states.

Physical facts (endurants) are objectives, explains Searle, *"but social facts may be both subjective and objective. Brute physical facts do not depend on our attitudes toward them. For example, mountains and valleys are brute physical facts, regardless of our attitudes toward them. On the other hand, social facts depend on our attitudes toward them. For example, the value of a five-dollar bill is a social fact that depends on our agreement that a five-dollar bill is worth something".*

"*However*", sealer continues the explanation: "*social facts may become objective if they are commonly accepted, and if they are not a matter of individual preference or opinion. For example, the duty of a policeman to enforce the law may be regarded as an objective social fact. Social facts may be epistemic objective, because they are not merely a matter of individual preference or opinion, but they may be ontologically subjective, because they depend for their existence on being agreed upon as facts".*

Institutional facts are constraints, laws according to Searle's comments are commonly accepted social object, and hence objective, but work with the spirit of the law will be subjective. Institutional facts are social facts that are commonly accepted in specific domain; therefore, it could be treated as sub-type of social objects.

5.9. Fitting Institutional Facts In DOLCE

DOLCE classified endurants into three main categories *physical endurants*, non-physical endurants and arbitrary some; As Searle stated, institutional facts are element of social we suppose to fit it within the social object category as the figure (5-7) depicts.

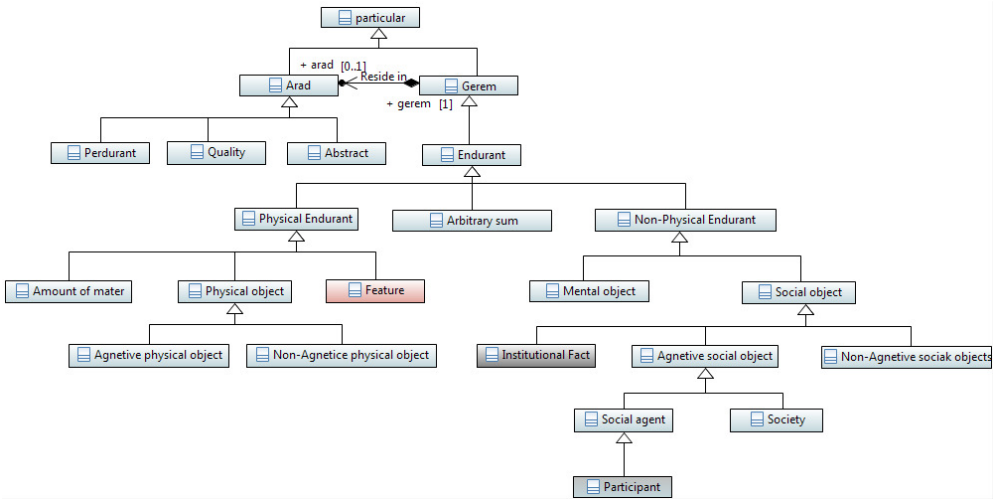

Figure 5-7: Institutional Fact within the Social Object Category

5.10. IWS Perdurant Ontology UML Profile

In previous section we have illustrate some terms related to our project, speech act term with its subclasses performatives and informatives, we will benefit from subclasses of them to enrich the interoperability between interlocking institutional worlds. When two information systems interoperate, they share their worlds of speech acts and institutional facts. We will use these term in building our UML perdurant profile.

5.11. Why Perdurant Ontology

In order for ontology to be used in a computing application, it must be represented as some sort of computer-readable data structure. In OMG, terminology ontology is an example of a data model. The syntactic rules for representing this data structure are called a Metamodel. Therefore, in order to develop ontology at all, there needs to be a Metamodel for it.

Interlocking institutional worlds need to share their data and business processes, ontology of endurants provide semantics to data interoperation thus IWs' participants can understand each other. Actually, each datum exchanged is an instance of institutional fact. The endurants ontology is a Metamodel of world of institutional fact.

While endurants ontology represents a Metamodel for IWs data – the institutional facts- perdurants ontology represents the Metamodel for events and business process that will occur in the interlocking institutional worlds, because all events which occur within the interlocking institutions is in the context of the IWs then all of them are Speech acts. Ontology of both endurants and perdurants provide a rich semantics for participants of IWs.

5.12. Why UML profiles

The Unified Modeling Language (UML) (OMG, 2000) is becoming a widespread standard modeling language for modeling automated information systems. Chapter Five has discussed UML profiles in more details.

UML was chosen for three main reasons: firstly, the status of UML as ***de facto*** standard modeling language; and secondly, the growing interest in its adoption as a

language for conceptual modeling and ontology representation (OMG, 2003a; Kogut, 2002); lastly, it is an extensible language it can be extended to cover different domains of modeling. UML ***profiles*** are OMG specification for extending UML basic structure for specific domain. UML Profile is a restricted extension mechanism, which defines extensions that can be dynamically added to or retracted from an existing model (extension flexibility), Guarantees that the model will remain consistent with the UML standard, even when extended by one or several profiles.

5.13. Perdurant Ontology UML Profile (POUP):

In this section, we will disclose our proposed UML profile of perdurant ontology is represents an enhancement of DEMO profile developed by Colomb and Nazir in (Ahmad et al., 2010), first we will take a look at basic DEMO system which introduced by Dietz in (Dietz, 1999).

DEMO (Dynamic Essential Modeling of Organizations) is a business process modeling methodology. This methodology offers a conceptual framework that is suited as a starting point and basis for modeling information systems (Gomez et al., 2005).

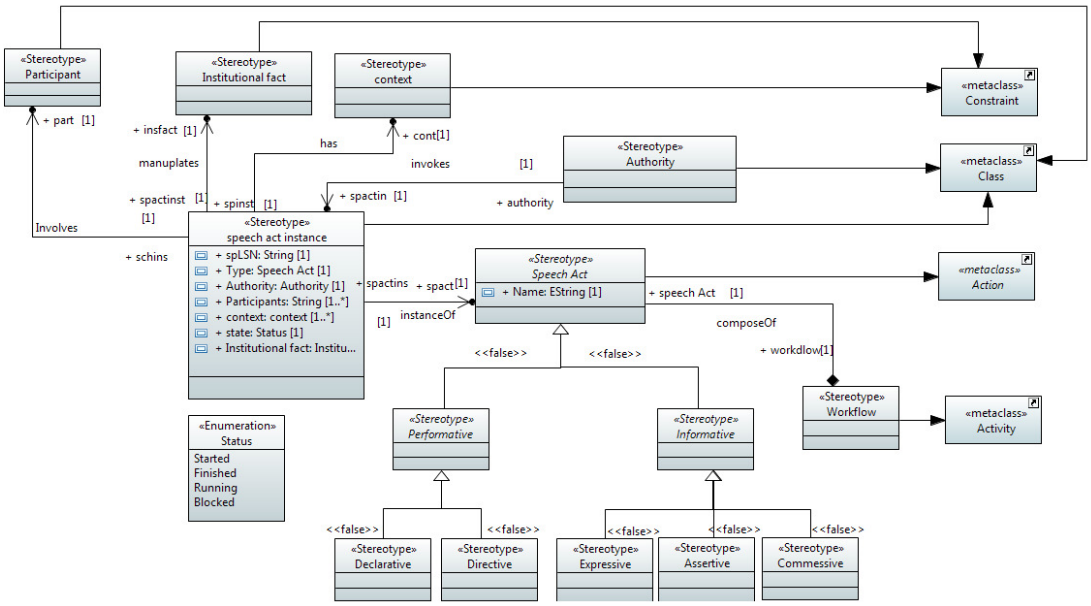

Figure 5-8: the Proposed POUP profile

During interoperating some speech acts are performed, some of them are achieved within one information system, some of them are expanded across different information systems, in interlocking institutional worlds, most of speech acts are cross-boundary and hence known by all participant involved in interoperation. Some of these speech acts perform acts and changes the situation of the surrounding objects, and some of them are just shows the situation, the former known as performative speech acts and the later as informative speech acts .The UML profile consists of the following stereo types:

Table 5-3: POUP Profile Basic Components and their Meta-classes

Stereotype	Metaclass	Description
Speech Act	Action	Abstract Metaclass
Perfomative Act	Action	Abstract Metaclass, subclass of Speech act, not substitutable
Informative Act	Action	Abstract Metaclass, subclass of Speech act, not substitutable
Assertive	Action	subclass of informative Act
Commissives	Action	subclass of informative Act
Directive	Action	subclass of performative Act
Expressive	Action	subclass of informative Act
Context	Constraint	
Authority	Class	
Declarative	Action	subclass of performative Act
Speech act Instance	Class	
Workflow	Activity	

- ***Speech Act***: Abstract class, extends UML **Action** *Metaclass. And it is a generalization for performative Act and Informative Act*, speech act abstract cannot have instances directly, but it can have via its sub-kinds.

- **Speech act instance***: represents an instance specification of speech act, which extend and stereo types UML Metaclass* ***class***

5.14. Running Example:

To show the benefit of using this profile in modeling perdurant ontology, we will take a simple example, paper submission, which starts with submitting the paper and ending with publishing it, activity diagram in figure (5.9) shows all operations and event involved in this activity. There are three participants: Author, Editor, and Reviewer. Background (context) needed for understanding the interoperation

between these participants is that, author is someone who synthesizes publications such as journal papers or books. To publish a paper, the author should submit the paper to authorized editor. The editor checks the paper structure and if it is related to journal domain then they assign reviewers to evaluate it, they notify the editor with decision of either to accept or reject the paper, in case the paper was accepted it would be published in the journal.

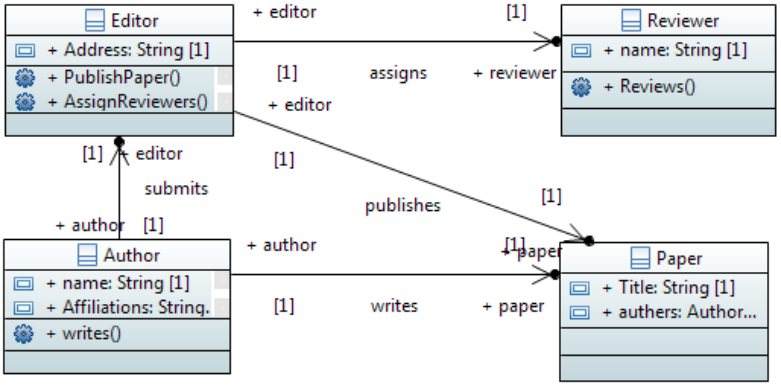

Figure 5-9: An Example of Paper Submission

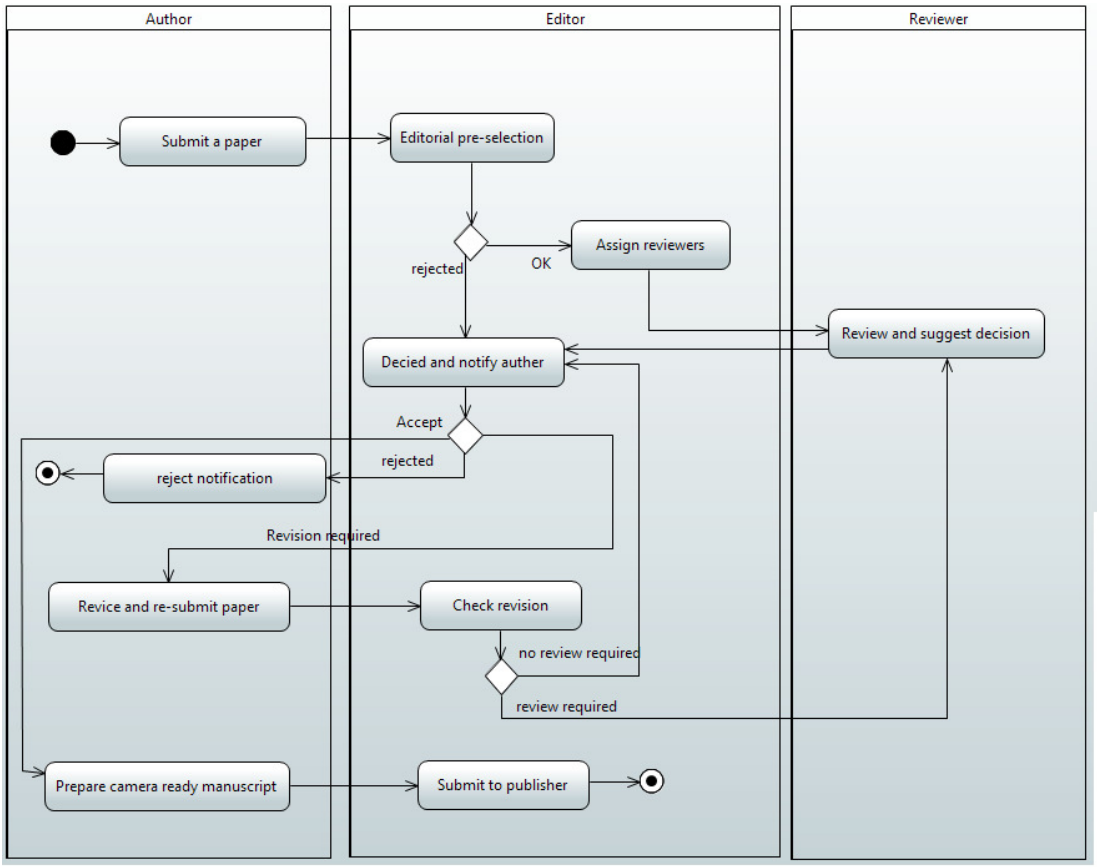

Figure 5-10: Activity Diagram for Paper Submission

Interoperation between above participants (Author, Editor and Reviewers) constantly, achieved via exchanging messages, these messages holds the speech act. As stated before every speech act compose of four sub-acts: Utterance act, Propositional act, Illocutionary act, and Per-locutionary act. Utterance is the production of a sequence of words that together form the message, propositional act is the semantics behind the text uttered, The illocutionary act is what the message do, and finally, pre-locutionary act is the intentional 'causing' of effects in interventionist (who received the message).

Each speech act involves two or more participants; in specific case speech act might involve only one participant. Table (5-4) displays operations (speech acts) of our simple examples and participants involved.

Table 5-4: Operations and Participant Involved

Operation	Participants involved
Submit a paper	Author, Editor
Editorial pre-selection	Editor
Desk rejection	

Assign reviewer	Editor, Reviewer
Review and suggest decision	Reviewer
Decide and notify author	Reviewer, editor, Author
Revise and re-submit paper	Editor, author
Check revision	Editor
Prepare camera ready manuscript	Author, editor
Submit to publisher	Editor (publisher)

Every speech act is produced by an authorized participant, and directed to specific participants (interventionist), for instance, editors are authorized to submit a paper, editors and reviewer are not authorized to do so unless they change their role (reviewer play the role of author). Furthermore every speech act either manipulate (create, update or delete) an institutional fact i.e. performative speech act, or display institutional fact status i.e. informative speech act, both informative and performative speech acts has non-substitutable sub-kind. Table (5-5) shows speech acts their types, authority and interventionist.

Table 5-5: Speech Acts Their Types, Authority and Interventionist

Speech Act	Type	Authority	participants
Submit a paper	Directive	Author	Editor
Editorial pre-selection	Expressive	Editor	Editor
Desk rejection	Declarative	Editor	Author
Assign reviewer	Declarative	Editor	Reviewer
Review and suggest decision	Assertive	reviewer	Editor
Decide and notify author	Expressive	Editor	Editor
Reject notification	Expressive	Editor	Author
Revise and re-submit paper	Directive	Author	Editor
Check revision	Assertive	Editor	Editor
Prepare camera ready manuscript	Directive	Author	Author
Submit to publisher	Declarative	Editor	Editor

Submit a paper speech act is a performative speech act of type ***Directive***, while Desk rejection is performative speech act of type ***Declarative***.

Table 5-6: Speech Act and Institutional Fact with Context

Speech Act	Context		Institutional fact
	Pre-condition	Post-condition	
Submit a paper			unpublished paper
Editorial pre-selection	In scope=yes Published before=no Submitted before=no		considerable paper
Desk rejection	considerable paper=no		Desk rejection notification
Assign reviewer	Considerable paper=yes	Qualified and responsible	Reviewer assigned to paper

	From considerable editor =yes	reviewer=yes	
Review and suggest decision	Considerable paper=yes		Reviewed paper
Decide and notify author			
Reject notification	Rejected paper=yes		Reject notification
Revise and re-submit paper	R&R paper=yes		Revised paper
Check revision			
Prepare camera ready manuscript	Accepted paper=yes		Accepted paper
Submit to publisher	Accepted paper=yes		Published paper

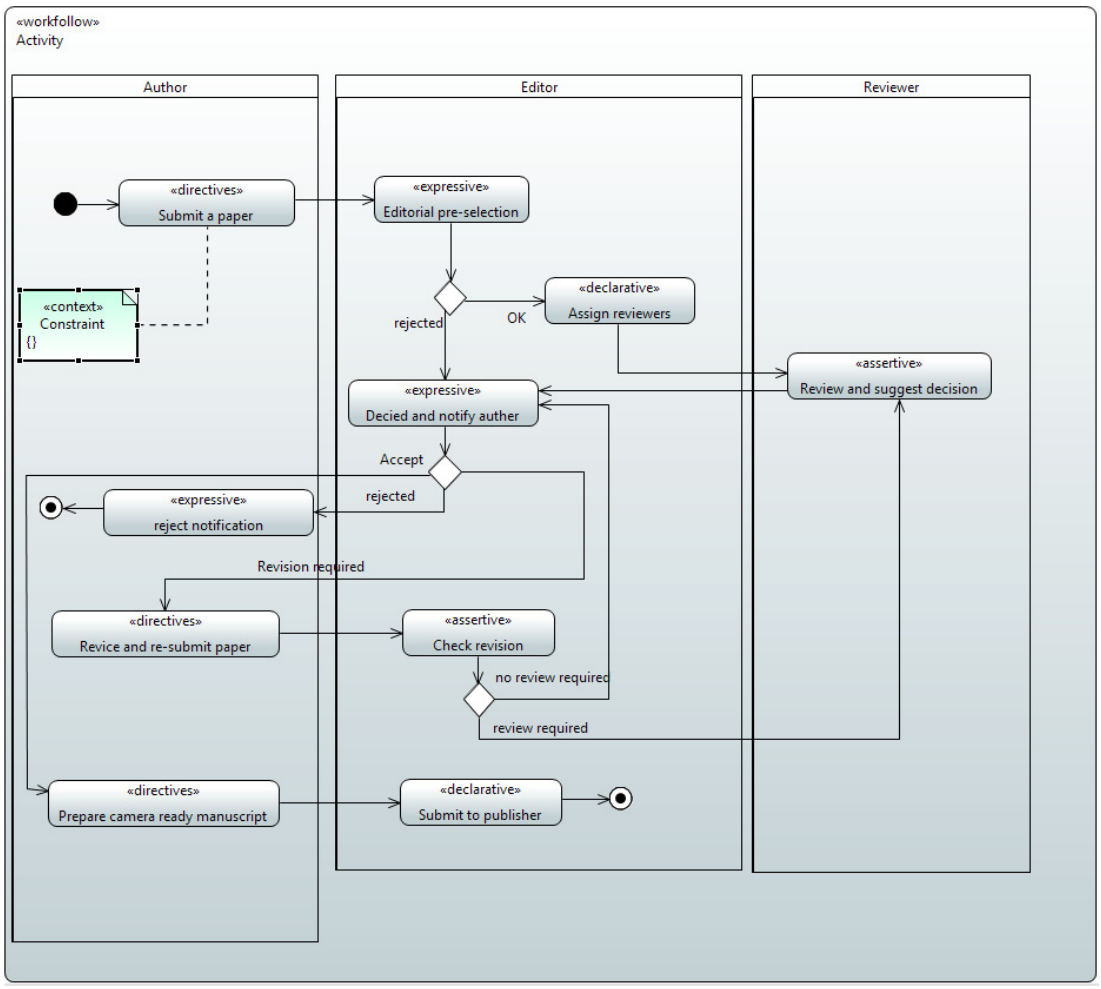

Figure 5-11: Part of Paper Submission Perdurant Ontology

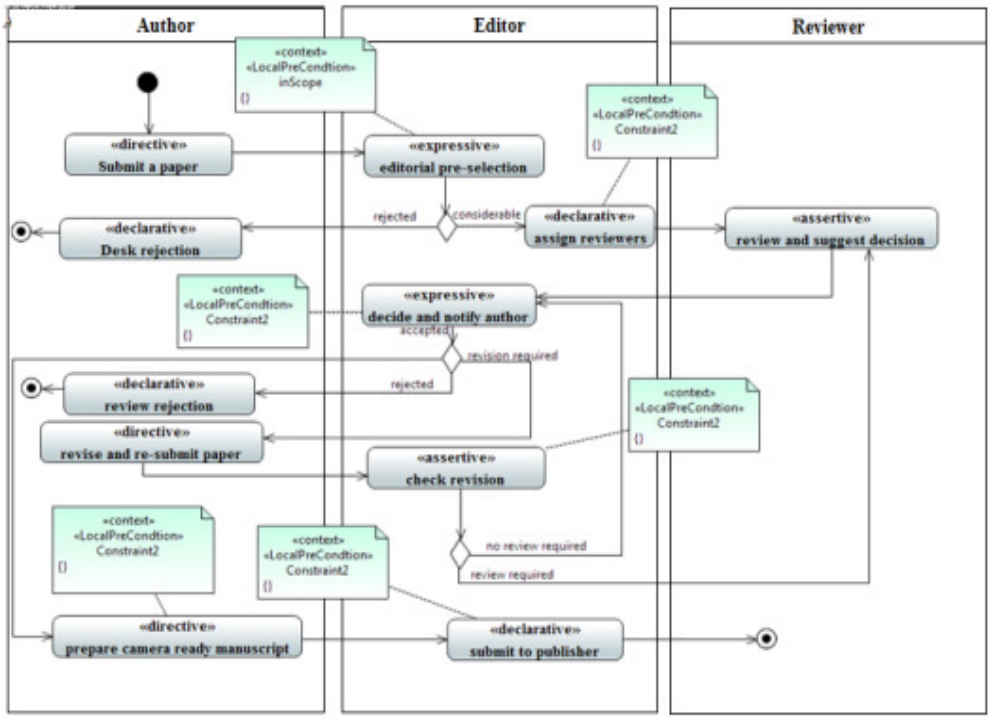

Figure 5-12: More Expressive Ontology Perdurant

5.15. Transaction Log and Speech Act instances:

A transaction log (also transaction journal, database log, binary log or audit trail) is a history of actions executed by a database management system to guarantee ACID properties over crashes or hardware failures. Physically, a log is a file listing changes to the database, stored in a stable storage format. Every SQL database has a transaction log that records all transactions and the database modifications made by each transaction. We will borrow this concept of database field to represent instances of speech acts. Each instance will be recorded in Speech acts log and will be stored in the ontology repository. Figure 5.13 shows snapshot of SQLSERVER transaction log. Some of these fields will be borrowed into our speech act log.

	Current LSN	Transaction ID	Operation	Transaction Name	CONTEXT	AllocUnitName	Page ID	Slot ID	Begin Time	End Time	Numbe
1	00000015:0000001a:0001	0000:0000022f	LOP_BEGIN_XACT	ReadingDBLog	LCX_NULL	NULL	NULL	NULL	2015/05/21 12:47:56:467	NULL	NULL
2	00000015:0000001a:0002	0000:0000022f	LOP_MODIFY_ROW	NULL	LCX_BOOT_PAGE	Unknown Alloc Unit	0001:00000009	0	NULL	NULL	0
3	00000015:0000001a:0003	0000:0000022f	LOP_MODIFY_ROW	NULL	LCX_BOOT_PAGE	Unknown Alloc Unit	0001:00000009	0	NULL	NULL	0
4	00000015:0000001b:0001	0000:00000230	LOP_BEGIN_XACT	Update Sysusers	LCX_NULL	NULL	NULL	NULL	2015/05/21 12:47:56:497	NULL	NULL
5	00000015:0000001b:0004	0000:00000230	LOP_DELETE_ROWS	NULL	LCX_MARK_AS_GHOST	sys.sysowners.nc2	0001:0000005f	3	NULL	NULL	2
6	00000015:0000001b:0006	0000:00000230	LOP_INSERT_ROWS	NULL	LCX_INDEX_LEAF	sys.sysowners.nc2	0001:0000005f	4	NULL	NULL	2
7	00000015:0000001b:0007	0000:00000230	LOP_MODIFY_ROW	NULL	LCX_CLUSTERED	sys.sysowners.clst	0001:0000005b	1	NULL	NULL	2
8	00000015:0000001b:0008	0000:00000230	LOP_COMMIT_XACT	NULL	LCX_NULL	NULL	NULL	NULL	NULL	2015/0...	NULL
9	00000015:0000001d:0001	0000:0000022f	LOP_DELETE_ROWS	NULL	LCX_MARK_AS_GHOST	sys.sysdbfiles.clst	0001:00000055	0	NULL	NULL	2

Figure 5-13: Snapshot of Sqlserver Transaction Log

Suppose that an author (Tom) has submitted a paper to IOS journal via an information system committed to above ontology of both endurant (class diagram) and perdurant (activity diagram), IOS editor received the paper and assigned reviewers according to steps illustrated above. Table (5.7) shows the snapshot of the ***speech acts instance log***, which holds required field for recording speech act instances.

Table 5-7: Snapshot of Speech Acts Instance Log

SPLSN	Speech act instance	Type	Authority	Participant	Context	state	Institutional fact
002001	Paper submitted	Directive	Ibrahim	IEEE Editor		finished	unpublished paper
002002	Pre-selection	Expressive	IEEE Editor	IEEE Editor		finished	Considerable paper
002003	Desk rejection	Declarative	IEEE Editor	Ibrahim		finished	Desk reg. notification
002004	Assign reviewer	Declarative	IEEE Editor	Reviewer		running	Assigned reviewers
002005	Review	Assertive	Reviewer	IEEE Editor		Not started	
002006	Decide and notify	Expressive	IEEE Editor	IEEE Editor		Not started	Reviewed paper
002007	Reject notification	Expressive	IEEE Editor	IEEE Editor		Not started	Reject notification
002008	Revise and re-submit	Directive	Ibrahim	IEEE Editor		Not started	Revised paper
002009	Check revision	Assertive	IEEE Editor	IEEE Editor		Not started	
0020010	Camera ready	Directive	Ibrahim	Ibrahim		Not started	Accepted paper
0020011	Submit to publisher	Declarative	IEEE Editor	IEEE Editor		Not started	Published paper

5.16. Features of POUP Profile

POUP has the following features

- It is capable of representing perdurants (speech acts); therefore it could be used to model the behavior of interlocking institutional worlds.
- It provides a specification for representing instances of transactions, which typified with speech acts. The specification looks like the transaction log in database systems every speech act occur will be recorded in the speech acts log.
- POUP is more expressive, POUP provide rich semantic representations for interlocking institutional worlds behavioral models. It is easy to distinguish between different types of transactions (speech acts). participants involved in IWs

5.17. Summary

In this chapter, we have proposed a UML profile for representing perdurant ontology. It conforms to DOLCE upper ontology and it borrows speech acts and institutional facts concepts from speech acts theory of Searle and Austin. The

profile can represent instances of perdurant via the ***instance*** of ***speech act*** specification**.** An example was conducted to illustrate how this profile could be used, and instances of speech acts are recorded in away like database transaction log. In the following chapter, we will show how ontology server can make use of this profile in querying and inferencing ontology data.

CHAPTER 6

PROPOSED ONTOLOGY SERVER FRAMEWORK

6.1. Introduction

Ontology server is an information system for managing ontologies, it provide many facilities to its participant in query, committing, inferring data from the ontology. In the following sections, we will propose a framework for an ontology server for serving interlocking institutional world's participant to interoperate and exchange information. First, we need to transform models modeled by POUP profile to be represented in OWL, specifically in RDF/XML serialization, and then store files in the repository and constructing the ontology server. The proposed ontology server was composed of semantic web components such as Jena apache, Pellet inference library, and RDF triple stores.

6.2. Models Transformation

Up to now, the profile is ready to be used; instances of workflows will represent a series of speech acts being executed. Instances of speech acts and its related details will be recorded via ***speech act instance's*** specification, which is look like transaction log mechanism in database management systems. Up to now, only the graphical prospective of the perdurant ontology is modeled. The metamodel of the perdurant ontology generates instances of workflow models; which looks like Activity diagrams in core UML but it shows more semantics. In semantic web, ontologies are mostly represented using OWL and RDF(S)/XML specifications. To be conformed to these specifications, we need to transform our instance of perdurant ontology from UML into RDF(s)/XML.

6.3. Transform UML model into RDFs/XML Model:

Model transformation is a key technique used in model-driven architecture (MDA) to transfer one model type to another type. In this section, we will discuss

transformation of models from UML to OWL2. RDF (Resource Description Framework) and RDF Schema (collectively called RDF(S)) are the normative language to describe the Web resource information. In other hand, UML (Unified Modeling Language) is being widely applied to data modeling in many application domains. The Idea is that, we would like to make use of UML modeling language capability to get the benefit of using UML modeling tools, and then transform UML diagrams into RDF(S) or OWL models to get the benefit of using RDF. The RDF model is made up of triples, as such; it can be efficiently implemented and stored. Furthermore, The RDF model has the important property of being modular. The union of knowledge (directed graphs) is mapped into the union of the corresponding RDF structures; this means that information processing can be fully ***parallelized***. Moreover, RDF is represented as XML syntax called RDF/XM and hence it is easy to be applied. How to transform UML diagrams into RDF(S) is a key question in our study, because once we have the RDF/XML version of our ontology then we can easily query RDF graphs to find facts and instances of actions, furthermore we can infer new data based on stored RDF graphs.

Before achieving these transformations, we must compare and analyze the characteristics of both UML and RDF(S). Table (5-8) shows the relative basic component for both UML and RDF/OWL, UML class is represented in RDF with rdfs:Class and UML Attribute are similar to rdf:Property.

Table 6-1: mapping between UML and RDF(s)

UML	RDF(S)
Class	rdfs:Class
Attribute	rdf:Property
Association	rdfs:Class
Association Class	rdfs:Class
Role in Association (Class)	rdf:Property
Generalization	rdfs:subClassOf
Aggregation	rdf:Property rdfs:domain rdfs:range
Dependency	rdf:Property rdfs:domain rdfs:range

There are many problems with transformation from UML to OWL, such as; Abstract classes cannot be transformed into OWL2. If a class is defined as abstract

in UML, no instances of this class (objects) can be created. In contrast, OWL2 has no language feature to specify that a class must not directly contain any individual.

Furthermore, in UML the visibility of model elements can be reduced by marking them as public, private or protected. It is also possible to declare UML model elements as read only, but OWL 2 does not have this kind of control mechanism to restrict the access to model elements.

Moreover, OWL ontologies also do not contain any actions; hence, transformation of activity diagrams to OWL in particular, will be faced with great challenges. Transformations of this kind are out of our study scope and will left for future research. We will use owl: classes to represent action (speech acts) to represent instances of POUP models.

6.4. Transformation of POUP Models into RDFs/XML Model:

By using perdurant ontology profiles (***POUP***), models for describing shared information about actions an event that might occur in interlocking institutional worlds of some domain could be created. For instance Halal food IWs including a series of speech acts for halal food registration, or buy, selling and so forth, this collection of speech acts compose ***Halal food ontology,*** this ontology will be transformed to RDF/XML format as depicted in figure() below.

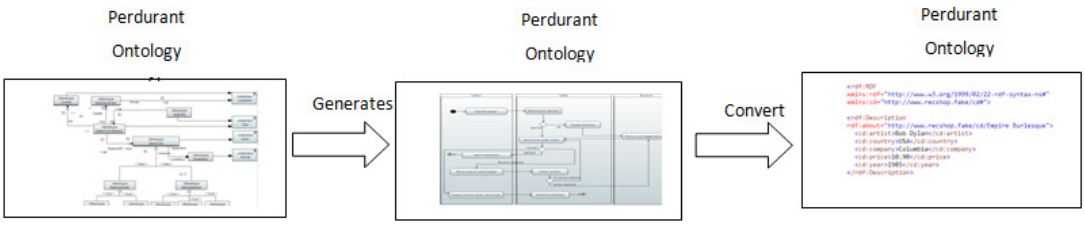

Figure 6-1: Transformation of POUP Models

QVT is the language used to model transformations. To transform instances of POUP, there must be transformation rules, and both source and destination models.

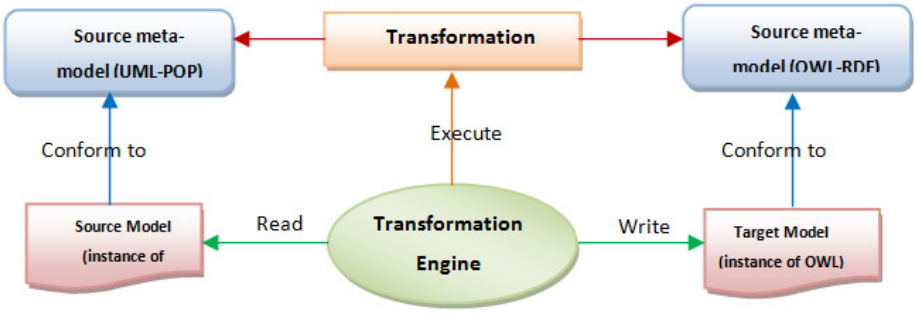

Figure 6-2: transformation to owl

6.5. Transformation Rules

As we stated in section (6.3), activity model transformation is key point for our study, class models easily can be transformed , we will manually mapping POUP instance to OWLs ignoring problems stated above, we will represent actions (speech acts) as owl classes. Table (6-3) shows mapping rules.

An element of Perdurant Ontology UML Profile that will be transformed to OWL or RDF/XML format:

Table 6-2: POUP Elements

Constraints:
▪ Institutional facts ▪ Context
Classes:
▪ Participants ▪ Authority ▪ Speech act instance
Actions:
▪ Speech act (abstract action) ▪ Informative (abstract action) ▪ Informative (abstract action) ▪ Declarative ▪ Directive ▪ Expressive ▪ Assertive ▪ Commissive
Activities
▪ Workflow
Relationships:
▪ ***Invoke*** Domain Authority Range Speech act instance

- ***instanceOf***
  domain: speech act instance
  range: speech act
- ***hasContext***
  domain: speech act instance
  range: context
- ***manipulates***
  domain: speech act instance
  range: institutional facts
- ***composeOf***
  domain: speech act
  range: workflow

For our purposes we will represent POUP element in OWL according to the following mapping table (6-3). Figure 6.3 shows an excerpt from POUP OWL ontology after transformation.

Table 6-3:profile components and RDF corresponding

Perdurant Profile Element	OWL element
POUP: Constraints:	Owl: Restriction
POUP: classes	Owl: classes
POUP: Actions	Owl: classes
POUP: Relationships	RDF: property
POUP: Properties	RDF: property

A number of studies in this area, but most of them are devoted to UML class diagram rather than activity diagrams. Automating transformations of perdurant ontologies to OWL are left for future research.

```
  <owl:Class rdf:ID="Workflow"/>
  <owl:Class rdf:ID="context"/>
  <owl:Class rdf:ID="Declarative">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Performative"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:about="#Performative">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Speech_Act"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Informative">
    <rdfs:subClassOf rdf:resource="#Speech_Act"/>
  </owl:Class>
  <owl:Class rdf:ID="commessive">
    <rdfs:subClassOf rdf:resource="#Informative"/>
  </owl:Class>
  <owl:Class rdf:ID="Participant">
    <rdfs:subClassOf
rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
  </owl:Class>
  <owl:Class rdf:ID="Institutional_fact"/>
  <owl:Class rdf:ID="Expressive">
    <rdfs:subClassOf rdf:resource="#Informative"/>
  </owl:Class>
  <owl:Class rdf:ID="Speech_Act_Instance"/>
  <owl:Class rdf:ID="Authority"/>
  <owl:Class rdf:ID="Directive">
    <rdfs:subClassOf rdf:resource="#Performative"/>
  </owl:Class>
  <owl:Class rdf:ID="Assertive">
    <rdfs:subClassOf rdf:resource="#Informative"/>
  </owl:Class>
  <owl:ObjectProperty rdf:ID="HasContext">
    <rdfs:range rdf:resource="#context"/>
    <rdfs:domain rdf:resource="#Speech_Act_Instance"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="Manipulates">
    <rdfs:range rdf:resource="#Institutional_fact"/>
    <rdfs:domain rdf:resource="#Speech_Act_Instance"/>
  </owl:ObjectProperty>
```

Figure 6-3: OWL Excerpt of Transformed POUP classes

6.6. Proposed IWs Ontology Server

As mentioned earlier in previous sections, the ontology is a complex information object consists of many entities in taxonomy with a complicated relationship between them. In order to manage this kind of information object we need appropriate information system, which intended to manage large amount of information within an enterprise, this information system is the ontology Server.

Thus, ontology server is an information system that responsible for managing ontologies. This server provides some tools to achieve essential tasks such as developing and editing the ontology.

Ontology servers are closely related to Computer-Aided Software Engineering (CASE) tools, which are a relatively mature technology. CASE tools are generally used to support the design of a system. Ontology servers are mainly in three lifecycle stages, design, commit and running time.

Our proposed ontology server expected to manage both entities of perdurants and endurants it will provide some of functionalities described in table (3-1) in chapter four. Proposed profile in previous section models perdurant ontology while ODM profile models endurants, there should be some mechanisms to connect instances of both ontologies, because endurant ontology represent the static view of the information system or interlocking information systems, while perdurant ontology represents the behavioral view of the same system or integrated systems. Figure (6.3) describes the static view and behavioral view and how they can be merged into a unified UML ontology.

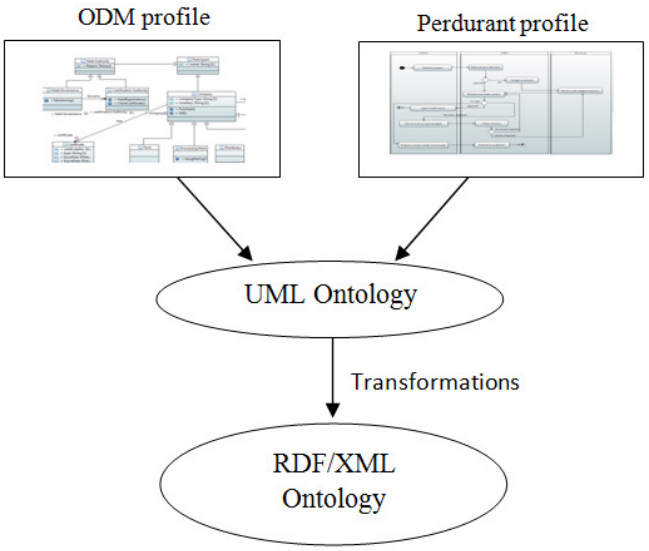

Figure 6-4: the Endurant and perdurant ontology merged into ontology

6.7. The framework

The proposed framework for ontology server as illustrated in figure (6.4) should provide an API to enable server's participants to interact with the server engine in

order to manipulate triples by adding, removing or updating. Furthermore, this API enables participant to commit to the server at first time by let them to query the ontology schema, to maintain their own schema conformed to ontology schema.

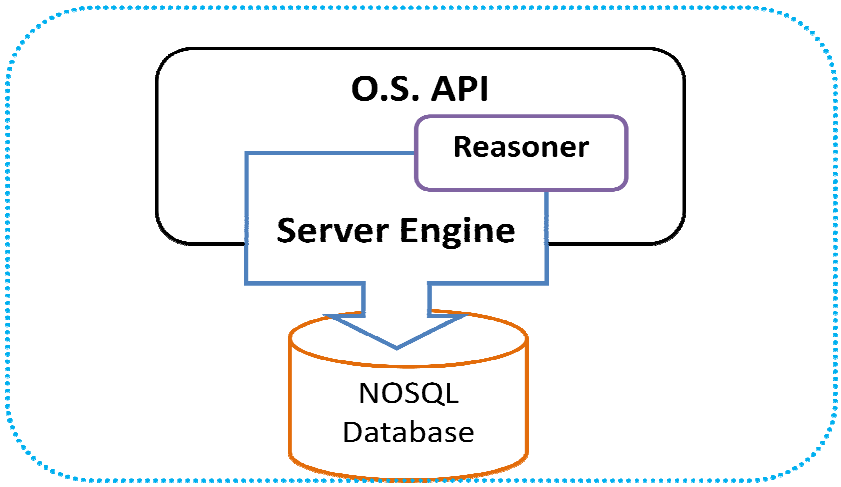

Figure 6-5: proposed framework and its general components

6.8. Ontology Server Components

In this section we will present the proposed approach to manage ontologies for IWS, the proposed framework, figure (6.4), consists of an ***API***, **reasoner**, **server engine** and persistent repository ***NOSQL.***

6.9. Ontology Server API

Ontology server aims to provide a consistent programming interface (API) for ontology application development, independent of which ontology language you are using in your programs, participant who commit to this ontology server make use of the ontology capability via implementing this interface.

The API is designed for managing ontology fragments by adding new class to ontology structure or new instances for existing class, simply the API provide an interface to handle and edit ontology components, as well as querying for existing data.

6.10. The Reasoner

Inferencing on the Semantic Web and ontology can be characterized by discovering new relationships. On the Semantic Web, data is modeled as a set of relationships between resources. ***Inference*** means that automatic procedures can generate new relationships based on the data and based on some additional information in the form of a vocabulary. Inference on ontology and the Semantic Web is one of the tools to improve the quality of data, by discovering new relationships, automatically analyzing the content of the data, or managing knowledge on the ontology in general

6.11. The Server Engine

Store and retrieve facts from database, it would be used for submitting and querying ontology data,

6.12. NONSQL Database

Finally, ***the Ontology repository***, it is a persistent storage for keeping ontology data and schema. NOSQL databases are recommended, but other databases are quite enough, in our case study, we will use available components to build up our ontology server. Table 6.4 shows some semantic web technologies.

Table 6-4: Current Available Semantic Web library

Requirement	Virtuoso	Oracle	OWLIM	Allegro	Bigdata	Mulgara	4Store	Sesame	Stardog	B*	DB2	Fuseki
Open source	Yes/no	No	No	No	Yes	Yes	Yes	Yes	No	No	No	Yes
Free edition	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
10 billion statements	Yes	Yes	Yes	Yes	Maybe	No	Maybe	No	Yes	Coming	?	No
Clustering	Yes	Yes	Yes	Yes	Yes	No	Yes	No	Yes	Cloud	Yes	No
SPARQL 1.0	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
SPARQL 1.1	Partial	Yes	Yes	Yes	Yes	Partial	Partial	Partial	Yes	Yes	Partial	Yes
SPARQL Update	Non-std	Yes	Yes	Yes	Yes	TQL Upd	Yes	Yes	Yes	Yes	No	Yes
Support	Yes	Yes	Yes	Yes	Yes	No	No	Yes	Yes	Yes	Yes	Yes
Events	Yes	Yes	Yes	No	No	No	No	Yes	No	No	Yes	Yes/no
Reasoning	Rules	Materialized	Rules	Rules	Datalog	Rules	Add-on	Little	OWL + rules	No	?	Rules
Constraints	No	Yes	No	No	No	No	No	No	Yes	No	No	No
Triple-level security	Coming	Yes	No	Some	No	No	No	No	No	No	No	No
Endpoint built in	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes
Live backup	Yes	Yes	Yes	Yes	?	Yes	Yes	Kind of	Kind of	Yes	Yes	Yes
Embeddable	Yes	No	Yes	?	?	Yes	Yes	Yes	Yes	Yes	No	Yes

6.13. Constructing the Server From Existing Semantic Web Technologies:

The proposed ontology server composed of semantic web related technologies see figure 6.6. Triple store for storing ontology data supported with Mysql Database. API is for accessing and manipulating ontology data. A reasoner to inferring new information from exiting triples. In the next paragraphs, we will describe these technologies in details.

6.14. Jena Project

Apache Jena is an open source Semantic Web framework for Java. It provides an API to extract data from and write to RDF graphs(Khan and Kumar, 2014). The graphs are represented as an abstract "model". A model can be sourced with data from files, databases, URLs or a combination of these. A Model can also be queried through SPARQL 1.1. Jena provides support for OWL The framework has various internal reasoner and the Pellet reasoner (an open source Java OWL-DL reasoner) can be set up to work in Jena.

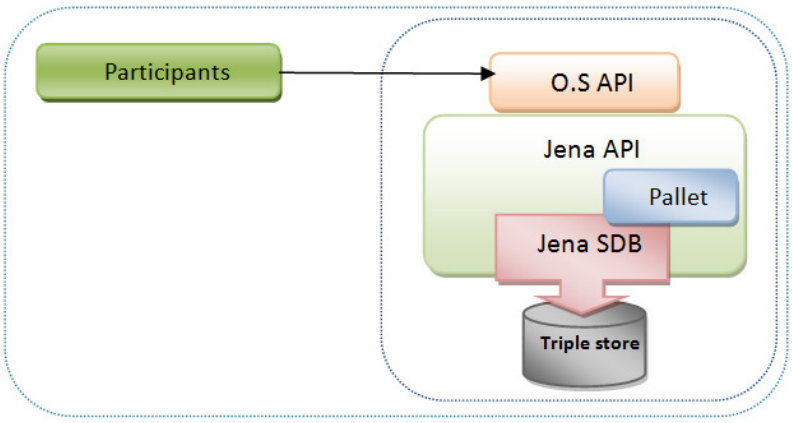

Figure 6-6: composing the server of current semantic libraries

6.15. Jena API

It is an open source Semantic Web framework for Java. It provides an API to extract data from and write to RDF graphs, an ontology API for handling OWL and RDFS ontologies. We can use it as a way for manipulating Halal food Ontology's

data during running our case study. We need it to add instances of the ontology classes and processes.

6.16. Jena SDB library

It is a component of **Jena server** for storing RDF, and query specifically to support SPARQL. The storage is provided by an SQL database and many databases are supported, both Open Source and proprietary. An SDB store can be accessed and managed with the provided command line scripts and via the Jena API. For Reasoning and extracting new data from ontology

6.17. Pellet library

It has been added, Pellet is an open-source Java based OWL 2 reasoner. It can be used in conjunction with both Jena and OWL API libraries. It incorporates optimizations for nominal, conjunctive query answering, and incremental reasoning(Khan and Kumar, 2014).

6.18. Committing to the ontology server

In order to commit to the ontology server, participants should maintain their system of institutional fact to conform to the schema of the ontology stored in the repository of the server. The participant can commit to whole ontology or a part of the ontology if the ontology was partitioned into modules.

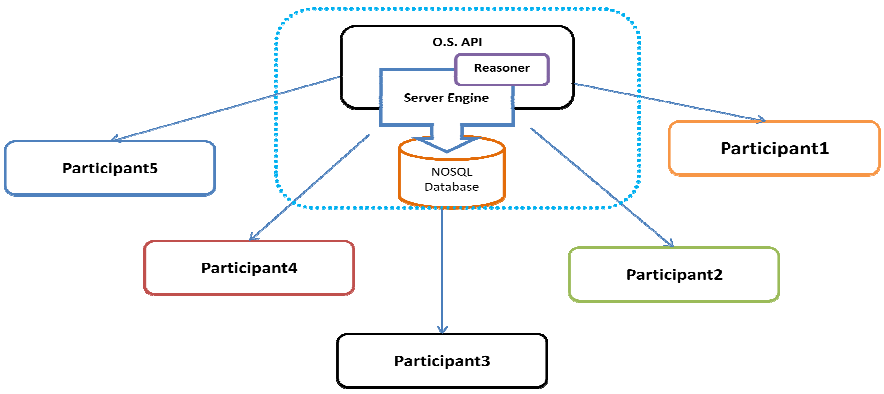

Figure 6-7: participants reuse the schema

6.19. Proposed O.S. Features:

The proposed ontology server expected to provide its participants with a number of facilities to enable them share their system of speech acts as well as their institutional facts. The server expected to provide the following functionalities:

- **At design time**: the instance of the proposed server, which composes of current libraries of the semantic web, does not provide any functionality at design time, in spite of the ability of achieving the task. Since these functionalities are out of the scope of the study, it was left for future work.

- **At commit**: the proposed ontology server provides its participant with the schema of the ontology and a facility to reuse this schema to commit to the ontology server.

- **At run time**: since all speech acts (transactions) are recorded to the ontology repository, in away analogous to database transaction log, then retrieving process will be possible, and hence the server could make use of this mechanism to trace back. In addition, track series of transactions in order to either provide information of what was happen, or make sure that speech acts are achieved perfectly.

6.20. Summary

In this chapter, we have presented a framework for ontology server for managing ontology component. The proposal shows two frameworks, the first is general, one can chose or build his own components, in the second we have made use of available semantic web libraries to build up our server, it compose of Jean library for manipulating ontology elements and querying via SPARQL, Pellet library for inferencing. What is new in this ontology server framework is that, it supports the representation of the perdurant entities. Furthermore, it helps IWs participants to trace and track transactions via querying ***speech act log***.

CHAPTER 7

CASE STUDIES:

Insuring Halal Food Integrity Using Ontology Server Supports Ontology of Perdurant

7.1. Introduction:

Insuring Halal food integrity becoming very important issue due to the internationalization of food trading and due to the increasingly number of Muslims around the world, for Muslims before start consuming food, they should check whether the food is Halal (permissible to consume) or not. However, the food now goes thought long processes and stages since producing by some company to storing in different places and circumstances, distributing and selling until it reach customer hand. There are a number of suppliers including in this series of stages, they need to interoperate and exchange information, and they need a consensus vocabulary to use in the interoperation. Ontologies are widely used to provide such vocabulary; but current ontology development tools are only representing static data, entities that exist in timeless (which are called endurants), and ignoring the representation of occurrence entities (perdurants), the proposed ontology for halal food IWs represents both perdurants and Endurants. In our case study, we will present ontology for Halal Food interlocking institutional worlds contain both endurant and perdurant entities. We will use ***ODM profile*** by ***OMG*** to represent the structural elements of Halal ontology, and will examine both **owl-s** and ***Demo profile*** and we will show why they are insufficient to represent perdurant ontology, we will present an enhanced profile of perdurant then we will apply our proposed ***Perdurant profile*** as alternative for mentioned approaches. Finally, we will use the ontology server (***HFOS***) to manage halal food ontology in order to resolve the problem of halal food integrity. This case study, we only concentrate on livestock flesh halal food.

7.2. Motivations:

The following attributes were seen as key drivers for the selection of this case study:

- In halal industry, there is no standardized ontological hierarchy of halal ingredients. Moreover, there is no consensus specification for producing halal products and issuing halal certificates. Therefore, semantically, there are differences in halal terms definitions.
- The case study is suitable for applying our model (profile) because it contains both static entities as well as behavioral entities, gives us the opportunity to examine both (endurant and perdurant). For example, halal registration leads to issue halal certificates, which compose of series of actions (workflow). Thus, this workflow should be pre-defined to achieve the automation process. Instances of these workflows represent individuals of perdurant ontology.
- The case study is complex enough and representing real interlocking institutional worlds. It involves different type of institutions (companies) and all of them work together, and interoperate. Semantics is needed in this interoperation and hence the existence of the ontology.
- Halal food industry is particularly vulnerable to unintentional and intentional threats:
  - Visible integrated standards and monitoring mechanisms approximately do not exist across Halal food supply chains.
  - Critical gaps in Halal food safety/security exist.
  - The risk of intentional threats to Halal food supply chain has increased
  - The impact of an incident (intentional or unintentional) can be significant

7.3. Case study strategy:

The following figure represents the strategy we will follow in developing our case study; it starts with defining case study essential terms including Halal food interlocking institutional worlds, its participants, and roles and ending with running queries, showing and discussing findings. Figure 7.1 shows the strategy.

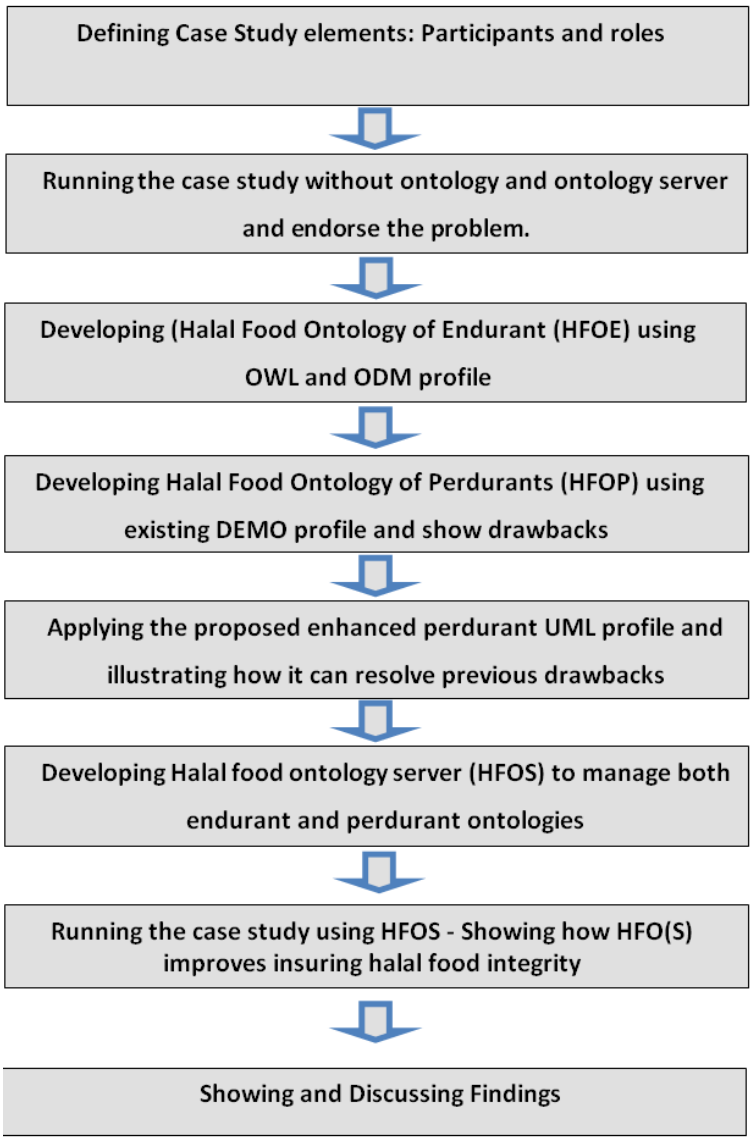

Figure 7-1: Case study Strategy

7.4. Halal food Integrity Problem

"60 percent of the meat which is being sold with halal certificate in New York and New Jersey is not halal,"

Ali Kucukkarca, the owner of the biggest halal slaughter facility in the east of the US[1] said.

One of the most important questions for Muslims in multicultural and multi-religion countries is “halal meat” certificate and how it can be trusted. Meat products before reaching consumer's hand, travel through a number of suppliers and distributors,

[1] - Halal focus daily Halal market news at: http://halalfocus.net/usa-muslims-having-problem-in-halal-meat-in-us accessed 21-11-2013:10:00

and during this travel there are specific processes should be followed to conceder these products as permissible (Halal) such as slaughter process, packaging and coating, distribution and storing conditions. The main question should be answered is that***: how can we insure that these products are Halal?*** For instance, when we buy a package of meat from butchery or from retailer, how can we trust ***Halality*** of that product? To answer the question we need to follow and trace back the product within the supply network and verify that any supplier in the supply network has achieved all processes according Halal and food safety. To accomplish this verification we need to collect all information regarding the product since producing to specific position in the supply chain, two more questions arise there, ***who can perform this process for the customer or supply chain's participant, in other words, who will provide a service that can check food Halality?*** Obviously all suppliers (participants) are making use of this service. Therefore, we need a third party independent of Halal food IWs participants' and hence independent of their information systems to achieve that service. The second question is that, ***how can we collect needed information since it might be private to some participants***? Moreover, ***how to make sure that this information is genuine and could be trusted?*** We can go more in depth, actually this information are institutional fact result from performing some speech acts, ***how can we record both records of institutional fact and the status of each speech act in order to provide accurate information***? This case study tries to cover all these aspects. Figure (7.2) represents a simple Halal food IWs consisting of some participants. It is obvious that there are many problems regarding the process of query information from previous participant directly or via another supplier in the interlocking institutional world. Moreover, if the supply chain is long enough this process will be too tedious, for example if a retailer needs information about a farm, then it either go directly, or through wholesaler and process plant. Both are difficult, for the first state there may not be a direct interoperation, for example pre-established EDI sessions between the retailer and the farm and thus it is difficult for them to understand each other. In the second situation, a retailer must query wholesalers, and wholesaler queries process-plant, and finally process-plant will query farms and send the result to the end retailer and at the end, we might have uncertain information. Furthermore, if the supply network very long then query process will be inconceivable.

7.5. Participants and Roles

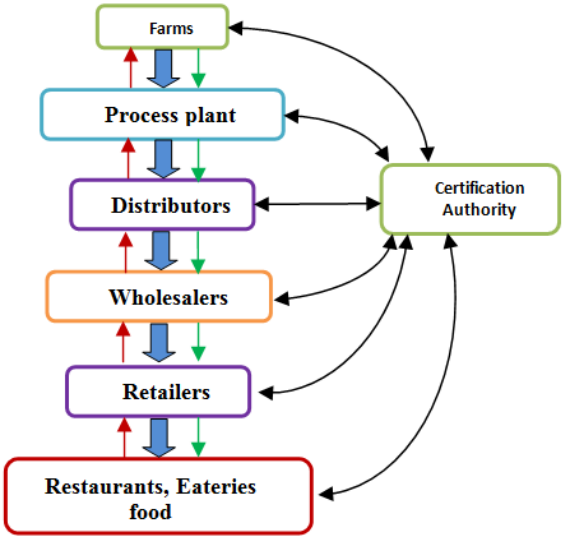

Figure 7-2: Case Study Participants

In this section, we will present a brief description of our case study. To simplify understanding, we will look at our case study as a game consisting of players and movements regulated by game-rules. Halal Food supply chain (since now we will nominate it Halal Food interlocking institutional worlds ***HFIWs***) consists of a number of ***Participants***, (***players***), each of which performs a number of activities and tasks (***playing specific roles***), regulated by regional or international laws and regulations (***framing rules***). **Halal Certification authority** is the body that governs and controls Halal Food trading within a region or country; it is responsible for issuing Halal certificates to all participants (suppliers) who are participating in Halal food interlocking institutional worlds. There are two types of Halal certification authorities, **certification authority** and **Halal Governance** the former responsible for issuing halal certificates while the latter monitors halal trading in all country involving monitoring **Certification authorities** themselves**. Farms** are business that devoted primarily to the practice of producing and managing Halal food (produce, grains, or livestock). Farms always imports livestock and breeding and feed them with feeds not containing Non-Halal ingredients. **Process-plant** or Slaughterhouses (Butcheries), or meat-works is a facility where animals are slaughtered (killed in specific way according to Shari'a) for consumption as food products. **Wholesaler**: a firm or person that buys large quantity of goods (Halal

food products in our case) from variant producers or vendors, warehouse them, and resell them to retailers. Wholesalers who carry only non-competing goods or lines are called **distributors**. **Retailers** Retail is the sale of goods and services from individuals or businesses to the end-user. A retailer purchases goods or products in large quantities from manufacturers directly or through a *wholesale*, and then sells smaller quantities to the consumer for a profit (Ibrahim et al., 2015).

There are more but not essential participants that are involved in these IWs such as ***bank***, ***credit-card*** companies and ***shipping*** companies. There also some entities that appear with some participant in specific time, examples of these entities are ***purchasing*** process which appear with companies in case of buying and selling products, and ***Slaughtering,*** which appear with slaughterhouse Participants in time of killing an animal for meat, see figure (4) UML class Diagram. These entities are Perdurants as recently called, in Islamic literature, more than fourteen thousands of years, they called it "**Arad**" which something that depend only on another entity Which called "**Gerem**" and appear in specific period of time. For example, the sun is "Gerem" but sunshine and the activity of shining is "Arad". Another example, a person is "Gerem" but when he be sick the illness, which appear on him is "Arad".

Each participant in this IWs plays specific roles, for instance ***Halal Certification Authority*** plays the role of issuing certificate (***halal registration***) and monitoring halal production procedures and materials, allowing ***HFIWs*** partners to do business in halal food market. It also controls other certification authorities by monitoring their businesses and applying international or country's Halal regulations. ***Farms*** play the roles of importing (buying) and breeding animals, sell them to slaughterhouses (process plant). ***Slaughterhouses*** play the role of slaughtering animals (livestock) for meat, and packaging it in products. ***Distributors***, ***Wholesalers*** and ***retailers*** play the role of purchase and sell halal food products as well as warehousing.

7.6. Static View of HFIWs

The following figure shows an UML class diagram representing Halal Food IWs structural view, it shows the participants as classes with properties and operations,

relations between classes as association. Beside participants, there are products, which are composed of ingredients, and certificates.

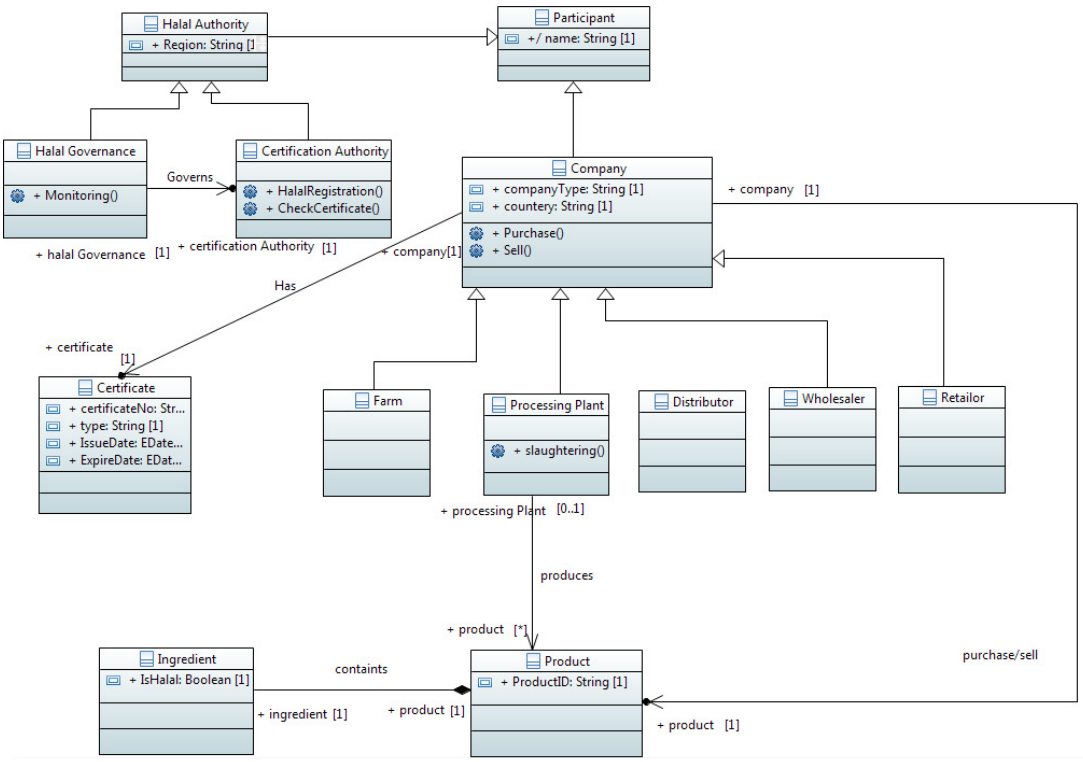

Figure 7-3: Static View of the Case Study

Participant parent class has two sub-classes, ***company*** and ***halal Authority***, which also has two sub-classes Halal governance and certification authority. The former has one informative speech act which monitoring processes of issuing halal certificates from certification authority. The second sub class has two speech acts, Halal ***Registration*** that is composite performative speech act, which result in producing halal certificate for specific company, and ***check certificate***, which is informative speech act result, is verifying of availability and validity of halal certificate. The sub-class company has many sub-classes such as farms, processing plant up to retailer. In the next section, we will concentrate on the basic processes (perdurant entities) of this diagram.

7.7. Behavioral View of the HFIWs

UML class diagram in Figure 7-4) shows classes their properties, operations, and relationship between classes themselves. It does not show details about processes and operations, For instance, Halal-Registration, slaughtering, Monitoring and purchasing; UML uses activity diagrams to describe dynamic aspects of the system. In this section will present the behavioral view of the ***HFIWs*** using UML activity diagrams.

7.7.1. Halal Registration Process

This process provided by Halal certification authority participant as shown in figure (7.4), other participant get involved in this process such as companies. Figure (7.5) illustrates details of Halal-Registration operation (Issuing **Halal** certificate process).

Halal registration has two different types of registrations as processes, ***registration for new certificate*** or ***renewing***. Registration process started when a participant (company) apply for either getting a new or renewing existent halal certificate, this could be achieved via submitting an application form supported with required documents, applied participant should pay registration and certificate fees before continue registration process; certificate authority check application for completeness, it either accept the application or requesting for more information documents. The certification authority sends inquiry to manufacturers for product ingredients and consulting Islamic committee to insure that all product and its gradients are halal and conforming to Shari'a. Then it performs an onsite auditing process to check halal and regional framing rules conformance and tools used in producing halal products, if the onsite auditing result is passed then the certification authority set approval to issue the certificate and make a contract with certified company.

In case of renewing the certificate, Certification Authority (CA) will send a renewal notice to the certificate holder together with an application before the expiry of the current Halal certificate. The applicant must submit the completed form with the necessary valid documents attached, CA will achieve onsite auditing and inspections and receive payments before issuing the certificate.

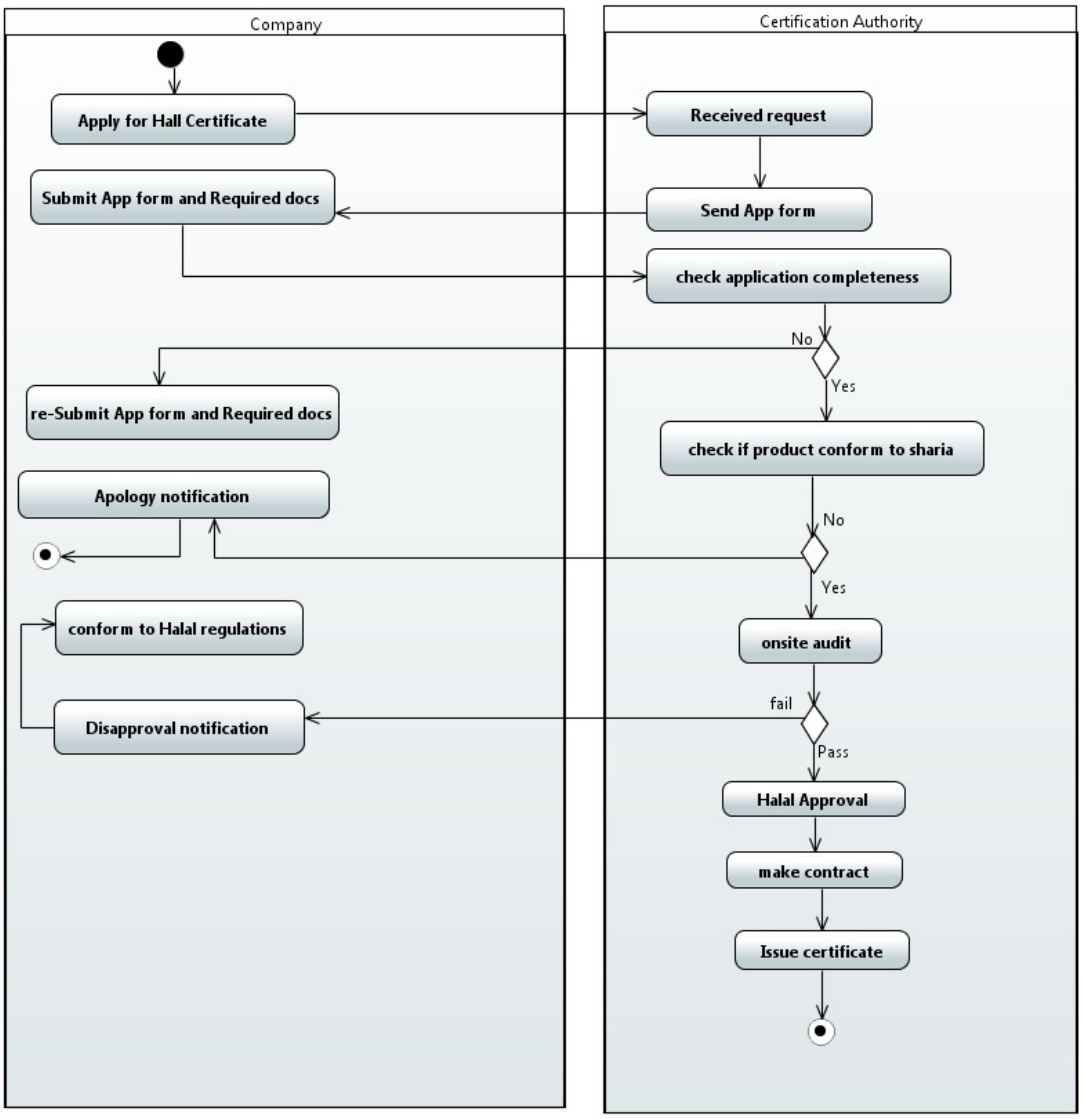

Figure 7-5: Registration Process Activity diagram

7.8. Selling and purchasing processes

Purchasing and Selling are opposite operations both buyers and sellers are instances of company participant, each company can play the role of seller or buyer, they are a composite processes. When a participant (institution, supplier or customer) purchasing a product, in order to investigate whether the product is halal or not, he performs a process consisting of dozens of tasks. For instance, it checks the existence and validity of all supply chain members' certificates who are involved, checks the origin of the product, and tracing back the product since produced up to current state. This process seems very complex, to make it easy to

be understood we will break it down into small operations shown in the following paragraphs respectively:

Purchasing could be divided into two interactions the first part between two companies, Buyer Company and Seller Company. During purchasing buyer company may need to check halal certificate of Buyer Company, or trace back the rout of the product via the supply chain. Therefore Buyer Company to communicate with certificate authority for check certificate existence and validity, and query for product transmission via the supply chain using one of the approaches mentioned in section 7.6. Purchasing process between buyer and seller could be represented as activity diagram and so certificate checking. However, for query about product transmission, it would be difficult to be represented, because the length of the supply chain is not known and hence number of participant (swim-lanes) could not be identified. Figure (7.6) shows the activity diagram for purchasing process.

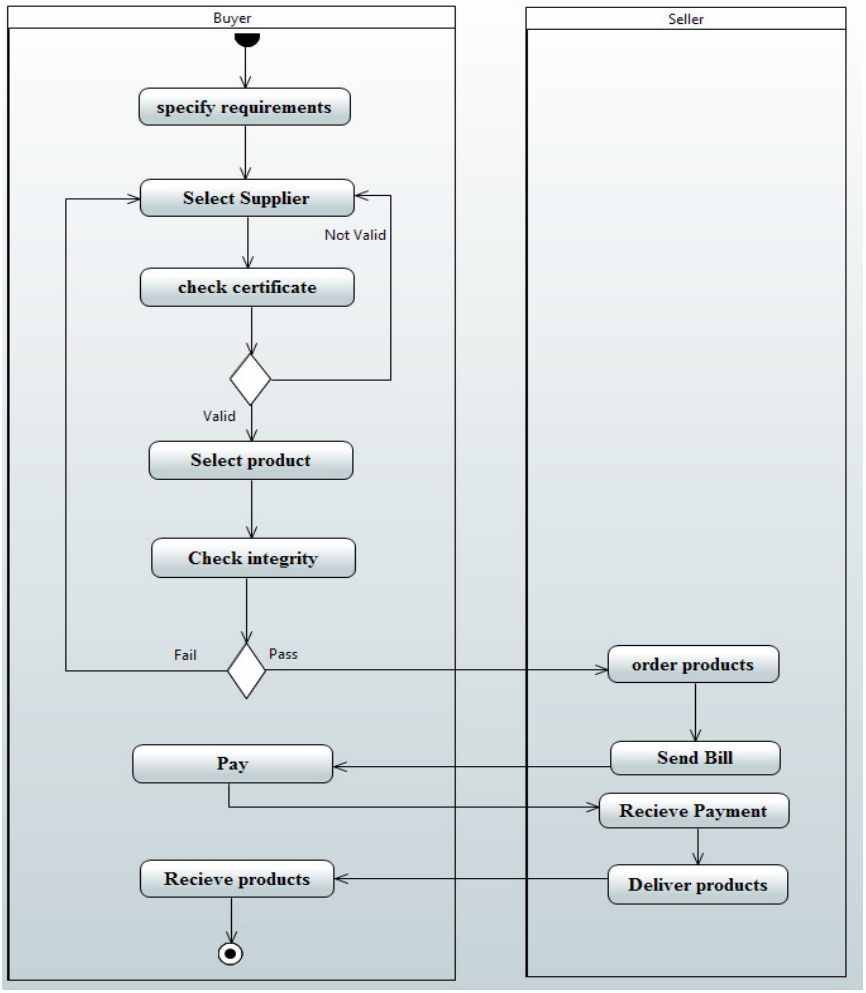

Figure 7-6: Purchasing diagram

Action check certificate and integrity check require more participants to get involved in this interoperation, but there are many problems to prevent these two actions to be achieved perfectly. For example, to check the halal certificate buyer need to refer to halal authority to check whether specific company has valid certificate or not, there might be many certification authorities in their region, to which one buyer can refer to?, moreover, if there is no direct connection between these companies, how to perform this operation?. For checking integrity buyer should query about products across the supply chain, how these queries could be conducted. There is necessarily an insistent third party helps in handle these problems. Suppose there is only one halal authority in this region then we can represent the process of checking certification validity via the following diagram (7.7).

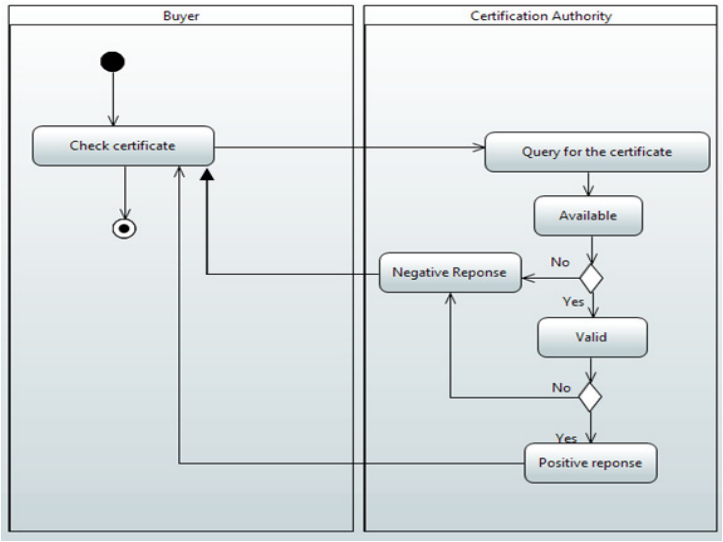

Figure 7-7: checking certificate

We have mentioned three approaches to query for product transmission; the most applicable one is by place the query to shared third party, refer to section 7.6, suppose we have one such central organization we can draw the following diagram.

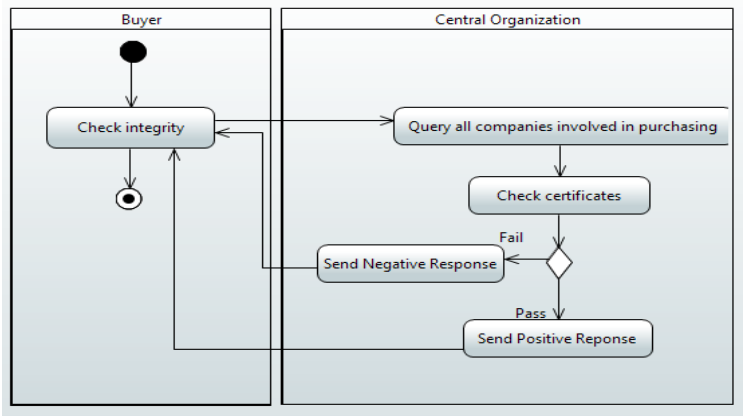

Figure 7-8: checking integrity

Certification check process, the aim of this process is to check whether a specific participant has halal certification or not, and this certificate originated from registered and trusted authority. This process is very important; because any participant need to participate on this IWs will make use of it. The first step of checking halal integrity is that, all members of the supply chain through which the product has passed must have the certification. Otherwise, the product might be counterfeited. Figure (7.9) shows the input and output parameters, it takes certification No and the name of certification authority which issued the certificate, and return two Boolean values, the availability and validity of the certificate.

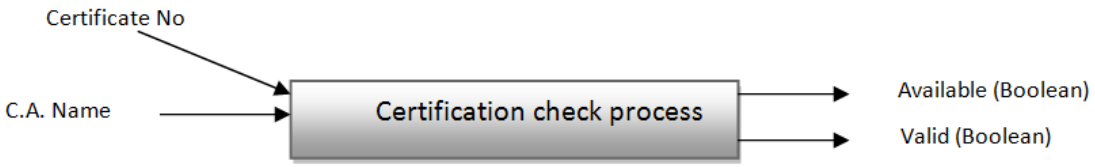

Figure 7-9: Certification check process

Checking Halal Integrity Process, achieves a series of checking, we can summarize them in the following sub-processes:

- **Checking product transmittance process:** traces back the product to insure that it has ambulated through trusted suppliers and every one of them has a certificate from registered halal authority. This process expected to return the sequence of participants and classify them into authorized and not

authorized, it make use of ***certificate check process*** to determine whether a participant has valid certificate. If it has valid certificate then it authorized, otherwise it is unauthorized.

- **Checking Halal status process** verifies whether product is halal or not via checking product's ingredients – all of them should be Halal – and make sure that the product have been prepared according to food safety procedures and conform to halal authority legal procedures.
- **Checking storage conditions**: insure that the product along the supply chain has stored in distribution containers and warehoused in a perfect circumstances.
- **Checking product validity:** verify that the product is valid and still valid to be consumed.

We can consider checking integrity process as a ***black box*** service tacking halal food product id and executing a number of internal; check halal status, integrity status, product validity, and storage conditions.

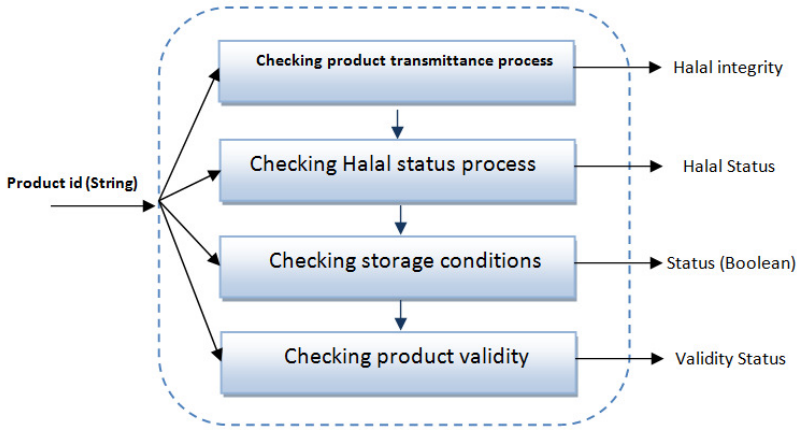

Figure 7-10: Checking halal integrity operation

If the results of all sub-operations are true, then the result of the whole process is true, and hence, the product is assured and halal.

7.9. Halal Monitoring Process

When certification authority issues certificate to specific participant, a record of all data and events that happened during registration process is stored in the halal certification authority's repository. Then, when *Halal Governance* needs to check whether that certificate was issued perfectly and according to regional and Governance framing rules, then it will take a copy of that record and compare the processes with stored rules in it database. Hence, it can decide whether this certification authority follows rules or breaking them also decide whether to prevent or allow them to issue certificates in future.

To achieve this task, Halal governance should have an instance of each process of issuing certificate; this will not be done unless there are records of transactions recorded in third party server accessible for halal governance.

7.10. Instantiating Case Study Elements

In this section, we will try to run our case study with concrete instances. Suppose we have five Halal food suppliers ***Halal Square (HQ), Sheep Australia Pty Farms (SAPF) Sydney slaughterhouse (SS)***, ***Halal Choice (HC) and VKS farms*** trading in Halal food products, ***Basfood*** is halal distributer. Moreover, we have three Halal Certification Authority responsible for issuing Halal certificates ***Australian Islamic Monitor*** (***AIM***), ***Halal Australian*** (HA), and ***Australian Halal Authority and Advisers (AHAA). Australian Halal Development and Accreditation (AHDA)*** governs all authority bodies in our case study and which represent Halal Governance Participant. ***See Figure 1***

Suppose ***VKS*** working in breeding livestock. It imports them from outside Australia. ***SAPF*** also trading in breeding livestock (sheep) and it has business with ***VKS.*** It sometimes buys livestock from it***. SS*** is a slaughterhouse always prepares halal meat and packs them in packages (*meat products*) after slaughtering animals according to Halal food regulations and laws. ***HC*** and ***HQ*** are wholesalers trading buying and selling halal products such as Beef meat and poultry.

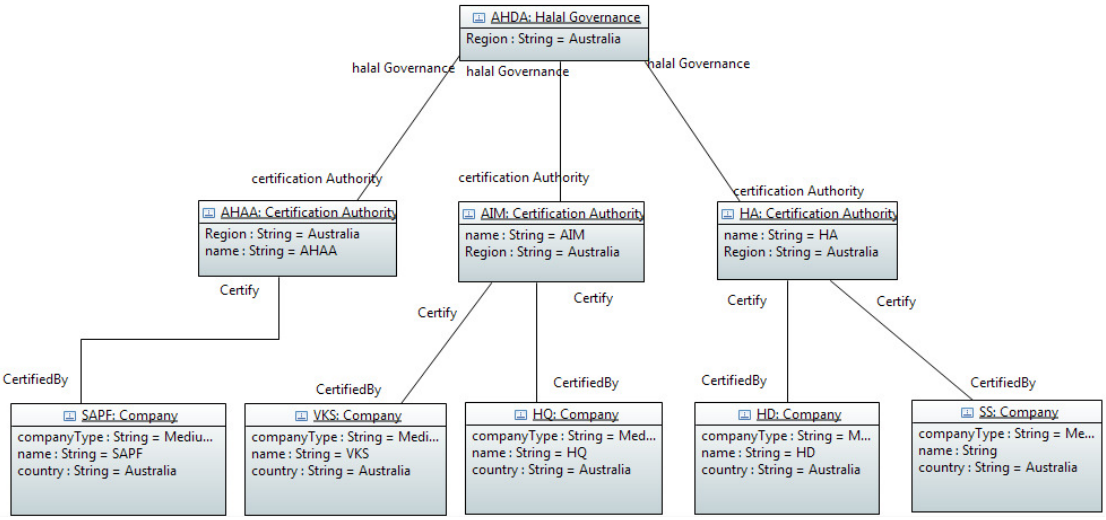

Figure 7-11: participant Instances

7.11. Non-Ontological Halal Food Tracking And Tracing Approaches:

In halal food industry there are three different types of tracking and tracing halal product within the Halal food interlocking institutional worlds (***HFIWs***) depicted in figure (7.3), the first type is ***cascade traceability systems*** (denoted with **CTS**) in this approach each link in the production chain gets its relevant information about the former participant from the former links. For instance, if a retailer queries for information about a farm four level above then it must get this information through a wholesaler, the wholesaler will get information from the above link and send it down to the retailer and so forth. One advantage of this approach is that the amount of information per transaction remains small and hence reduces transaction costs. But a considerable drawback of this approach is that it is largely based on trust, each link has to trust the former link on the quantity and quality of information. Another noticeable point is that if the supply chain is highly complicated and interlocked, retrieving information would be so complicated too, and there would misunderstand of terms and messages used in interoperation, and hence there must be a standard way for interoperation between participants in order to interchange information and documents such as Electronic Data Interchange (**EDI**).

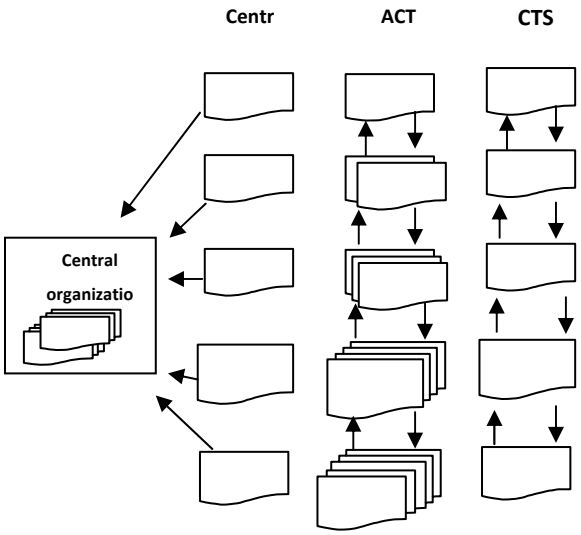

Figure 7-12: Halal Food IWs tracking approaches

The second approach **accumulative cascade traceability system** (denoted by **ACTS**), in this approach each link gets the relevant information about all former participant in the supply chain. This approach also depends on trust but it differs from CTS in the step for fetching information. Because each link in the chain receive all other former participant's information, the amount of data and document increases per link. In addition, there should be a pre-defined framework for interoperation.

The third approach uses a ***central organization*** for hosting and storing information and document. In this approach, each link provides relevant information to the central organization, which combines the information of all other link in whole supply chain thus overcome trust matter. Also if the supply chain complicated and interlocked the interoperation between participants and the central organization would be inextricable due to different participants with different way of doing business. There would be a semantic heterogeneity.

7.12. Running The Case Without Ontology-Based Approaches

Suppose we have some halal products transported via a series of suppliers within halal food interlocking institutional worlds as depicted in the following figure(7.12), there are three products A,B and C each of which transported via specific workflow instance, for instance product A transported via A1,A2,A3 and A4, crossing ***VKS,SS,HQ, Stock-land*** to ***Ash-Sedap***. According to each company has certificate from specific certification authority, for instance ***VKS*** has halal certificate from ***AIM*** see figure (7.9). Suppose there is a new company, ***Basfood*** as halal food distributer, and not yet has halal certificate from any certification authorities. It has applied for halal certificate from AHAA. In the following sections we will discuss three different scenarios, ***Basfood*** **Registration** scenario, **buy- sell** scenario and **Monitoring** scenario.

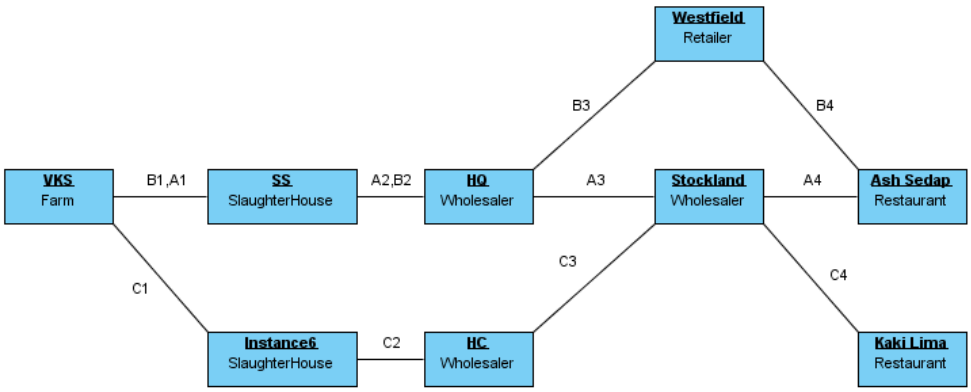

Figure 7-13: Workflow instances

7.13. Registration Scenario

Without halal food ontology and ontology server, all information regarding to Halal registrations will be kept in certification authority database which accessible only for certification authority itself and not others. This information is either about items such as halal product data or events that happen during halal registration. for the first type all of them are actually recorded in the database but the second might not recorded, even if it was recorded it only represent database transaction log

which save only events of operations in database management system and not other operations.

For example, when ***Basfood*** applied for halal certificate from ***AHAA,*** there was an interoperation between them, and they have exchanged messages including data and documents. If we look to the database of the certification authority we can only see ***Basfood*** certification record, we can only query about ***Basfood*** registration data but we cannot query about how these certificate was issued. Furthermore, this record is only available for ***AHAA*** staff but not sharable. If the registration process automated, some of these exchanged messages will represent speech acts, which perform events in ***AHAA*** information system resulting in some changes of ***AHAA*** data, which is not accessible to ***Basfood,*** and though prevent automation. The following table shows interoperation between the two participants, speech act, its type and institutional fact result of the speech act.

Table 7-1: Speech acts between AHAA and Basfood

Speech Act	Type	Participant	Inst. Fact
Apply for H.C.	Performative	Basfood	Request
Receive request	Informative	AHAA	Request
Send App. Form	Informative	AHAA	App. form
Submit App. Form	Informative	Basfood	App. form
Check application completeness	Performative	AHAA	App. form
Resubmit App. Form	Informative	Basfood	App. form
Check product conformance	Performative	AHAA	App. form
Apology notification	Informative	Basfood	Apology
Onsite audit	Performative	AHAA	--
Conform to halal regulations	Performative	Basfood	--
Disapproval notification	Informative	Basfood	Apology
Halal approval	Performative	AHAA	Approval
Make contract	Performative	AHAA	Contract
Issue certificate	Performative	AHAA	H.C.

You can see that the interoperation is directly between AHAA and Basfood there is no mediator between them. Records of the interoperation are kept in both participants database. Hence, it is difficult to query for whole speech acts of the interoperation. However, if the data are stored in one sharable place then it will possible for all participants who have the permission to access them.

7.14. Buy-Sell Scenario

Halal food products travels through a number of suppliers , between each pair there is sell/buying process, some supplier play the role of buyer and seller at the same time, for example product A was produced and sold by VKS bought by ***SS,*** and then SS sold it to HQ which sold it to Stock-land and so forth.

With every buy/sell process, there are a series of checking to insure that the product is original and Halal. to perform these checks, buyer should collect information from different resources, in current methods for example ***cascade traceability systems*** information are transferred via different participants, and in ***accumulative cascade traceability system*** information is accumulate in the last seller, while ***central organization*** is better for sharing data but it is lack of semantics.

7.15. Monitoring scenario

Some times Halal Governance needs to check whether certification authority follows halal and healthy rules, or it oversteps them. In order to achieve this process they need information about how these certification authorities issue Halal certificates, in other words they need to know the steps of issuing the certificate, which contains both data used to issue the certificate and processes that was followed. For example, if we have an instance of ***Basfood*** registration process in ***AHAA*** we can easily compare it with standard registration process.

Current approaches does not support such mechanism because of semantic-less and non-sharable data and processes. To apply this mechanism, we need ontology to support interlocking institutional worlds' participants with reusable model to provide consensus standards in representing data and processes.

7.16. Insuring HFIWS Integrity Using Ontology Server

The proposed approach in figure 7.13 is somewhat looks as the third approach discussed in section (7.6), but it provides a consensus vocabulary between all participants i.e. ontology (perdurant & endurant). All participants should conform to this ontology in order to contribute to the Halal food interlocking institutional

world. In addition, this approach provides an information system to manage this ontology, store and retrieve, query, and inference data.

The proposed approach requires ontology to which all HFIWs' participants should conform to; this ontology should contain both endurant and perdurant entities, as well as an information system (ontology server) for managing all HFIWs data and processes; it should provide services during design-time, commit-time and run-time. The ontology server supported with a repository. Figure 7.13 below shows HFIWs' participants and the Halal Food ontology server (**HFOS**) with the repository.

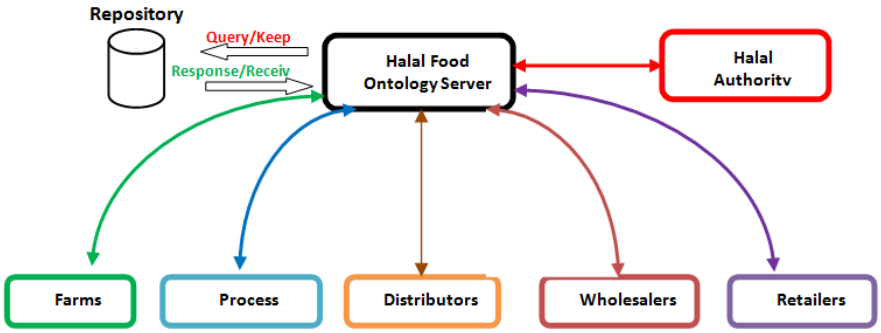

Figure 7-14: Halal food ontology server

7.17. Halal Food Ontology of Endurants HFOE:

There are many representation languages are being used for modeling ontology of endurants; among these languages is web ontology language (OWL) which is one of OMG specification for representing ontologies. UML is the most widely used in modeling these days; One of UML features in the ability to be extended; The Ontology Definition Meta-Model (***ODM***) which is an Object Management Group (OMG) specification makes the concepts of Model-Driven Architecture applicable to the engineering of ontologies. Thus providing a UML profile can be used to model the meta-model of endurant ontology. Figures below show a fragment of Halal Food Ontology model using both OWL and ODM UML profile, the former developed using ***protégé*** case tool and the latter developed using ***Eclipse Luna*** with ***papyrus*** plug-in.

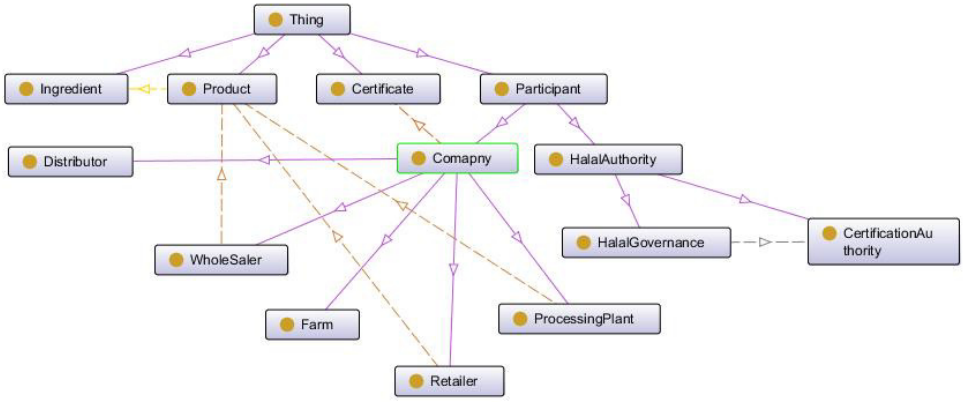

Figure 7-15: Protégé graph for HFO

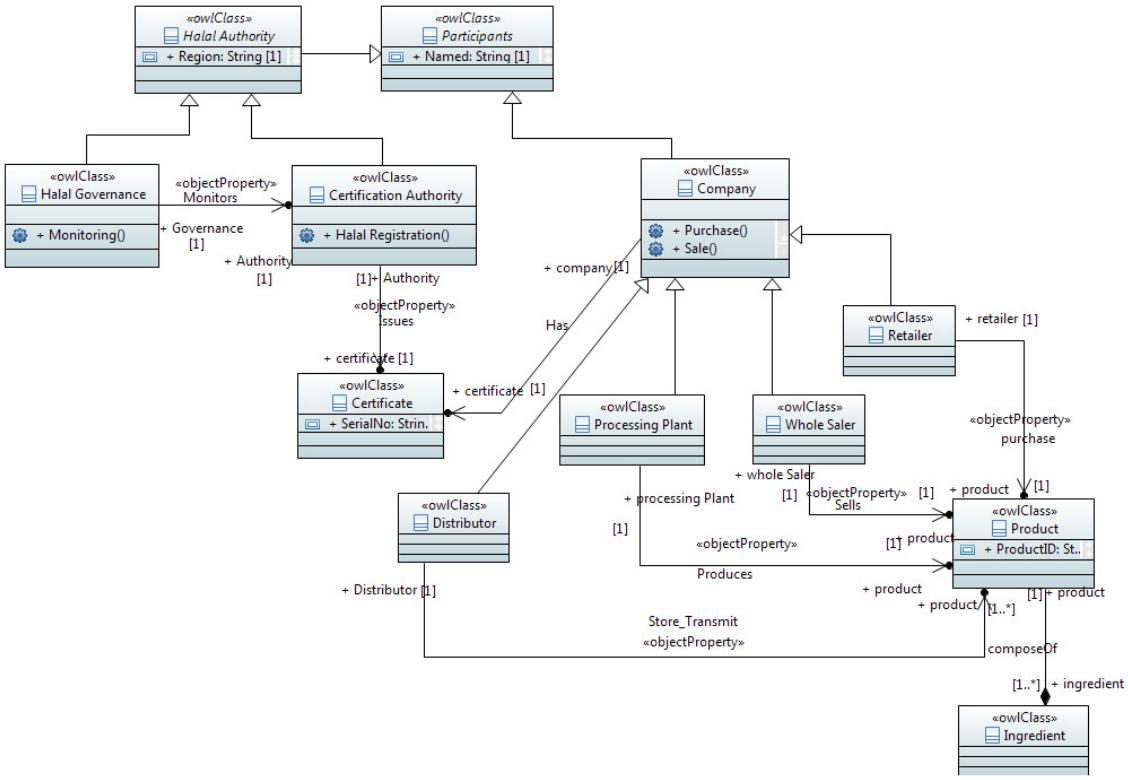

Figure 7-16: Halal Food ontology using ODM Profile

The model shows that the owlClass stereotype which extends UML ***meta-class*** class. There is abstract owl-class ***Participant*** with sub-owl-classes, Company and Halal Authority respectively. Each of which is a parent owl-class of other owl-

classes. However, the profile was not able to represent actions that might happen in Halal interlocking institutional worlds although UML provides activity diagram for representing processes and events because it does not provide model instances.

7.18. Halal Food Ontology of Perdurants HFOP

Now that we have developed the ontology of endurant using UML, and UML profiles i.e. ***ODM*** profile, now we can go ahead in developing the Halal food ontology of perdurant. Endurant ontology is not enough to represent all Halal food domain vocabulary, because it represents only continuant entities (entities that exists in a timeless way), and not showing states and events that might occur. All information systems are founded to perform speech acts in order to achieve their tasks, and record results of these acts. Referring to figure (Pervious), we have some processes (***activities***) in our model, Halal registration process, purchasing process, and Monitoring process, thus we need to represent these processes and events via perdurant ontology. In our case study, we will concentrate only on purchasing, monitoring, and halal registration processes.

7.18.1. Representing HFOP OWL-S

Now we are going to look carefully to our processes Halal Registration, Monitoring, and Purchasing as s services. Because owl doesn't support developing ontology of perdurant, and so UML we will borrow OWL-S ontology (an ontology, built on top of Web Ontology Language - OWL, for describing Semantic Web Services) terms to describe these scenarios. OWL-S organizes a service description into four conceptual areas: the process model, the profile, the grounding, and the service, in our case study we will concentrate on process models.

Consider the processes shown in figure 7.4, **Halal registration, Monitoring,** and **purchasing services**. Each of which will has a **process model** which describes how a service performs its tasks. It includes information about inputs, outputs, preconditions, and results. In addition, they have profiles, which provide a general description of these services. Moreover, they have a **grounding,** which specifies how a service is invoked.

OWL-s process model distinguishes between three types of processes: ***atomic***, ***simple*** and ***composite***. For a ***composite*** process, the process model shows how it breaks down into simpler component processes, and the flow of control and data between them. ***Atomic*** processes are essentially ``black boxes" of functionality, and ***simple*** processes are abstract process descriptions that can relate to other composite or atomic processes

7.18.2. Weaknesses of OWL-S:

During executing Halal food case study, we noticed the following some weaknesses prevent using of OWL-s as a metamodel for representing perdurant ontologies:

- OWL-s does not provide and answer for our main question, ***how can we record an instance of specific process***? The answer of this question represents a key point for our study. To illustrate this point we take an example, can we record an instance of Halal registration process using owl-s, to be more specific can we have the steps and events that was happened during registration of ***VKS*** Company by ***HA***?
- OWL-s represent services as black box without showing service tasks, for example owl-s ***Process Model*** does not show ***Halal registration service*** details – speech acts and task status, it treat it as black box specifying its input and output and pre-conditions. If you refer to table (1) which shows the speech acts records (institutional facts), a participant can easily query to the status of this kind of processes, but we fail to represent this in our OWL-s ontology.
- OWL-S takes a service point of view to describe service activities, so I think it works better for agents than humans.
- Owl-s oriented to Services in service-oriented architecture rather than representing perdurant ontology, thus it is not suitable for modeling perdurant ontology.

7.19. Representing HFOP Using DEMO Profile:

The second approach is the DEMO profile, which was presented by Mohammad Nazir and Robert Colomb 2010 (Colomb and Ahmad, 2010), their profile depends

on Dietz's theory of DEMO (Dietz, 1999) and the theory of speech act which was coined by John Searle (Searle, 1995). Figure 7.16 depicts the profile's stereotypes and UML ***meta-classes***.

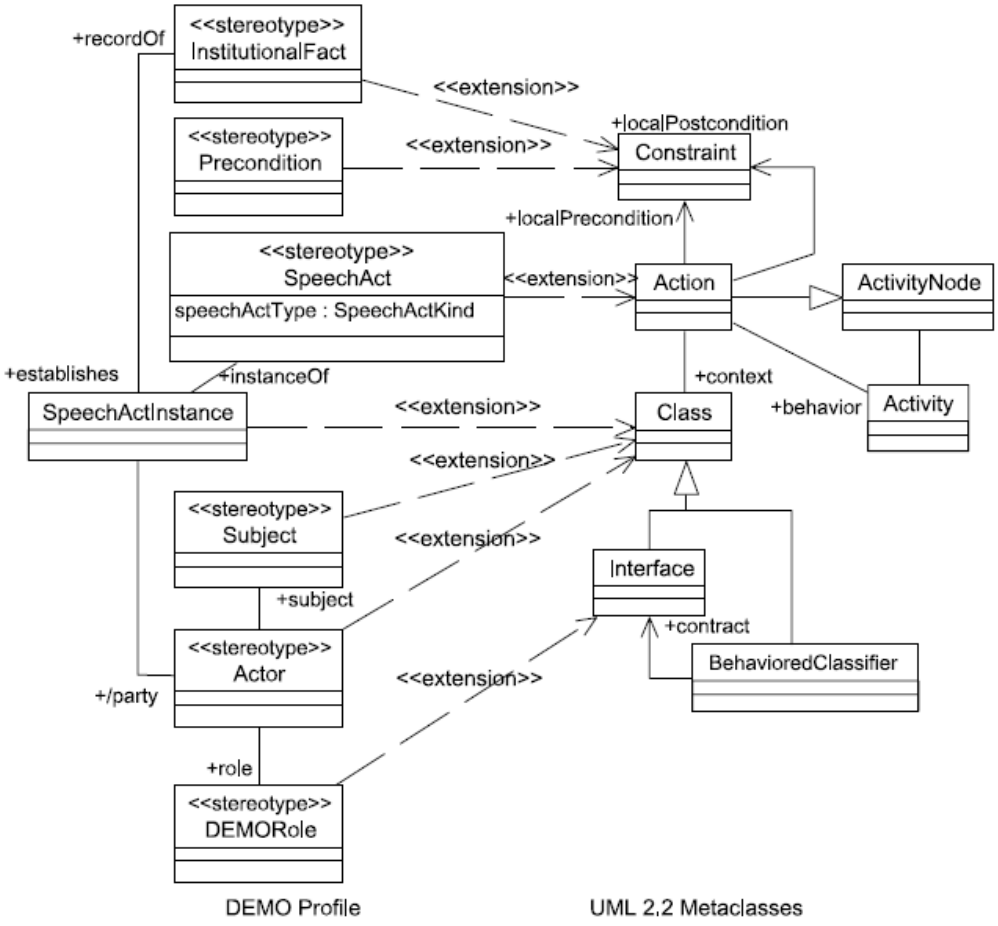

Figure 7-17: DEMO profile (Ahmad et al., 2010)

The following figure shows an activity diagram of ***Registration process*** Using DEMO profile. It shows that every action in the process is a speech act, the diagrams does not differentiate between informative and performative speech acts. Furthermore, the profile does not show details about speech act instance. The profile shows context and pre-condition for speech act, which is very important, for our case study, to track operations with workflow instance, but does not show the status of each speech act, whether it is started, running, blocked or finished.

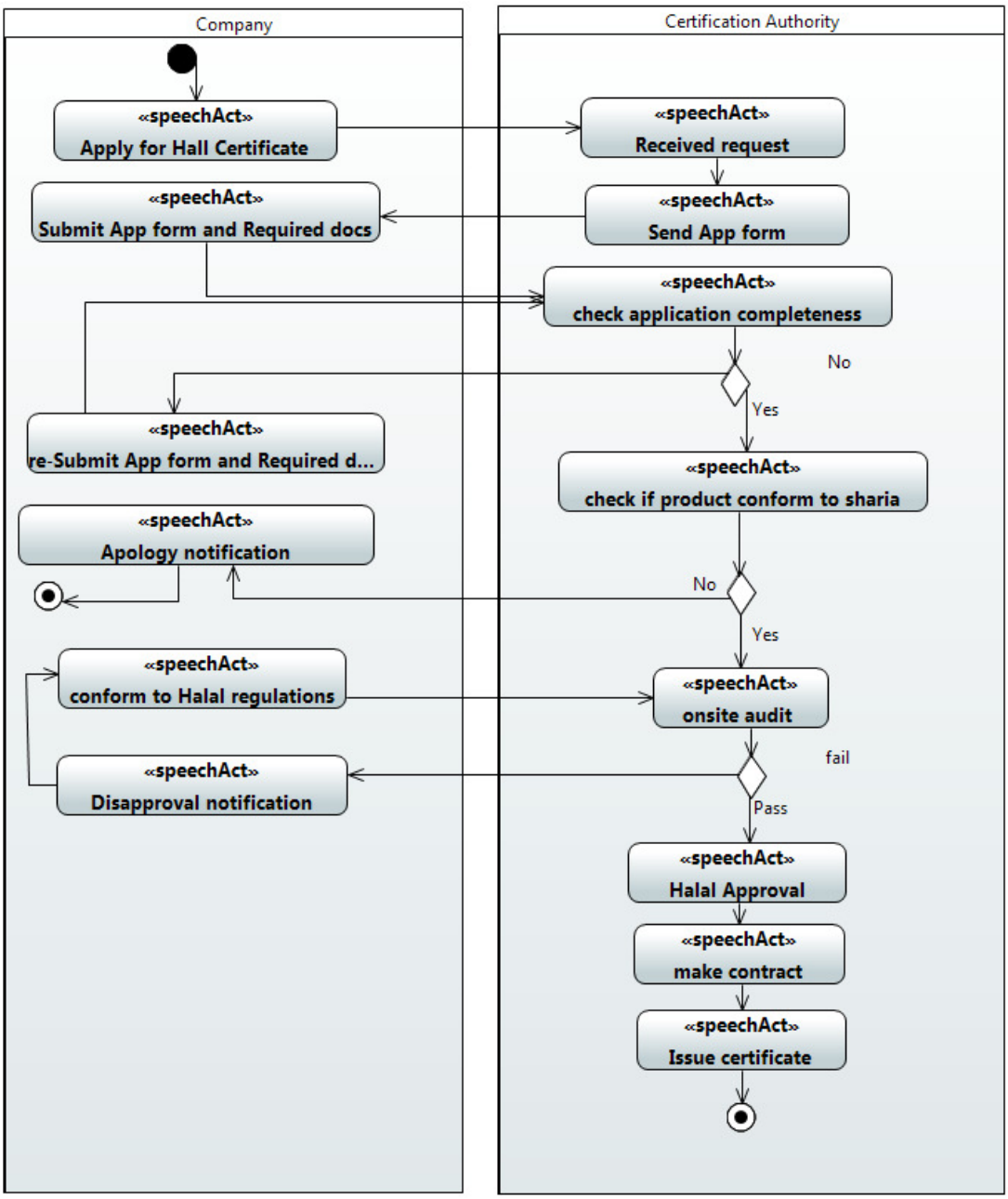

Figure 7-18: Registration process using DEMO profile

7.20. Representing HFOP Using POUP:

The following figures show previously mentioned processes; they represent behavioral view of the case study we have described them using the new terms of POUP profile, every event is represented as speech acts. Figure (7.15) describes issuing halal certificate workflow; company sends directive speech acts to certification authority apply for halal certificate

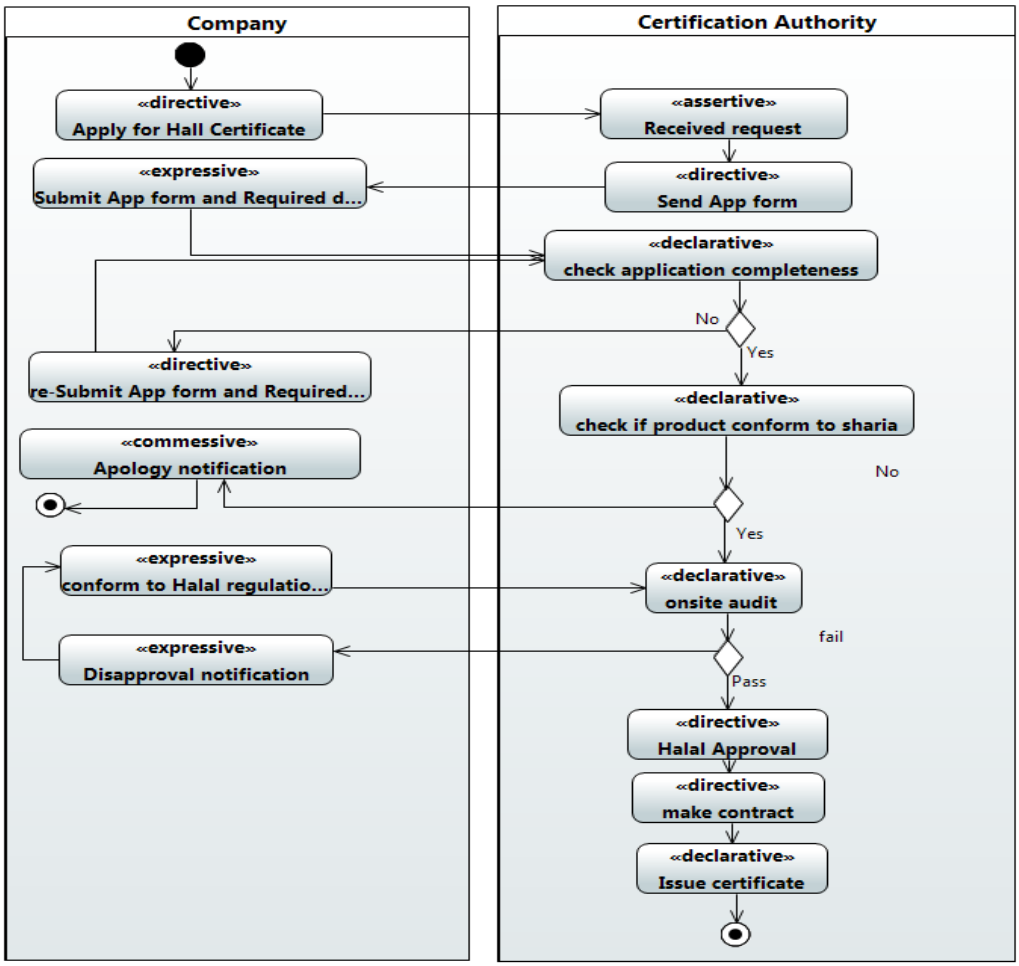

Figure 7-19: Halal Registration Diagram using POUP

Figure (7.19) shows the process of purchasing between tow companies, one represent the buyer and the second represents the seller, two supper speech acts need more investigations, check certificate and check integrity, because they will involve more participants, respectively, halal authority and the central organization which will be substituted with the ontology server. Figure 7.19,20,21 respectively shows work their workflow.

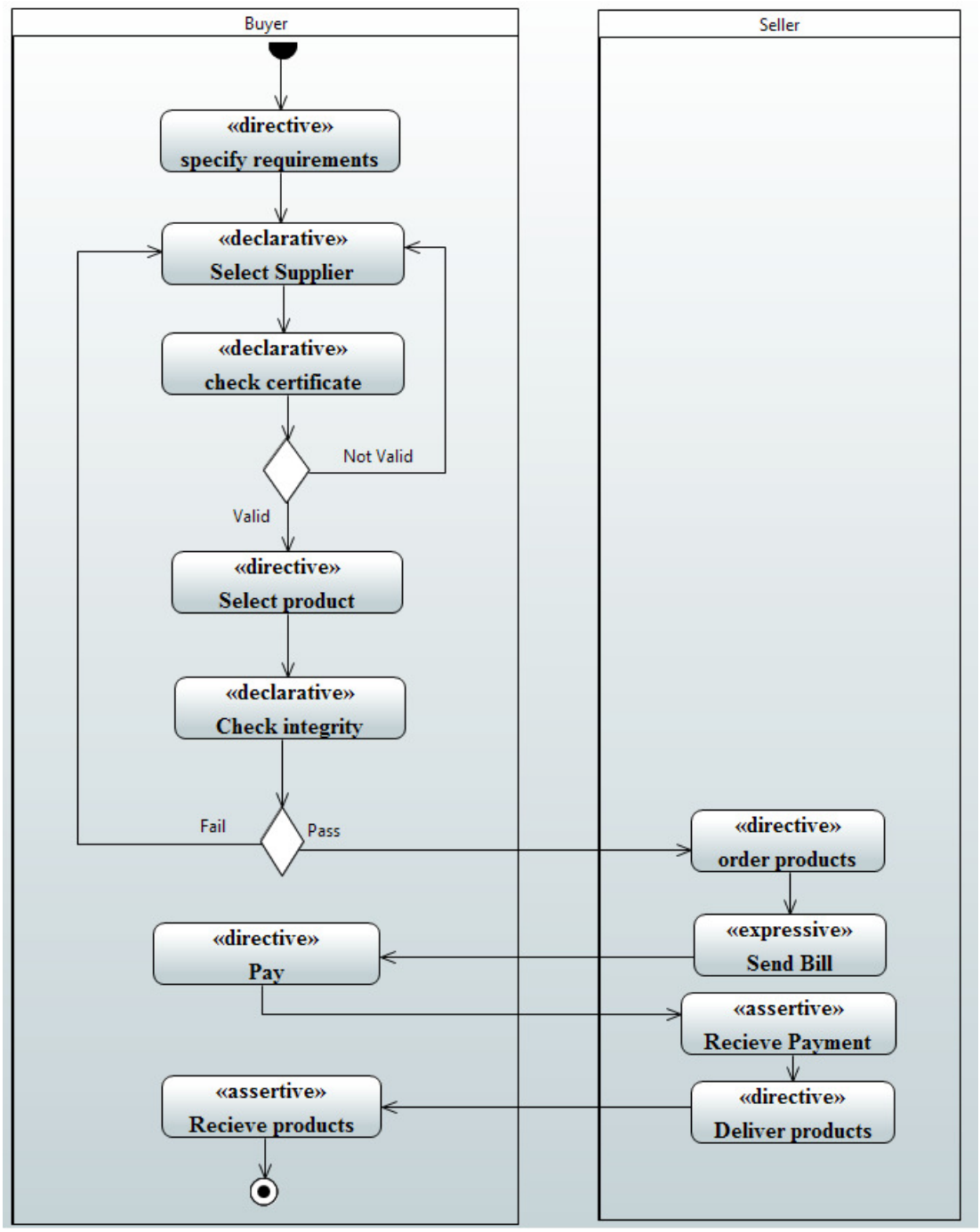

Figure 7-20: Purchasing diagram using POUP

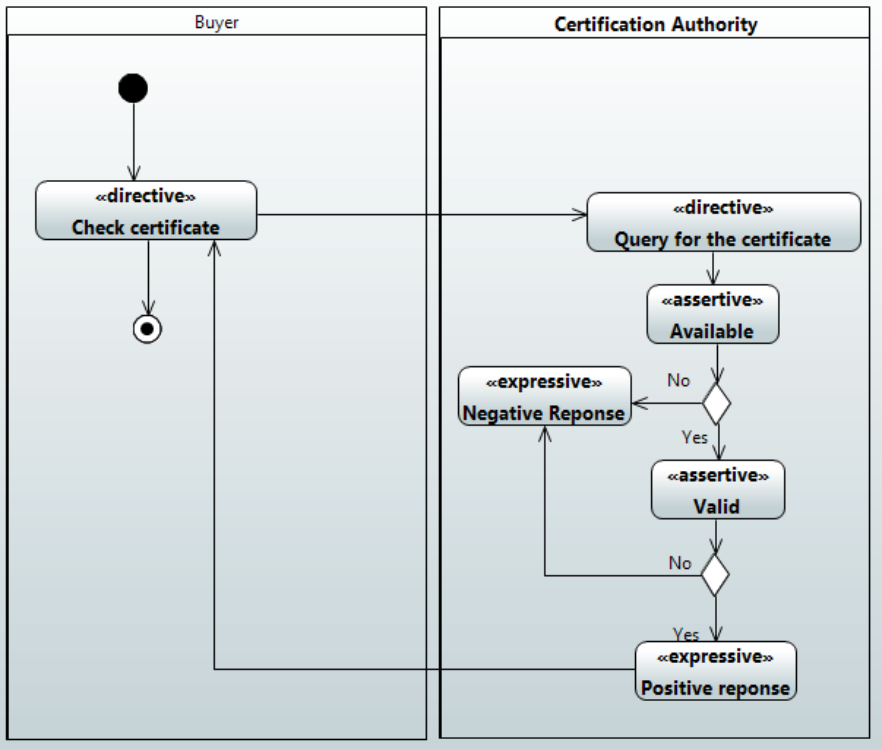

Figure 7-21: check Certificate diagram using POUP

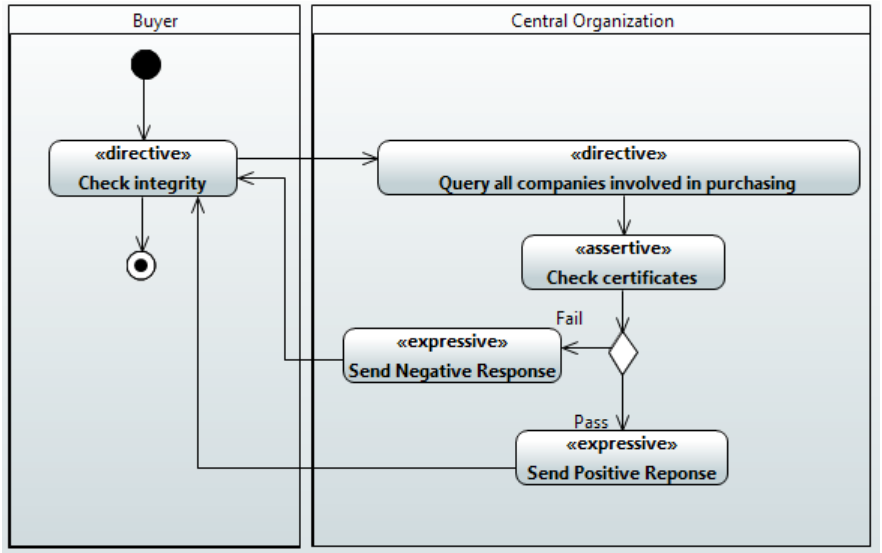

Figure 7-22: checking integrity diagram using POUP

7.21. Running the Case Using the Ontology Server

In this section, we will use the proposed ontology server to run the case study, and we will show how this server could help in insuring HFIWs integrity. We will start by transforming the combination of HFOE and HFOP into OWL using transformation rules addressed in chapter 6. Although OWL does not have concepts

of events and perdurants, but we can use RDFS resource to create classes suitable for representing speech acts as perdurant entities.

7.22. Transformations to OWL

Above models will be transformed into RDF/XML format of OWL to be used in the ontology server as described in transformation rules table in the previous chapter.

Using the ontology server, we can create instances of participants of the case study, which was shown in section (7.11). Once we have the instances of participants and instances of workflows then we can easily query the ontology server. In the following section, we will show some queries.

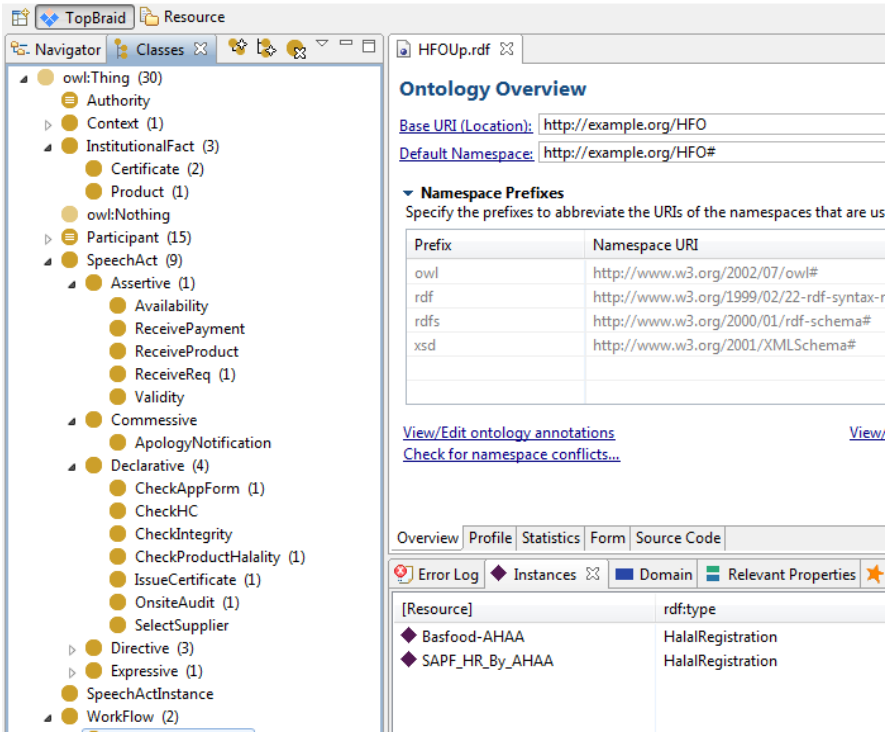

Figure 7-23: Snapshot of Halal Food Ontology (TopBraid)

7.23. Querying The Ontology Server:

Up to now, we have created endurants and perdurants models, and we have transformed then into OWL file and stored them in the ontology repository. Fist we will generate instances of endurant entities (companies, authorities, and products). Secondly, we will run some workflows, some for generating and issuing certificates for some companies, and others for running purchasing. Then we will show some

queries of endurant entities and perdurant entities from the ontology server proposed in the previous chapter.

7.23.1. Query For Halal Certificate:

This query extracts Certification data from HFO. It shows Certificate number, Issue Date, Expire Date, Authority Issued it and its type. Figures 7.23 and 24 show SPAQL query code of java using Jena apache API, and query results respectively. The results are for halal certificate with the number (`u-90-0002022`) entered by the user.

```
public static void queryCertificate(String cert)
{
        String queryString ="PREFIX hf: <http://example.org/HFO#>"+
            "PREFIX i:<http://www.w3.org/2001/XMLSchema#string>"+
            "PREFIX :<http://www.w3.org/2001/XMLSchema#date>"+
            "PREFIX t:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>"+
            "PREFIX l:<http://www.w3.org/2000/01/rdf-schema#>"+
                "SELECT ?Property ?Value " +
              " WHERE {" +
                      "?certif hf:CertificateNo \""+cert+"\" ." +
                      "?certif ?Property ?Value"+
                      "}";

                Query query = QueryFactory.create(queryString);

                // Execute the query and obtain results
                QueryExecution qe = QueryExecutionFactory.create(query, mymodel);
                ResultSet results = qe.execSelect();

                // Output query results
                ResultSetFormatter.out(System.out, results, query);

                // Important - free up resources used running the query
                qe.close();
```

Figure 7-24: SPAQL Java Code Querying H. Certificate

```
-----------------------------------------
| Property          | Value             |
=========================================
| hf:CertificateNo  | "u-90-0002022"^^i:  |
| hf:CerType        | "Mt"^^i:            |
| hf:ExpireDate     | "2015-12-30"^^:     |
| hf:IssueDate      | "2013-12-31"^^:     |
| hf:IssuedBy       | hf:AHAA             |
| t:type            | hf:Certificate      |
```

Figure 7-25: Results of above code

7.23.2. Querying Speech Acts Of Workflow Instance:

This query shows instances of speech acts of specific instance of workflow, for example, the certificate with the number ***u-90-0002022*** above was issued by AHAA, and we can query the ontology for the series of operations occurred leading

to issue this Certificate. The query was executed using the certificate number. Figures 7.25 and 7.24 display the code and results.

Results contains speech act instance, its authority, participant affected by it, and the status. In case of monitoring certificate issuing, Halal Governance can query for the workflow structure (speech acts classes building up the workflow) and workflow speech acts instances and then compare them.

```
public static void querying()
{

    String queryString ="PREFIX hf: <http://example.org/HFO#>"+
            "PREFIX s:<http://www.w3.org/2001/XMLSchema#string>"+
            "PREFIX :<http://www.w3.org/2001/XMLSchema#date>"+
            "PREFIX t:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>"+
            "PREFIX l:<http://www.w3.org/2000/01/rdf-schema#>"+
            "SELECT ?Order ?speechactInstance ?Authority ?Participant_Affected ?Status " +
            " WHERE {" +
            "?n hf:Manipulates hf:Basfood009091 ."+
            "?workflow hf:ComposeOf ?n ."+
            "?workflow hf:ComposeOf ?spact ." +
            "?spact hf:order ?Order ."+
            "?spact hf:SpStatus ?Status ."+
            "?spact hf:HasAuthority ?Authority ."+
            "?spact hf:AffectedParticipant ?Participant_Affected ."+
            "?spact hf:SpeechActTitle ?speechactInstance ."+
            "}"+
            "ORDER BY (?Order)";

            Query query = QueryFactory.create(queryString);

            // Execute the query and obtain results
            QueryExecution qe = QueryExecutionFactory.create(query, mymodel);
            ResultSet results = qe.execSelect();

            // Output query results
            ResultSetFormatter.out(System.out, results, query);

            // Important - free up resources used running the query
            qe.close();
}
```

Figure 7-26: Java Code and SPARQL Query

Order	speechactInstance	Authority	Participant_Affected	Status
1	"Basfood Applying for Halal certificate"^^s:	hf:Basfood	hf:AHAA	"Finished"^^s:
2	"AHAA Received request from Basfood"^^s:	hf:AHAA	hf:Basfood	"Finished"^^s:
3	"Basfood submitted the application form"^^s:	hf:Basfood	hf:AHAA	"Finished"^^s:
4	"AHAA check Basfood's Application form completeness "^^s:	hf:AHAA	hf:Basfood	"Finished"^^s:
6	"AHAA make halal approval for Basfood"^^s:	hf:AHAA	hf:Basfood	"Finished"^^s:
7	"AHAA performed onsit auditing"^^s:	hf:AHAA	hf:Basfood	"Finished"^^s:
8	"AHAA made contract for basfood"^^s:	hf:AHAA	hf:Basfood	"Finished"^^s:
9	"AHAA issue halal certificate for Basfood"^^s:	hf:AHAA	hf:Basfood	"Finished"^^s:

Figure 7-27: Results of above code

7.23.3. Query For Halal Product:

If someone want to query about a specific halal product, he only need to send product number to the ontology server API, the ontology server will replay with

basic product information such as date of production, date of expiration and company that produced the product. Figure 7.27 and 28 shows query code and the results of product *Beef0001*.

```
public static void queryProduct(String PrNO)
{

    String queryString ="PREFIX hf: <http://example.org/HFO#>"+
            "PREFIX s:<http://www.w3.org/2001/XMLSchema#string>"+
            "PREFIX :<http://www.w3.org/2001/XMLSchema#date>"+
            "PREFIX t:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>"+
            "PREFIX l:<http://www.w3.org/2000/01/rdf-schema#>"+
            "SELECT ?Company ?ProductionDate ?ExpireDate " +
            " WHERE {" +
            "hf:Beef0001 hf:ProducedBy ?Com ."+
            "?Com hf:Has_Name ?Company ."+
            "hf:Beef0001 hf:DatOfProduction ?ProductionDate ."+
            "hf:Beef0001 hf:DateOfExpire ?ExpireDate "+
            "}"+
            "ORDER BY (?order)";

            Query query = QueryFactory.create(queryString);

            // Execute the query and obtain results
            QueryExecution qe = QueryExecutionFactory.create(query, mymodel);
            ResultSet results = qe.execSelect();

            // Output query results
            ResultSetFormatter.out(System.out, results, query);

            // Important - free up resources used running the query
            qe.close();
}
}
```

Figure 7-28: product query code

```
-----------------------------------------------------------------------
| Company                       | ProductionDate  | ExpireDate      |
=======================================================================
| "Sydney slaughterhouse "^^s: | "2015-05-25"^^: | "2015-06-09"^^: |
-----------------------------------------------------------------------
```

Figure 7-29: product query results

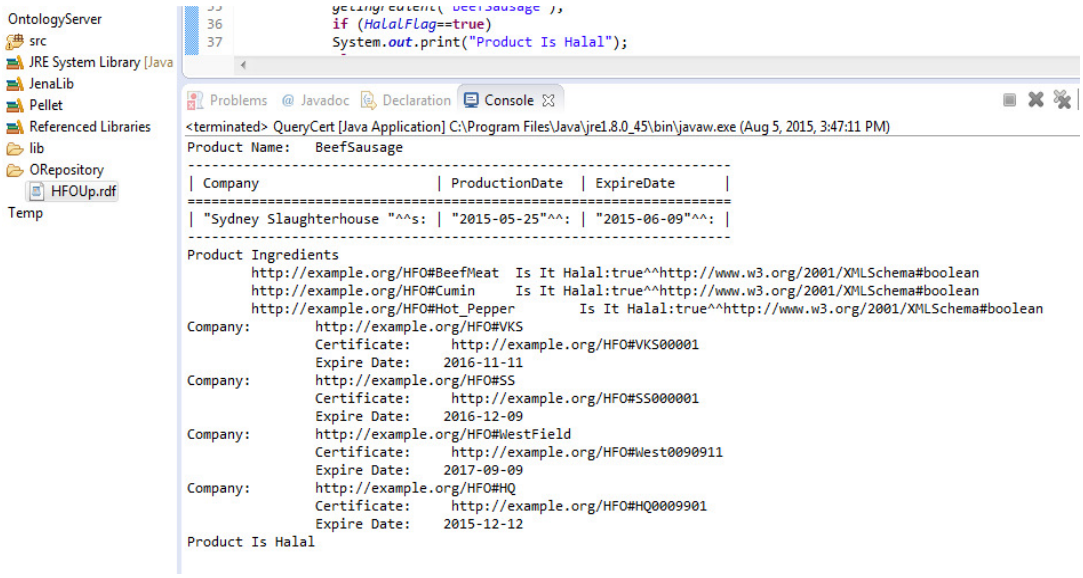

Figure 7-30: Query for product validty and halalness

Figure (7-29) shows an example of Halal product verified by the ontology server. Beef-Sausage was produced by Sydney Slaughter-House (SS) Company in ***25/5/2015*** and will expire in ***9/6/2015***. It compose of Beef-meat, cumin and hot-pepper and all of them are Halal according to Sharia, the product was delivered by 4 company VKS, SS, West-Field, and HQ respectively all these companies has valid certificate. In total the product in halal and valid, in the same time it was travelled via insured and certified companies hence its halal and permissible to be consumed.

7.23.4. Comments on Queries

All queries shown in previous section was created by using SPARQL1.1 via Jena API. These queries are executed upon simple constructed ontology server and has small ontology repository, but it is quite enough to test the functionality of the server. We expect these queries will remain valid even the server has a great amount of data, because ontology could be portioned into number of graphs and could be distributed in different repositories. New technologies of large-triple-store Such as Hadoop-based (H-base) or Jena-H-based could be applied to enlarge ontology repository to store large amount of semantic data and hence enormous of participants could commit to the ontology server and get benefit of its functionalities.

7.24. Evaluation of Case Study

The case study shows that, in halal food IWs there are different type of operations we treat them as speech acts, each of which has specific semantic. Some of them are intended to provide systems with information and do not need response from other intervened systems, while others allow system to make changes and achieve tasks. The last type needs response from other system. POUP has mange to equip the case study with a tool to represent speech acts of the two different types. For example, in halal certificate issuing, the speech act ***Apply for halal certificate*** is directive speech act. This speech act starts the process of halal registration; therefore, it needs response from other involved systems. Another example the speech act ***issue certificate*** which declarative speech act change the status of the firm from unauthorized company to authorized to do business in halal IWs. We think that the

capability of representing these types of operation in IWs is great value in semantic interoperability.

Moreover, the case study shows that, instances of speech acts could be recorded into ontology repository and it would be ready to be used by different authorized participants in halal IWs via the ontology server API.

In comparison with other approaches, such ***DEMO*** profile, ***POUP*** profile can draw expressive behavioral diagrams with more semantics. While the ontology server ***HFOS*** provided the halal IWs participants with an API for query and inference that facilitate and enhance integrity of halal IWs.

7.25. Summary

In this chapter, we have developed a case study for implementing proposed approaches in chapter five and six. First we have developed ontology of both endurants, which was represented as the structural view of halal food IWs, we have make use of ODM profile to show its entities. Ontology of perdurants, which was represented using POUP profile proposed in chapter six. Second, we have make transformation of the two models after merging them together to OWL. Third, we have developed Halal Food Ontology server HFOS, which was composed of semantic web libraries such as apache jean and pellet. The ontology server was able to manage Halal food ontology entities, in adding, deleting, updating. An important result of using the ontology server is the ability of querying the log of transactions (speech acts) which was recorded in the ontology repository, this ability helps in tracking events in halal food interlocking institutional worlds therefore help in insuring the integrity.

CHAPTER 8

CONCLUSIONS AND FUTURE WORK

8.1. Introduction

This chapter discusses some conclusions about the work presented in this thesis. In particular, it provides summary of the results and findings obtained by the research, furthermore we will show how these results lead to realizing the objectives stated in chapter one these are shown (in Section 8.2). The chapter also provides a summary of the most important research contributions of the thesis to knowledge in this field (in Section 8.3) and finally we will suggest a list of related topics for further work (Section 8.4).

8.2. Summary of the results

The aim of this study is to insure that all transactions within the interlocking institutional worlds are achieved perfectly. To carry out this task, there must be a mechanism to record all instances of tasks. Transactions themselves are divided between IWs' participants', therefore, these transactions should be represented as a shared ontology. To represent these transactions and events, there must be a specific modeling profile for representing them. This research has developed a UML profile for representing ontology elements of perdurant types and it also provide a framework for an ontology server capable of managing ontologies of both perdurants and endurants. The study manage to find answers for research questions stated in chapter one, and realize the related objective.

Most research objective was realized, and the research questions have been answered. First, we have established a profile for representing transactions – (POUP), then we have developed the ontology server as a mechanism to govern ontology operations and hold cooperating organizations together in interlocking institutional worlds, and finally we have apply this mechanism in Halal Food IWs, and It has given a promise results. Successfulness of applying this mechanism in Halal Food case study will give a significant evidence to generalize the mechanism to cover other different IWs', for example in Olympic IWs or Drug Supply Chains.

8.3. Research Contributions

This study has contributed in three different directions, to upper ontology it proposes a new taxonomy for what exist in the reality according to the concept of ***Arad*** and ***Gerem***, this taxonomy is very important to distinguish between particular that depend on themselves from those depend on others to appear. The second contribution is to the area of modeling ontologies; the study presented a UML profile for representing perdurant ontology. The profile has the specification of speech act and speech act instances, which enables the profile to represent instances of event and transactions (speech acts). Furthermore, the profile could be use to model speech acts across interlocking institutional worlds with rich semantics, therefore, it facilitate the interoperability between IWs' participants and hence alleviate misunderstanding. The third contribution is the mechanism of managing components of the ontology, of both endurants and perdurants could be managed via usage of ontology server mechanism, which provides functionalities to its participants at design, commit, and run time. At run time, for instance, it can provide functionality to querying data or querying a series of speech acts instances from the ontology, furthermore we can inference new information from existing ontology data and events.

8.4. Future work

The study has managed to give a significant answers for most of research questions stated in chapter one, but still there are some issues left for further research in the same domain. In the following paragraph, we will show some of them:

- Propose profile capable of representing speech acts as sub-type of events of DOLCE upper ontology, representing statives are left for more researches.

- The ontology servers provide functionalities at different stages, at design, commit, and run time, the proposed ontology server provides limited functionalities. More functionality needs to be added, such as viewing a portion of the ontology at commit time, control versioning,

- Mapping between ontology metamodels are out the scope of the study, therefore automating the transformation of POUP models into OWL/RDF/XML was left for future work.

References

ABUGESSAISA, I. E. A. & SIVERTUN, Å. Year. Ontological approach to modeling information systems. *In:* Computer and Information Technology, 2004. CIT'04. The Fourth International Conference on, 2004. IEEE, 1122-1127.

AHMAD, A., MOLLAGHASEMI, M. & RABELO, L. Year. Ontologies for supply chain management. *In:* Proceedings 13th Annual Industrial Engineering Research Conference, Houston, 2004.

AHMAD, M. N. 2009. An Ontology Server for Ontology-based Interoperation of Information Systems. *School of Information Technology and Electrical Engineering, the University of Queensland, Australia*.

AHMAD, M. N., COLOMB, R. M. & SADIQ, S. W. 2010. A UML profile for perdurant ontology of domain interlocking Institutional Worlds. *International Journal of Internet and Enterprise Management,* 6**,** 213-232.

AL-ANDULISY, I. H. 1475. Ibn Hazm treatises *Arabian institution for arabian Studies,* Volume 4**,** 111.

ALTOVA, S. 2006. Visual Semantic Web design tool for RDF and OWL.

AUSTIN, J. L. 1975. *How to do things with words*, Oxford university press.

AUSTIN, J. L. & URMSON, J. 1962. *How to do Things with Words. The William James lectures delivered at Harvard University in 1955.[Edited by James O. Urmson.]*, Clarendon Press.

BACH, K. & HARNISH, R. 1979. Linguistic communication and speech acts.

BERNERS-LEE, T., HENDLER, J. & LASSILA, O. 2001. The semantic web. *Scientific american,* 284**,** 28-37.

BEZIVIN, J. 2005. On the unification power of models. *Software & Systems Modeling,* 4**,** 171-188.

BRICKLEY, D. & GUHA, R. V. 2000. Resource Description Framework (RDF) Schema Specification 1.0: W3C Candidate Recommendation 27 March 2000.

COLOMB, R. M. 2007. *Ontology and the semantic web*, IOS Press.

COLOMB, R. M. & AHMAD, M. N. 2010. A perdurant ontology for interoperating information systems based on interlocking institutional worlds. *Applied Ontology,* 5**,** 47-77.

COMPOSER, T. 2007. TOPBRAID COMPOSER 2007 Features and getting Started Guide Version 1.0, created by TopQuadrant, US. US.

DECTIONARY.COM 2015. Intitution definitions.

DIETZ, J. L. 1999. Understanding and modelling business processes with DEMO. *Conceptual Modeling—ER'99.* Springer.

FARQUHAR, A., FIKES, R. & RICE, J. 1997. The ontolingua server: A tool for collaborative ontology construction. *International journal of human-computer studies,* 46**,** 707-727.

FAVRE, J.-M. Year. Foundations of model (driven)(reverse) engineering: Models. *In:* Proceedings of the International Seminar on Language Engineering for Model-Driven Software Development, Dagstuhl Seminar 04101, 2004a.

FAVRE, J.-M. Year. Towards a basic theory to model model driven engineering. *In:* 3rd Workshop in Software Model Engineering, WiSME, 2004b. Citeseer, 262-271.

FENSEL, D. 2003. Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce. Secaucus. NJ, USA: Springer-Verlag New York, Inc.

GENESERETH, M. 1998. Knowledge Interchange Format draft proposed American National Standard (dpANS) NCITS. T2/98-004, 1998. Available at logic. stanford. edu/-kif/dpans. html.

GOMEZ, J. M., HERNANDEZ, G. A., OLMEDO, J. O. & BUSSLER, C. Year. A B2B conversational architecture for Semantic Web Services based on BPIMS-WS. *In:* Engineering of Complex Computer Systems, 2005. ICECCS 2005. Proceedings. 10th IEEE International Conference on, 2005. IEEE, 252-259.

GRUBER, T. R. 1993. A translation approach to portable ontology specifications. *Knowledge acquisition,* 5**,** 199-220.

GRUBER, T. R. 1995. Toward principles for the design of ontologies used for knowledge sharing? *International journal of human-computer studies,* 43**,** 907-928.

GUARINO, N. 1995. Formal ontology, conceptual analysis and knowledge representation. *International journal of human-computer studies,* 43**,** 625-640.

GUIZZARDI, G. 2005. *Ontological foundations for structural conceptual models*, CTIT, Centre for Telematics and Information Technology.

HEFLIN, J. & HENDLER, J. Year. Dynamic ontologies on the web. *In:* AAAI/IAAI, 2000. 443-449.

HENDLER, J. 2001. Agents and the semantic web. *IEEE Intelligent systems,* 16**,** 30-37.

HERRE, H., HELLER, B., BUREK, P., HOEHNDORF, R., LOEBE, F. & MICHALEK, H. 2006. General Formal Ontology (GFO)–a foundational ontology integrating objects and processes. *Onto-Med Report,* 8.

HORKOFF, J. & MAIDEN, N. 2015. Creativity and Conceptual Modeling for Requirements Engineering.

HOUSE, R. 2015. speech act.

IBRAHIM, A. A., COLOMB, R. M. & AHMED, A. H. 2015. Insuring Halal Food Interlocking Institutional Worlds Integrity Using an Ontology Server.

IBRAHIM, A. A. & SALMAN, N. 2015. Multi-Lingual Ontology Server (MOS) for discovering Web services. *arXiv preprint arXiv:1507.05274*.

INGARDEN, R. 1964. Time and Modes of Being, translated by Helen R. *Michejda, Springfield, Illinois: Charles C. Thomas*.

KABBAJ, A., BOUZOUBA, K., EL HACHIMI, K. & OURDANI, N. 2006. Ontologies in amine platform: structures and processes. *Conceptual Structures: Inspiration and Application.* Springer.

KALFOGLOU, Y. 2001. Exploring ontologies. *Handbook of Software Engineering and Knowledge Engineering,* 1**,** 863-887.

KHAN, J. & KUMAR, S. Year. OWL, RDF, RDFS inference derivation using Jena semantic framework & pellet reasoner. *In:* Advances in Engineering and Technology Research (ICAETR), 2014 International Conference on, 2014. IEEE, 1-8.

KIFER, M., DE BRUIJN, J., BOLEY, H. & FENSEL, D. 2005. A realistic architecture for the semantic web.

KLEIN, M., BROEKSTRA, J., FENSEL, D., VAN HARMELEN, F. & HORROCKS, I. 2003. Ontologies and schema languages on the web. *Spinning the Semantic Web: Bringing the World Wide Web to its full potential***,** 95-140.

KLYNE, G. & CARROLL, J. J. 2006. Resource description framework (RDF): Concepts and abstract syntax.

LENAT, D. B. 1995. CYC: A large-scale investment in knowledge infrastructure. *Communications of the ACM,* 38**,** 33-38.

MARTIN, D., BURSTEIN, M., HOBBS, J., LASSILA, O., MCDERMOTT, D., MCILRAITH, S., NARAYANAN, S., PAOLUCCI, M., PARSIA, B. & PAYNE, T.

2004. OWL-S: Semantic markup for web services. *W3C member submission,* 22**,** 2007-04.

MASOLO, C., BORGO, S., GANGEMI, A., GUARINO, N. & OLTRAMARI, A. 2003a. Ontology Library. WonderWeb Deliverable D18 (ver. 1.0, 31-12-2003). Technical report, Laboratory For Applied Ontology-ISTCCNR.

MASOLO, C., BORGO, S., GANGEMI, A., GUARINO, N., OLTRAMARI, A. & SCHNEIDER, L. 2003b. The WonderWeb library of foundational ontologies preliminary report WonderWeb deliverable D17. *ISTC-CNR, Padua, Italy*.

MCGUINNESS, D. L. & VAN HARMELEN, F. 2004. OWL web ontology language overview. *W3C recommendation,* 10**,** 2004.

MILLER, G. & FELLBAUM, C. 1998. Wordnet: An electronic lexical database. MIT Press Cambridge.

NOY, N. F., SINTEK, M., DECKER, S., CRUBEZY, M., FERGERSON, R. W. & MUSEN, M. A. 2001. Creating semantic web contents with protege-2000. *IEEE Intelligent systems***,** 60-71.

ODM, O. 2007. Ontology Definition Metamodel–OMG Adopted Specification. OMG Document ptc/2007-09-09 [Online]. Available: http://www. omg. org/cgi-bin/apps/doc.

OMG 2000. Unified Modeling Language (UML) Version 1.3. *OMG web site www.omg.org, march 2000*.

OMG 2003. Model Driven Architecture (MDA) Guide v1. 0.1 [documento en línea]. 2003a.

OMG 2007. Group. OMG Unified Modeling Language (OMG UML), Infrastructure. V2. 1.2. Technical report.

OMG, Q. 2008. Meta object facility (mof) 2.0 query/view/transformation specification. *Final Adopted Specification (November 2005)*.

PERISIC, B. 2014. Model Driven Software Development-State of the Art and Perspectives. *Invited Paper, INFOTEH***,** 1237-1248.

SCHMIDT, D. C. 2006. Guest editor's introduction: Model-driven engineering. *Computer,* 39**,** 0025-31.

SEARLE, J. R. 1969. *Speech acts: An essay in the philosophy of language*, Cambridge university press.

SEARLE, J. R. 1985. *Expression and meaning: Studies in the theory of speech acts*, Cambridge University Press.

SEARLE, J. R. 1995. *The construction of social reality*, Simon and Schuster.

SEGERS, R., VOSSEN, P., ROSPOCHER, M., SERAFINI, L., LAPARRA, E. & RIGAU, G. 2015. ESO: a Frame based Ontology for Events and Implied Situations. *Proceedings of Maplex2015*.

SEIDEWITZ, E. 2003. What models mean. *IEEE software*, 26-32.

SELIC, B. 2003. The pragmatics of model-driven development. *IEEE software*, 19-25.

SMITH, B. 2003. Ontology.

SMITH, B. & GRENON, P. 2002. Basic formal ontology. *Draft. Downloadable at http://ontology. buffalo. edu/bfo*.

STUDER, R., BENJAMINS, V. R. & FENSEL, D. 1998. Knowledge engineering: principles and methods. *Data & knowledge engineering,* 25, 161-197.

SWARTOUT, W. & TATE, A. 1999. Guest editors' introduction: Ontologies. *IEEE Intelligent systems,* 14, 18-19.

VIINIKKALA, M. 2005. Ontology in Information Systems. Citeseer.

W3C, S. W. 2007. *TopBraid Suite* [Online]. Available: https://www.w3.org/2001/sw/wiki/TopBraid [Accessed 8 june 2015].

WEITEN, M. 2009. *Ontostudio® as a ontology engineering environment*, Springer.

YE, Y., YANG, D., JIANG, Z. & TONG, L. 2008. Ontology-based semantic models for supply chain management. *The International Journal of Advanced Manufacturing Technology,* 37, 1250-1260.

ZAILANI, S., ARRIFIN, Z., ABD WAHID, N., OTHMAN, R. & FERNANDO, Y. 2010. Halal traceability and halal tracking systems in strengthening halal food supply chains for food industry in Malaysia (a review). *Journal of food Technology,* 8, 74-81.

Appendixes

RDF/XML format of Halal Food Ontology (Endurants + Perdurants)

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns="http://example.org/HFO#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xml:base="http://example.org/HFO">
  <owl:Ontology rdf:about="">
    <owl:versionInfo rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Created with TopBraid Composer</owl:versionInfo>
  </owl:Ontology>
  <owl:Class rdf:ID="InstitutionalFact"/>
  <owl:Class rdf:ID="HalalRegistration">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="WorkFlow"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Pay">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Directive"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="DeliverProducts">
    <rdfs:subClassOf>
      <owl:Class rdf:about="#Directive"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="submitAppForm">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Expressive"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="IssueCertificate">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Declarative"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Certification_Authority">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="HalalAuthority"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="ReceiveProduct">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Assertive"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="ReSubmit_AppForm">
```

```
  <rdfs:subClassOf>
    <owl:Class rdf:about="#Directive"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="Commessive">
  <rdfs:subClassOf>
    <owl:Class rdf:ID="SpeechAct"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="Availability">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#Assertive"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="Halal_Governance">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#HalalAuthority"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="#HalalAuthority">
  <rdfs:subClassOf>
    <owl:Class rdf:ID="Participant"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="SendAppForm">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#Directive"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="Purchasing">
  <rdfs:subClassOf rdf:resource="#WorkFlow"/>
</owl:Class>
<owl:Class rdf:ID="Distributer">
  <rdfs:subClassOf>
    <owl:Class rdf:ID="Company"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="#Participant">
  <owl:equivalentClass>
    <owl:Class rdf:ID="Authority"/>
  </owl:equivalentClass>
</owl:Class>
<owl:Class rdf:ID="CheckAppForm">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#Declarative"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="SpeechActInstance"/>
<owl:Class rdf:ID="ReceiveReq">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#Assertive"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="Retailer">
  <rdfs:subClassOf>
```

```
    <owl:Class rdf:about="#Company"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="#Directive">
  <rdfs:subClassOf rdf:resource="#SpeechAct"/>
</owl:Class>
<owl:Class rdf:about="#Declarative">
  <rdfs:subClassOf rdf:resource="#SpeechAct"/>
</owl:Class>
<owl:Class rdf:ID="Context"/>
<owl:Class rdf:ID="Farm">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#Company"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="ReceivePayment">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#Assertive"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="CheckIntegrity">
  <rdfs:subClassOf rdf:resource="#Declarative"/>
</owl:Class>
<owl:Class rdf:ID="Disapproval_notification">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#Expressive"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="MakeContract">
  <rdfs:subClassOf rdf:resource="#Directive"/>
</owl:Class>
<owl:Class rdf:ID="SpecifyRequirements">
  <rdfs:subClassOf rdf:resource="#Directive"/>
</owl:Class>
<owl:Class rdf:ID="QueryAllCompsInvolved">
  <rdfs:subClassOf rdf:resource="#Directive"/>
</owl:Class>
<owl:Class rdf:ID="Processing_Plant">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#Company"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="Conform2HalalRegulation">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#Expressive"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="PostCondition">
  <rdfs:subClassOf rdf:resource="#Context"/>
</owl:Class>
<owl:Class rdf:ID="SelectProduct">
  <rdfs:subClassOf rdf:resource="#Directive"/>
</owl:Class>
<owl:Class rdf:about="#Expressive">
  <rdfs:subClassOf rdf:resource="#SpeechAct"/>
```

```
</owl:Class>
<owl:Class rdf:ID="NegativeResponse">
  <rdfs:subClassOf rdf:resource="#Expressive"/>
</owl:Class>
<owl:Class rdf:ID="HalalApproval">
  <rdfs:subClassOf rdf:resource="#Directive"/>
</owl:Class>
<owl:Class rdf:ID="PositiveReponse">
  <rdfs:subClassOf rdf:resource="#Expressive"/>
</owl:Class>
<owl:Class rdf:ID="Apply_For_HC">
  <rdfs:subClassOf rdf:resource="#Directive"/>
</owl:Class>
<owl:Class rdf:ID="Validity">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#Assertive"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="OnsiteAudit">
  <rdfs:subClassOf rdf:resource="#Declarative"/>
</owl:Class>
<owl:Class rdf:about="#Assertive">
  <rdfs:subClassOf rdf:resource="#SpeechAct"/>
</owl:Class>
<owl:Class rdf:ID="ApologyNotification">
  <rdfs:subClassOf rdf:resource="#Commessive"/>
</owl:Class>
<owl:Class rdf:ID="Product">
  <rdfs:subClassOf rdf:resource="#InstitutionalFact"/>
</owl:Class>
<owl:Class rdf:ID="OrderProducts">
  <rdfs:subClassOf rdf:resource="#Directive"/>
</owl:Class>
<owl:Class rdf:ID="SelectSupplier">
  <rdfs:subClassOf rdf:resource="#Declarative"/>
</owl:Class>
<owl:Class rdf:ID="PreCondition">
  <rdfs:subClassOf rdf:resource="#Context"/>
</owl:Class>
<owl:Class rdf:ID="Certificate">
  <rdfs:subClassOf rdf:resource="#InstitutionalFact"/>
</owl:Class>
<owl:Class rdf:ID="Query4HC">
  <rdfs:subClassOf rdf:resource="#Directive"/>
</owl:Class>
<owl:Class rdf:ID="CheckProductHalality">
  <rdfs:subClassOf rdf:resource="#Declarative"/>
</owl:Class>
<owl:Class rdf:about="#Company">
  <rdfs:subClassOf rdf:resource="#Participant"/>
</owl:Class>
<owl:Class rdf:ID="SendBill">
  <rdfs:subClassOf rdf:resource="#Expressive"/>
</owl:Class>
<owl:Class rdf:about="#Authority">
```

```
  <owl:equivalentClass rdf:resource="#Participant"/>
</owl:Class>
<owl:Class rdf:ID="CheckHC">
  <rdfs:subClassOf rdf:resource="#Declarative"/>
</owl:Class>
<owl:Class rdf:ID="WholeSaler">
  <rdfs:subClassOf rdf:resource="#Company"/>
</owl:Class>
<owl:ObjectProperty rdf:ID="Certifies">
  <owl:inverseOf>
    <owl:ObjectProperty rdf:ID="CertifiedBy"/>
  </owl:inverseOf>
  <rdfs:domain rdf:resource="#Certification_Authority"/>
  <rdfs:range rdf:resource="#Company"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="IssuedBy">
  <rdfs:range rdf:resource="#Certification_Authority"/>
  <rdfs:domain rdf:resource="#Certificate"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="Buys">
  <owl:inverseOf>
    <owl:ObjectProperty rdf:ID="Sell"/>
  </owl:inverseOf>
  <rdfs:range rdf:resource="#Product"/>
  <rdfs:domain rdf:resource="#Company"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="Has_Context">
  <rdfs:domain rdf:resource="#SpeechAct"/>
  <rdfs:range rdf:resource="#Context"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="ComposeOf">
  <rdfs:domain rdf:resource="#WorkFlow"/>
  <rdfs:range rdf:resource="#SpeechAct"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="AffectedParticipant">
  <rdfs:domain rdf:resource="#SpeechAct"/>
  <rdfs:range rdf:resource="#Participant"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="ProducedBy">
  <rdfs:domain rdf:resource="#Product"/>
  <rdfs:range rdf:resource="#Processing_Plant"/>
  <owl:inverseOf>
    <owl:ObjectProperty rdf:ID="Produces"/>
  </owl:inverseOf>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#Sell">
  <owl:inverseOf rdf:resource="#Buys"/>
  <rdfs:range rdf:resource="#Product"/>
  <rdfs:domain rdf:resource="#Company"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="Manipulates">
  <rdfs:range rdf:resource="#InstitutionalFact"/>
  <rdfs:domain rdf:resource="#SpeechAct"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="HasAuthority">
```

```
  <rdfs:domain rdf:resource="#SpeechAct"/>
  <rdfs:range rdf:resource="#Participant"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#Produces">
  <rdfs:domain rdf:resource="#Processing_Plant"/>
  <rdfs:range rdf:resource="#Product"/>
  <owl:inverseOf rdf:resource="#ProducedBy"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="Governs">
  <rdfs:range rdf:resource="#Certification_Authority"/>
  <rdfs:domain rdf:resource="#Halal_Governance"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="HasCertificate">
  <rdfs:range rdf:resource="#Certificate"/>
  <rdfs:domain rdf:resource="#Company"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#CertifiedBy">
  <rdfs:range rdf:resource="#Certification_Authority"/>
  <rdfs:domain rdf:resource="#Company"/>
  <owl:inverseOf rdf:resource="#Certifies"/>
</owl:ObjectProperty>
<rdf:Property rdf:ID="DatOfProduction">
  <rdfs:domain rdf:resource="#Product"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#date"/>
</rdf:Property>
<rdf:Property rdf:ID="DateOfExpire">
  <rdfs:domain rdf:resource="#Product"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#date"/>
</rdf:Property>
<rdf:Property rdf:ID="order">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#integer"/>
  <rdfs:domain rdf:resource="#SpeechAct"/>
</rdf:Property>
<owl:DatatypeProperty rdf:ID="CompanyType">
  <rdfs:domain rdf:resource="#Company"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="HasRegNo">
  <rdfs:domain rdf:resource="#Participant"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="SpeechActTitle">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#SpeechAct"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="Has_Name">
  <rdfs:domain rdf:resource="#Participant"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="CerType">
  <rdfs:domain rdf:resource="#Certificate"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="IssueDate">
  <rdfs:domain rdf:resource="#Certificate"/>
```

```
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#date"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="ExpireDate">
  <rdfs:domain rdf:resource="#Certificate"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#date"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="SpStatus">
  <rdfs:domain rdf:resource="#SpeechAct"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="CertificateNo">
  <rdfs:domain rdf:resource="#Certificate"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="HasRno">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#boolean"/>
  <rdfs:domain rdf:resource="#PreCondition"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="Countery">
  <rdfs:domain rdf:resource="#Participant"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="wfLSN">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#SpeechActInstance"/>
</owl:DatatypeProperty>
<Retailer rdf:ID="Kaki_lima">
  <Has_Name rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >Kaki Lima</Has_Name>
  <HasRegNo rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >u-223300001</HasRegNo>
  <Countery rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >Australia</Countery>
  <CompanyType rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >mid</CompanyType>
  <CertifiedBy>
    <Certification_Authority rdf:ID="HA">
      <Certifies>
        <Processing_Plant rdf:ID="SS">
          <Produces>
            <Product rdf:ID="Beef0001">
              <DateOfExpire rdf:datatype="http://www.w3.org/2001/XMLSchema#date"
              >2015-06-09</DateOfExpire>
              <DatOfProduction rdf:datatype="http://www.w3.org/2001/XMLSchema#date"
              >2015-05-25</DatOfProduction>
              <ProducedBy rdf:resource="#SS"/>
            </Product>
          </Produces>
          <HasRegNo rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
          >au-120-00090909</HasRegNo>
          <CertifiedBy rdf:resource="#HA"/>
          <Has_Name rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
          >Sydney slaughterhouse </Has_Name>
        </Processing_Plant>
      </Certifies>
```

```
      <Certifies>
        <WholeSaler rdf:ID="HC">
          <HasRegNo rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
          >au20-220002</HasRegNo>
          <Countery rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
          >Australia</Countery>
          <CompanyType rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
          >Small Business</CompanyType>
          <CertifiedBy rdf:resource="#HA"/>
          <Has_Name rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
          >Halal Choice </Has_Name>
        </WholeSaler>
      </Certifies>
      <Countery rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >Australia</Countery>
      <HasRegNo rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >au-20-000012</HasRegNo>
      <Has_Name rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >Halal Australian </Has_Name>
    </Certification_Authority>
  </CertifiedBy>
</Retailer>
<Retailer rdf:ID="Ash_Sedap">
  <Has_Name rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >Ash Sedap</Has_Name>
  <HasRegNo rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >90909992222</HasRegNo>
  <Countery rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >Australia</Countery>
  <CompanyType rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >mid</CompanyType>
  <CertifiedBy>
    <Certification_Authority rdf:ID="AIM">
      <Certifies>
        <WholeSaler rdf:ID="HQ">
          <CompanyType rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
          >mid</CompanyType>
          <HasCertificate>
            <Certificate rdf:ID="HQ0009901">
              <IssuedBy rdf:resource="#AIM"/>
              <CerType rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
              >MT</CerType>
              <IssueDate rdf:datatype="http://www.w3.org/2001/XMLSchema#date"
              >2013-12-12</IssueDate>
              <ExpireDate rdf:datatype="http://www.w3.org/2001/XMLSchema#date"
              >2015-12-12</ExpireDate>
              <CertificateNo rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
              >1</CertificateNo>
            </Certificate>
          </HasCertificate>
          <HasRegNo rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
          >au-20-200032</HasRegNo>
          <Countery rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
          >Australia</Countery>
          <CertifiedBy rdf:resource="#AIM"/>
```

```
        <Has_Name rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >Halal Square </Has_Name>
       </WholeSaler>
      </Certifies>
      <Has_Name rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >Australian Islamic Monitor</Has_Name>
      <HasRegNo rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >u-0005641</HasRegNo>
      <Countery rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >Australia</Countery>
      <Certifies>
       <WholeSaler rdf:ID="StockLand">
        <Has_Name rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >Stock Land </Has_Name>
        <HasRegNo rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >U-20-00090011</HasRegNo>
        <Countery rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >Australia</Countery>
        <CompanyType rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >large</CompanyType>
        <CertifiedBy rdf:resource="#AIM"/>
       </WholeSaler>
      </Certifies>
      <Certifies rdf:resource="#SS"/>
     </Certification_Authority>
    </CertifiedBy>
  </Retailer>
  <Halal_Governance rdf:ID="AHDA">
   <Has_Name rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >Australian Halal Development and Accreditation </Has_Name>
   <HasRegNo rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >908a-900</HasRegNo>
   <Governs rdf:resource="#HA"/>
   <Governs rdf:resource="#AIM"/>
   <Governs>
    <Certification_Authority rdf:ID="AHAA">
     <Has_Name rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
     >Australian Halal Authority and Advisers </Has_Name>
     <HasRegNo rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
     >au20-220001</HasRegNo>
     <Countery rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
     >Australia</Countery>
     <Certifies>
      <Farm rdf:ID="SAPF">
       <HasRegNo rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
       >U-20-200012</HasRegNo>
       <Countery rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
       >Australia</Countery>
       <CompanyType rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
       >Limited</CompanyType>
       <CertifiedBy rdf:resource="#AHAA"/>
       <Has_Name rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
       >Sheep Australia Pty Farms </Has_Name>
      </Farm>
     </Certifies>
```

```
    </Certification_Authority>
  </Governs>
</Halal_Governance>
<HalalRegistration rdf:ID="SAPF_HR_By_AHAA"/>
<Farm rdf:ID="VKS">
  <HasRegNo rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >U-20-2000010</HasRegNo>
  <Countery rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >Australia</Countery>
  <CompanyType rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >large</CompanyType>
  <CertifiedBy rdf:resource="#AIM"/>
  <Has_Name rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >VKS farms </Has_Name>
</Farm>
<HalalRegistration rdf:ID="Basfood-AHAA">
  <ComposeOf>
    <Apply_For_HC rdf:ID="Basfood_Apply_For_HC">
      <SpStatus rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >Finished</SpStatus>
      <SpeechActTitle rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >Basfood Applying for Halal certificate</SpeechActTitle>
      <HasAuthority>
        <Distributer rdf:ID="Basfood">
          <HasRegNo rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
          >09222222901</HasRegNo>
          <Countery rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
          >Australia</Countery>
          <CompanyType rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
          >large</CompanyType>
          <Has_Name rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
          >Basfood limited</Has_Name>
          <HasCertificate>
            <Certificate rdf:ID="Basfood009091">
              <IssuedBy rdf:resource="#AHAA"/>
              <IssueDate rdf:datatype="http://www.w3.org/2001/XMLSchema#date"
              >2013-12-31</IssueDate>
              <ExpireDate rdf:datatype="http://www.w3.org/2001/XMLSchema#date"
              >2015-12-30</ExpireDate>
              <CerType rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
              >Mt</CerType>
              <CertificateNo rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
              >u-90-0002022</CertificateNo>
            </Certificate>
          </HasCertificate>
          <CertifiedBy rdf:resource="#AHAA"/>
        </Distributer>
      </HasAuthority>
      <AffectedParticipant rdf:resource="#AHAA"/>
      <Has_Context>
        <PreCondition rdf:ID="PreCondition_1">
          <HasRno rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean"
          >true</HasRno>
        </PreCondition>
      </Has_Context>
```

```
    <order rdf:datatype="http://www.w3.org/2001/XMLSchema#integer"
    >1</order>
  </Apply_For_HC>
</ComposeOf>
<ComposeOf>
  <MakeContract rdf:ID="AHAAMakeContract_4Basfood">
    <SpStatus rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Finished</SpStatus>
    <SpeechActTitle rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >AHAA made contract for basfood</SpeechActTitle>
    <AffectedParticipant rdf:resource="#Basfood"/>
    <HasAuthority rdf:resource="#AHAA"/>
    <Has_Context rdf:resource="#PreCondition_1"/>
    <order rdf:datatype="http://www.w3.org/2001/XMLSchema#integer"
    >8</order>
  </MakeContract>
</ComposeOf>
<ComposeOf>
  <CheckProductHalality rdf:ID="AHAA-CheckProductHalality-ofBasfood">
    <SpStatus rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Finished</SpStatus>
    <HasAuthority rdf:resource="#AHAA"/>
    <Has_Context rdf:resource="#PreCondition_1"/>
    <order rdf:datatype="http://www.w3.org/2001/XMLSchema#integer"
    >5</order>
    <SpeechActTitle rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >AHAA checks Basfood prducts if its conform to Halal specification</SpeechActTitle>
  </CheckProductHalality>
</ComposeOf>
<ComposeOf>
  <OnsiteAudit rdf:ID="AHAAOnsiteAuditBasfood">
    <SpStatus rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Finished</SpStatus>
    <SpeechActTitle rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >AHAA performed onsit auditing</SpeechActTitle>
    <AffectedParticipant rdf:resource="#Basfood"/>
    <HasAuthority rdf:resource="#AHAA"/>
    <Has_Context rdf:resource="#PreCondition_1"/>
    <order rdf:datatype="http://www.w3.org/2001/XMLSchema#integer"
    >7</order>
  </OnsiteAudit>
</ComposeOf>
<ComposeOf>
  <HalalApproval rdf:ID="AHAAHalalApprovalBasfood">
    <SpStatus rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Finished</SpStatus>
    <SpeechActTitle rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >AHAA make halal approval for Basfood</SpeechActTitle>
    <AffectedParticipant rdf:resource="#Basfood"/>
    <HasAuthority rdf:resource="#AHAA"/>
    <Has_Context rdf:resource="#PreCondition_1"/>
    <order rdf:datatype="http://www.w3.org/2001/XMLSchema#integer"
    >6</order>
  </HalalApproval>
</ComposeOf>
```

```
  <ComposeOf>
    <IssueCertificate rdf:ID="AHAAIssueCertificate_4Basfood">
      <SpStatus rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >Finished</SpStatus>
      <SpeechActTitle rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >AHAA issue halal certificate for Basfood</SpeechActTitle>
      <AffectedParticipant rdf:resource="#Basfood"/>
      <HasAuthority rdf:resource="#AHAA"/>
      <Has_Context rdf:resource="#PreCondition_1"/>
      <Manipulates rdf:resource="#Basfood009091"/>
      <order rdf:datatype="http://www.w3.org/2001/XMLSchema#integer"
      >9</order>
    </IssueCertificate>
  </ComposeOf>
  <ComposeOf>
    <CheckAppForm rdf:ID="AHAACheckAppForm_4_Basfood">
      <SpStatus rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >Finished</SpStatus>
      <AffectedParticipant rdf:resource="#Basfood"/>
      <HasAuthority rdf:resource="#AHAA"/>
      <Has_Context rdf:resource="#PreCondition_1"/>
      <order rdf:datatype="http://www.w3.org/2001/XMLSchema#integer"
      >4</order>
      <SpeechActTitle rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >AHAA check Basfood's Application form completeness </SpeechActTitle>
    </CheckAppForm>
  </ComposeOf>
  <ComposeOf>
    <ReceiveReq rdf:ID="AHAA-Receive_BasfoodReq">
      <SpStatus rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >Finished</SpStatus>
      <HasAuthority rdf:resource="#AHAA"/>
      <Has_Context rdf:resource="#PreCondition_1"/>
      <AffectedParticipant rdf:resource="#Basfood"/>
      <order rdf:datatype="http://www.w3.org/2001/XMLSchema#integer"
      >2</order>
      <SpeechActTitle rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >AHAA Received request from Basfood</SpeechActTitle>
    </ReceiveReq>
  </ComposeOf>
  <ComposeOf>
    <submitAppForm rdf:ID="Basfood-submitAppForm">
      <SpStatus rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >Finished</SpStatus>
      <SpeechActTitle rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >Basfood submitted the application form</SpeechActTitle>
      <AffectedParticipant rdf:resource="#AHAA"/>
      <HasAuthority rdf:resource="#Basfood"/>
      <Has_Context rdf:resource="#PreCondition_1"/>
      <order rdf:datatype="http://www.w3.org/2001/XMLSchema#integer"
      >3</order>
    </submitAppForm>
  </ComposeOf>
</HalalRegistration>
<Processing_Plant rdf:ID="instance6"/>
```

```
  <Retailer rdf:ID="WestField">
    <CertifiedBy rdf:resource="#AHAA"/>
    <CompanyType rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >mid</CompanyType>
    <Countery rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Austrakia</Countery>
    <HasRegNo rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >u-20-11022210</HasRegNo>
  </Retailer>
</rdf:RDF>
```